EXTENDING THE SENSING OF ONLINE SOCIAL NETWORKS – A HEALTHCARE PERSPECTIVE

Asma' Wasfi Fayez Mustafa

UAEU College of Graduate Studies

جامعة الإمارات العربية المتحدة
United Arab Emirates University

United Arab Emirates University

College of Engineering

Department of Electrical Engineering

EXTENDING THE SENSING OF ONLINE SOCIAL NETWORKS – A HEALTHCARE PERSPECTIVE

Asma' Wasfi Fayez Mustafa

This thesis is submitted in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering

Under the Supervision of Dr. Qurban Memon

November 2014

Declaration of Original Work

I, Asma' Wasfi Mustafa, the undersigned, a graduate student at the United Arab Emirates University (UAEU) and the author of the thesis, entitled "Extending the Sensing of Online Social Networks – A Healthcare Perspective", Hereby solemnly declare that this thesis is an original work done and prepared by me under the supervision of Dr. Qurban Memon, in the College of Engineering at UAEU. This work has not been previously formed as the basis for the award of any degree, diploma or a similar title at this or any other university. The materials borrowed from other sources and included in my thesis have been properly cited and acknowledged.

Student's Signature ……………………….. Date ……………………...

Approval of the Master Thesis

This Master Thesis is approved by the following Examination Committee Members:

1) Advisor (Committee Chair): Dr. Qurban Ali Memon

Title: Associate Professor

Department of: Electrical Engineering

College of: Engineering, UAE University

Signature:________________________________Date:__________________

2) Member: Dr. Atef Abdrabou

Title: Assistant Professor

Department of: Electrical Engineering

College of: Engineering, UAE University

Signature:________________________________Date:__________________

3) Member (External Examiner): Dr. Yakoob Siyal

Title: Associate Professor

Department of: Electrical & Electronic Engineering

Institution: Nanyang Technological University (Singapore)

Signature:________________________________Date:__________________

This Mater Thesis is accepted by:

Dean of the College of Engineering: Prof. Mohsen Sherif

Signature:___________________________________Date:_________________

Dean of the College of Graduate Studies: Prof. Nagi Wakim

Signature:___________________________________Date:_________________

Abstract

Obesity is a rising health concern in the United Arab Emirates. Obesity leads to health risks such as diabetes, heart disease, and other conditions. It may also lead to eating disorders, depression, and low self-esteem. Regular exercising helps to protect from obesity, diabetes, heart disease, high blood pressure and better manage stress. Despite all the benefits of exercising, people are doing less of it.

In this research, an effort is made to encourage people to exercise by extending the prevalent sensing abilities of Online Social Networks (OSNs). Sensors are used to collect and generate data to be processed, clustered and shared via a locally developed application named ‘Care’. Obviously, different issues (such as privacy and personal-data access) have also been addressed.

The purpose of this research is to improve social networking in the area of healthcare by extending the integration of Wireless Sensor Networks (WSNs) with Online Social Networks (OSNs). The integrated system tracks and locates friends and family members, encourages exercise, and enables doctors to view specific information such as heart rate.

The solution is developed for an Android operating system. It provides the user with the set of services: (i) showing specific places (ii) sharing the user location; (iii) showing nearby friends; (iv) clustering nearby friends; (v) calculating and sharing distance moved, calories burned and active time; (vi) getting and sharing weather temperature; and (vii) calculating, tracking and sharing the user heart rate. A data privacy model is developed using data levels and user roles to ensure privacy.

Keywords: social networks, privacy, role based access, Healthcare Network

توسعة استشعار شبكات التواصل الاجتماعي عبر الانترنت من منظور الرعاية الصحية

الملخص

أصبحت البدانة مشكلة صحية مثيرة للقلق في دولة الإمارات العربية المتحدة، وذلك لما تسببه البدانة من أضرار صحية كمرض السكري، و أمراض القلب و غيرها من الحالات المرضية. كما تؤدي البدانة أيضا إلى اضطرابات في الأكل، و حالات من الاكتئاب، و أضرار أخرى نفسية كقلة احترام الذات. لذلك، فإن ممارسة الرياضة بانتظام تقي من البدانة، وأمراض السكر، وأمراض القلب، و ارتفاع ضغط الدم، وتساعد في السيطرة على حالات التوتر. وبالرغم من كل فوائد ممارسة الرياضة، فإن الكثيرين لا يقومون بها.

يتناول هذا البحث خطوة لتشجيع الناس على ممارسة الرياضة عن طريق توظيف قدرات الاستشعار المنتشرة في شبكات التواصل الاجتماعي عبر الانترنت(OSNs). ويتم ذلك عن طريق استخدام أجهزة الاستشعار لجمع و توليد البيانات و معالجتها ثم تجميعها قبل إمكانية مشاركتها مع الآخرين باستخدام برنامج تم تطويره محليا يدعى (Care). ولخصوصية البيانات، فقد تم تناول قضايا الخصوصية و الوصول الى البينات الشخصية و أخذها بعين الاعتبار.

يهدف هذا البحث لتطوير شبكات التواصل الاجتماعي في مجال العناية الصحية من خلال الوصول الى التكامل بين كل من شبكات الاستشعار عن بعد (WSNs) و شبكات التواصل الاجتماعي عبر الانترنت (OSNs). حيث يقوم النظام المتكامل بتتبع الأصدقاء وأفراد العائلة مشجعا إياهم على ممارسة الرياضة، ويُمَكِّن الأطباء من التعرف على معلومات معينة كمعدل نبضات القلب لمتابعة مرضاهم.

لقد تم تطوير البرنامج (Care) ليتناسب مع نظام التشغيل (أندرويد-Android) والذي يقدم عدة خدمات للمستخدم مثل (1)إظهار أماكن تواجد المستخدم (2) مشاركة مكان تواجد المستخدم مع الآخرين(3) إظهار الأصدقاء المجاورين لمحيط المستخدم (4) تصنيف الأصدقاء المجاورين لمحيط المستخدم في مجموعات (5) حساب المسافات التي يقطعها المستخدم إضافة إلى السعرات الحرارية التي تم حرقها والوقت الفعلي للقيام بذلك ومشاركتها مع الأصدقاء (6) الحصول على درجة حرارة الجو ومشاركتها مع الأصدقاء (7) حساب ومتابعة معدل نبضات القلب ومشاركتها مع الأصدقاء. ويجدر بالذكر أنه قد تم تطوير نموذج للمحافظة على خصوصية البيانات باستخدم مستويات البيانات و قوانين المستخدم لضمان الخصوصية.

Acknowledgments

First of all, I would like to thank Allah for giving me the faith, luck and strength to successfully complete my work. My deepest thanks go to my family and fiancé who have supported me, and given me strength and encouragement to complete this work. I would like to express my deepest thanks to everyone who helped me during this unforgettable period of my life. Firstly, I would like to thank and show my deep appreciation to my thesis supervisor Dr. Qurban Memon for his unlimited support, and guidance. He was always approachable, punctual and provided me with precious knowledge and advice during this entire project. My acknowledgments are extended to all faculty members of the Department of Electrical Engineering at the United Arab Emirates University for their continuous support and encouragement.

Dedication

To my beloved family, fiancé and friends

Table of Contents

List of Figures

List of Abbreviations

WSNs: Wireless Sensor Networks

OSNs: Online Social Networks

ADCs: Analog Digital Converters

MEMS: Micro Electro Mechanical Systems

GPS: Global Positioning System

APIs: Application Programming Interfaces

NGOs: Non-Governmental Organizations

FOAF: Friend of a Friend

SIOC: Semantically-Interlinked Online Communities

RDBMS: Relational Database Management System

RDF: Resource Description Framework

WoT: Web of Things

BSP: Business System Planning

WWW: World Wide Web

OS: Operating System

GSM: Global System for Mobile communications

OBD-II: Updated On-Board Diagnostics standard effective in cars sold in the US after 1-1-96

RPM: Revolutions Per Minute

CO2: Carbon Dioxide

SD Cards: Secure Digital Cards

OSM: Open Street Map

XML: Extensible Markup Language

GUI: Graphical User Interface

NP-hard: Non-deterministic Polynomial-time hard

RBAC: Role-Based Access Control

IRT: Item Response Theory

IC: Information Criterion

AIC: Akaike Information Criterion

BIC: Bayesian Information Criterion

PQ: Privacy Quotient

HTTP: Hypertext Transfer Protocol

FQL: Facebook Query Language

SQL: Structured Query Language

URL: Uniform Resource Locator

SDK: Software Development Kit

IDE: Integrated Development Environment

ADT: Android Development Tools

Chapter 1

1.1 Introduction

This chapter addresses the distinctive features of Wireless Sensor Networks (WSNs) and Online Social Networks (OSNs), the interconnection of both networks and the privacy concerns. First, it introduces WSNs, its definition and components, sensor types, desirable functions for sensor nodes, and WSNs benefits and applications. Then, it examines social networks, their growth, usage and the services provided by them. After that, it discusses the integration of OSNs with WSNs, the available tools, technologies, platforms, integration examples, issues, drivers, and challenges. It also presents clustering, the factors for forming social network communities and the benefits of organizing users into groups or clusters. Finally, it summarizes the privacy concerns, the various threats that OSNs are facing and the protection mechanisms used to protect user data.

1.2 Sensor Networks

WSNs consist of large numbers of small lightweight wireless nodes which have sensing, computation and communication capabilities to monitor the environment or system, by measuring physical parameters such as temperature, pressure, humidity, sound etc. In spite of all these capabilities sensor nodes still have limited processing capacity and power. Advances in wireless communications have made it possible to develop WSNs made of small devices which collect information by cooperating with each other.

WSN consists of four components: (i) Sensing unit: is made up of sensor and Analog Digital Converters (ADCs). A sensor transfers the physical phenomenon to electrical signals and an ADC converts the analog signals produced by sensor to digital signals; (ii) Processing unit: is composed of a microprocessor or microcontroller. The microcontroller controls sensors, signal processing algorithms on the collected sensor data and the execution of communication protocols; (iii) Transmission unit: receives the information from the processing unit, and then transmits it to the outside world; and (iv) Power unit: battery power is the main source of power in WSN [1].

Sensors are small sensing devices called nodes and made up of a transducer an embedded processor (for data processing), small memory (for data storage), battery (for energy) and transceiver (for receiving and sending signals or data from one node to another). In a sensor network, each sensor node communicates wirelessly with a few other local nodes within its radio communication range. Some sensor nodes are Micro Electro Mechanical Systems (MEMS), which react to change in some physical condition for example, temperature and pressure by producing a measurable response. As wireless sensor nodes are typically very small electronic devices, they can only be equipped with a limited power source. Each sensor node has a certain area of coverage for which it can reliably and accurately report the particular area that it is observing.

There are three types of sensor nodes: (i) Passive, Omni Directional Sensors: passive sensor nodes sense the environment without manipulating it by active probing; (ii) passive, narrow-beam sensors: these sensors are passive and they are concerned about the direction when sensing the environment; and (iii) Active Sensors: these sensors actively probe the environment [2].

Sensors provide information about various aspects of the real world. Advances in the research of wireless sensor networks in the last decade have led to their deployment in application domains. Sensors have become more prevalent in mobile devices in recent years. Nowadays, mobile devices have Global Positioning System (GPS) and accelerometer sensors coupled with Bluetooth and Wi-Fi communication stacks. A wide range of Bluetooth sensors, such as heart monitors, and environmental monitors can now be found on these mobile phones enabling a new paradigm [3-4].

Desirable functions for sensor nodes include: ease of installation, self-identification, self-diagnosis, reliability, time awareness for coordination with other nodes, some software functions and DSP, and standard control protocols and network interfaces, such as IEEE 1451, standard for smart sensor networks. The objective of this standard is to make it easier for different manufacturers to develop smart sensors and to interface them to networks [5]. A simple view of interfacing transducers or sensors to real world networks based on IEEE 1451 is shown in Figure 1: A smart sensor network.

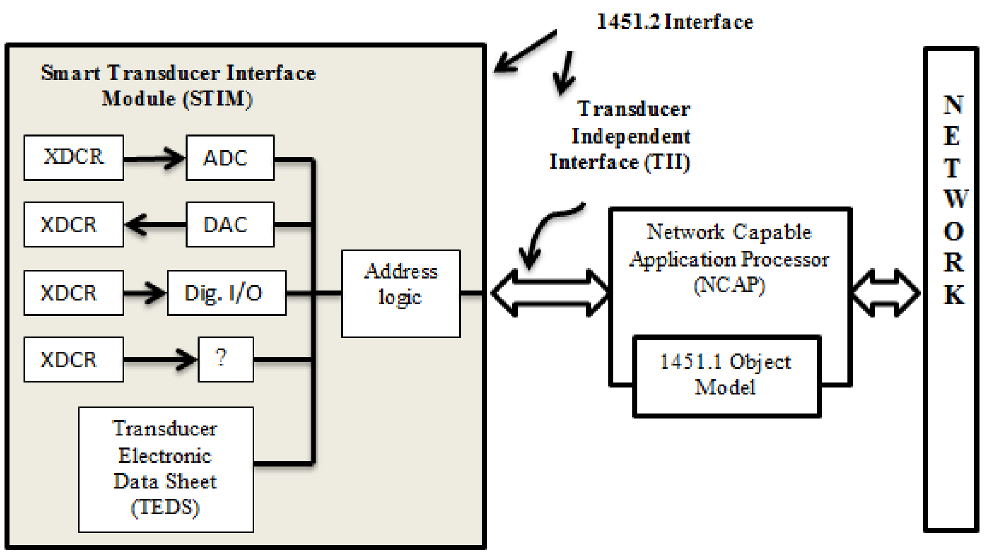

1451.2 Interface
Smart Transducer Interface Module (STIM)
XDCR
ADC
XDCR
DAC
XDCR
Dig. I/O
XDCR
?
Transducer Electronic Data Sheet (TEDS)
Address logic
Transducer Independent Interface (TII)
Network Capable Application Processor (NCAP)
1451.1 Object Model
NETWORK

Figure 1: A smart sensor network

1.2.1 Benefits

In recent years, WSN applications have increased rapidly due to WSNs advantages. The advantages of WSNs can be summarized as follows: (i) Network setups can be done without fixed infrastructure; (ii) WSNs are the best for non-reachable places such as the sea, rural areas, deep forests or mountains; (iii) flexible if there is ad hoc situation when an additional workstation is required; (iv) implementation cost is cheap [2]. WSN is useful in all the areas nowadays.

In comparison to traditional networks, wireless sensor networks present better functionalities to monitor larger scaled and changing topology with limited power and computational abilities in dense deployments. WSNs have been deployed in application areas for environment, military, health, traffic control and other areas of interest for gathering data. Examples of wireless sensor network include applications for military surveillance, habitat monitoring, structural monitoring, cargo tracking, battle field, disaster recovery, environmental monitoring, automobile industries, construction, healthcare, and many other applications. The size of each sensor node differs with applications, and its cost depends on parameters such as memory size, processing speed and battery.

1.3 Social Networks

Recent years have witnessed the dramatic popularity of OSNs (e.g., Facebook, Google+, Twitter, MySpace, and Friendster). The number of users participating in these networks is large and still growing. An OSN can be considered as a user-generated content system that enables its users to share information and communicate with others. An OSN can be represented as a graph, where nodes are individual users and edges are relationships between them. These relationships can be either directed or not. An OSN enables its users to (i) create a profile page; (ii) publish information; and (iii) explicitly connect to other users. A user’s profile contains entries for name, age, education, work or home town. OSNs provide a variety of features only for users who are logged in. Users can update their profiles, browse other users’ profiles and search for their friends via categories such as education or work history. They can invite new friends, adjust their privacy settings, and join groups of people or friends from the physical world, or from the virtual one. Those relationships might be people to people peering, people to stuff peering, stuff to stuff peering, and many other possibilities. Friends can be generated from common values and interests, from virtual gaming environments, or from the same purchasing behaviors for specific products.

OSN platforms provide authorization services to control access to user information from third party web sites. Most systems use the OAuth standard to allow mobile and desktop applications to access OSNs as well as third party sites. OSNs support Application Programming Interfaces (APIs) for third party applications to improve the system by accessing and manipulating social network data. An important API supported by all OSNs is to be able to share or receive updates to the user's activity stream.

OSNs impact almost every aspect of networking, including network architecture and system design, privacy and user behavior, and measurements and performance modeling/analysis. Real-world OSNs are highly dynamic structures. These dynamics result from new users joining the system by creating a new account, existing users leaving the system, and active users interacting with each other. Members of OSN are classified into three groups. The first group is Singletons (i.e., Loners who have never made a connection with another user and don't participate actively in the network). The second one is Giant component (i.e., A large group of highly active people who are connected to one another) and the last one is Middle region (i.e., The remainder which consists of small groups and isolated communities) [6]. To better understand how OSNs can be integrated with the physical world we need to understand the services provided by current social network platforms which are (i) Identity and authorization services, (ii) Application Programming Interfaces (APIs) to access and manipulate the social network graph and publish and receive updates and (iii) Container facilities for hosting third party applications [7].

1.3.1 Benefits

People use social networks to achieve a variety of social needs including: self-expression and self-presentation, to connect and share information, experiences and ideas with others via the Internet, and for building friendships and maintaining family relationships by allowing people to easily share personal thoughts, pictures, and accomplishments. Social network use can enhance how people feel about themselves and their well-being. OSNs also hold a promise for building business and professional relationships.

Social Networking services have become extremely popular in recent years. These services include: (i) advertising: many universities and organizations now advertise by sending invitations on Facebook instead of distributing paper fliers. This type of advertising is quicker and cheaper; (ii) invitees on Facebook can easily check which of their friends are attending a particular event, and accept or decline the invitation accordingly; (iii) promoting events, offering tutorials and training, facilitate discussions, and posting useful information; and (iv) creating profiles containing information about users (e.g., pictures, interests, and personal information) and connecting to others to expand their personal networks.

Recently online social networking has become a popular work tool among NGOs (nonprofits, charities, civil society organizations). These organizations are using social networks in order to promote awareness, recruit followers, and raise money. Some of the advantages of setting up online social network are: (i) OSNs are great for target marketing to find employees, volunteers and supporters from different backgrounds and groups; (ii) Professional networking sites, such as LinkedIn, enable

NGOs to build professional credibility; (iii) Social networks enable similar NGOs to connect and work together; and (iv) Blogging websites, such as Twitter, can tell people what the day-to-day operations of NGOs are as well as what major events will be taking place [8].

1.4 Integration of Online Social Networks with Wireless Sensor Networks

WSNs have become mature enough for widespread adoption. However, most of the related use cases are highly application-specific, and do not usually affect everyday life. As a consequence, WSNs are not considered as valuable by most people. Indeed, WSNs could be made more appealing to end users by leveraging online social networks (OSNs). This includes developing novel application scenarios and interaction paradigms between WSNs and OSNs.

1.4.1 Available Tools, Technologies, and Platforms

Semantic Web and lightweight models such as Friend of a Friend (FOAF) and Semantically-Interlinked Online Communities (SIOC) are used to integrate data from different networks, or to find all related content about a particular topic. They provide a unified layer on the top of existing applications. This means that a social semantic web is used to extract data from different places (i.e., Social Networks, Social data, Relational Database Management System (RDBMS) and legacy data) to use them in a new application. The pictorial view of such interoperability is shown in Figure 2.

At present, social networks and sensors information may be modeled using semantic web technologies. The flexibility of Resource Description Framework (RDF) allows data to be linked together from various sources, while the use of shared ontology offers common semantics for this data [3-9].

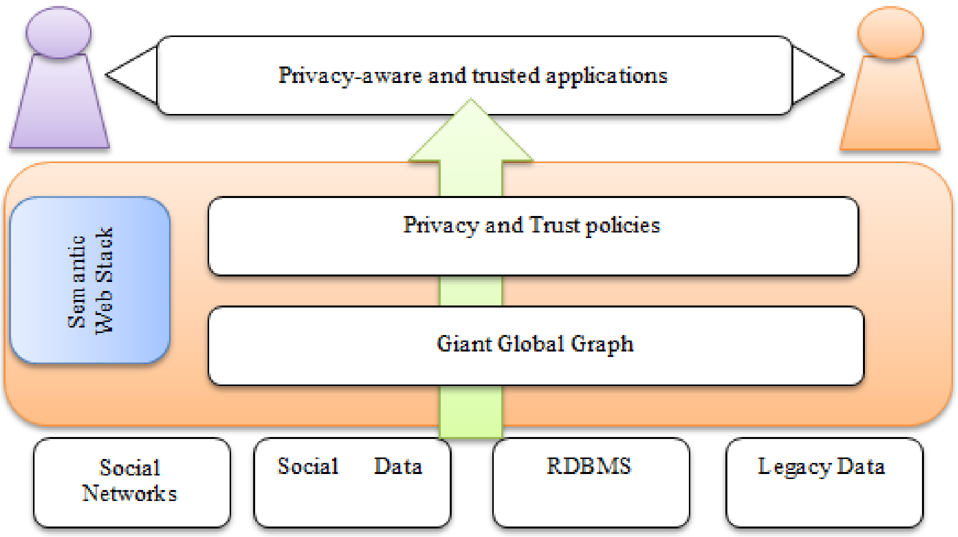


Figure 2: Interaction between social networks and sensor networks [3]

A number of recent technological advances in hardware and software have initiated the integration of sensors and social networks. These technological advances are the development of fast stream processing platforms, the development of stream synopsis algorithms and software, increased bandwidth and increased storage [4].

There are several popular OSN platforms. Facebook is the most popular OSN platform today. Facebook supports OAuth 2.0 for authentication and authorization of third party web sites and applications. OAuth is an open standard to authorization, which provides client applications a safe access to server resources on behalf of a resource owner. It enables resource owners to authorize third party access to their server without sharing their credentials. OAuth 2.0 is the next evolution of the OAuth protocol, which is not compatible with OAuth 1.0. OAuth 2.0 concentrates on client developer simplicity while supporting specific authorization flows for desktop applications, web applications, mobile phones, and living room devices. Facebook's Graph API and Google APIs only support OAuth 2.0 [10].

Facebook supports two APIs for developers: a legacy RESTful API as well as the new Graph API. Facebook's Graph API now supports notifications to support detecting some changes in the graph. Several types of applications are supported by Facebook: desktop and mobile applications, and third party web sites or canvas applications which are contained in the Facebook UI. Another OSN platform is Twitter. The purpose of twitter is to connect users who send short messages called tweets to followers. Twitter uses OAuth for third party applications. Twitter has three APIs - two REST APIs and a long polling API. One is for interacting with the system as a user, and the other for searching for tweets. A third API provides streaming services [7].

1.4.2 Integration Examples, Issues, and Drivers

Some examples of integration of social and sensor networks are as follows: The Google Latitude application collects mobile position data of users, and shares this data among different users. The city sense application collects sensor data extracted from fixed sensors, GPS-enabled cell phones and cabs in order to determine where the people are, and then carries this information to clients who subscribe to this information. MacroSens analyzes customer location behaviors in order to determine individuals which behave in a similar way to a given target. A number of real-time automotive tracking applications such as ‘Automotive Tracking Application’ determine the important points of congestion in the city by pooling GPS data from the vehicles in the city. Animal Tracking uses tracking data collected with the use of radio-frequency identifiers [4-7]. One of the examples of integrating a social network into the future Internet is ‘Social Sign On’, which enables website visitors to authenticate to the company’s website using their existing credentials such as Facebook, Twitter, OpenID etc. The benefits for ‘Social Sign On’ are: increasing the engagement rate by noticing which friends visited the same website and interacting with them; providing more accurate contextual information and reducing time in marketing [11].

Sharing sensor data creates new venues for loss of privacy, resulting in new privacy attacks that exploit physical-side channels or a-priori information about the physical environment. Research is needed on both privacy specification and enforcement to put such specification and enforcement on solid analytic foundations, much like specification and enforcement of safety requirements of high-confidence software. Specification calls for new physical privacy specification interface that is easy to understand and use for the non-expert. Enforcement calls for two complementary types of privacy mechanisms: (i) a protection mechanism from involuntary physical exposure, and (ii) control of voluntary information sharing [4].

There are a couple of important drivers for integrating sensor and social networks. One driver for integrating sensors and social networks is to allow the actors in the social network to both publish their data and subscribe to each other's data either directly or indirectly after discovery of useful information from such data. The idea is that such collaborative sharing on a social network can increase real-time awareness of different users about each other. A second driver for integrating sensors and social networks is to better understand or measure the aggregate behavior of self-selected communities or the external environment in which these communities function. Examples may include understanding traffic conditions in a city, understanding environmental pollution levels, or measuring obesity trends [4].

1.4.3 Challenges

Social sensors provide numerous research challenges from the perspective of analysis. Since the collected data typically contains sensitive personal data (e.g., location data), it is extremely important to use privacy-sensitive techniques to perform the analysis. A recent technique called PoolView designs privacy-sensitive techniques for collecting and using mobile sensor data. Another challenge is that the volume of data collected can be very large. For example, in a mobile application, one may track the location information of millions of users simultaneously. Therefore, it is useful to be able to design techniques which can compress and efficiently process the large amounts of collected data. A further challenge is the dynamics and real timeliness. For example, applications that trigger alerts are typically time-sensitive and may require real-time response [4-12].

To achieve stronger integration and continued convergence of the real world into OSNs there are three challenges to overcome. The first one is to support two-way interaction, where it may be required to make sure that the data integrity and timeliness is maintained. The second one is to extend OSN APIs and Models for the Web of Things (WoT). This means that the social network service and data models support not only users and social relationships, but also communities, content and web pages, physical places and things. The third one is real world user interface issues, i.e., how to present social networks information to users while creating a bridge between OSN graphs and situated sensors, actuators and displays [7].

1.5 Clustering

Social networks are graph structures whose nodes or vertices represent people or other entities embedded in a social context, and whose edges represent interaction or collaboration between those entities. Social networks are highly dynamic, evolving relationships among people or other entities. This dynamic property of social networks makes studying these graphs a challenging task [13].

A fundamental problem related to these networks is the discovery of clusters or communities. A cluster is a collection of individuals with dense friendship patterns internally and sparse friendships externally [14]. The clustering in social networks is different from traditional clustering. It requires grouping objects into classes based on their links as well as their attributes. While traditional clustering algorithms group objects only based on objects' similarity, which can't be applied to social networks [13]. The biggest challenge of social network clustering is how to divide objects into classes based on objects' links. One of the proposed algorithms to meet this challenge is Business System Planning (BSP) [15]. The authors in [13] have proposed an algorithm of social network clustering analysis based on the BSP clustering algorithm [15]. It divides a social network into different classes according to objects in the social network and links between objects, and it can also identify relations among clusters. The main disadvantage of the BSP algorithm is that it uses matrices to store edges and reachable relations. A reachable relation is the directed link between two objects. A reachable relation can be through one and only one directed edge or through two or more directed edges. In a real social network, these matrices are so huge, that they can't be loaded into main memory. But these matrices are very sparse, so an efficient data structure is designed to overcome this shortcoming.

1.5.1 Factors for formation of social network communities

The factors that affect the formation of social network communities are still not well understood. Different theories have been put forward to explain why social groups arise. The common identity theory states that individuals gather into groups when they share a common interest or purpose, such as the fans of a sports team. The common bond theory proposes that communities are held together by the social ties between their members, as in families or in groups of close friends. The factors that keep communities alive are often not evident in the network structure, so additional information about their members must be exploited to understand why communities form. The results in [16] demonstrate that online communities can arise for different reasons. For some social networks, geography is essential to characterize communities fully. Furthermore, since places can play a vital role for communities, social theories such as the common bond and common identity theories could be used in conjunction with location information to capture how communities form in real systems, and give more accurate models of community evolution.

1.5.2 Benefits

The tendency to organize into groups or clusters is a fundamental property of human nature. Clusters in social networks are associated with several benefits, for instance, target marketing schemes can be designed based on clusters, and it has been claimed that terrorist cells can be identified [14]. Also, individuals rely on groups to achieve ends they could not achieve alone, and as a means for defining their personal identity. It is widely accepted that social ties between individuals are critical to the success of social movements in recruiting new members [17]. Furthermore, learning in groups creates motivation for a user by offering social interaction. The author in [18] focuses on how to form a group for collaborative learning in an eLearning-enabled OSN.

1.6 Privacy

Privacy and security are serious topics in a variety of areas of computer science. The area of online social networks is of special interest because of the critical data involved. Users generate massive amounts of data in online social networks. Users share their personal information, opinions, activities, videos and photos on online social networks. This collection of sensitive, personal, identifiable data form online social networks has never been before in a single place like now. The gathering of this data is highly dangerous. For example, birthdate and hometown are all that is needed to define one's social security number and these are usually available on a user's profile in an online social network. The concern keeps increasing about the security of users data entrusted to online social networks and how the privacy of this data is managed. There are various weaknesses from which an adversary can access user data in online social networks so users should be careful about who has access to their published content. Practical levels of access are divided into four categories: other users, third party advertising agencies, third party online social network applications and the online social network provider.

Privacy is the selective exposure of information about oneself. In recent years, it has become increasingly hard for people to control what they are exposing to whom.

The innovations in World Wide Web (WWW) [19] and the recent trends in data protection [20- 21] have increased the attraction of using online social networks. However, these advancements in technology and tools are also complimented by corresponding privacy concerns. The important thing to note is the lack of awareness of the potential risks involved when data is being shared online [22]. Specifically, external entities can mine this data and use it for different purposes like spamming [23], discovering interaction patterns in the enterprise to offer and develop innovative services, identifying important people in the network, detecting hidden clusters, identifying user sentiments for proactive strategies and so forth [24].

1.6.1 Threats

Online social networks are facing various threats. Other users are a constant threat because online social networks make it easy for anybody to create an account and be able to access other users' data. Other users on the social network are divided into three categories: (i) Directly Connected Users: users who have a link between them; (ii) Unconnected or Indirectly Connected Users: users who are two or more hops away from each other and (iii) General Public: Everyone who has access to some information in online social networks. Online social networks make it possible for third party developers to write codes that have access to a set of API functions which allow the developed application to access content available to the online social network provider. Third party applications such as games and productivity, have become very popular and penetrative. Users are required to grant these applications access to their data in order to use them. Privacy risks are increasing by disclosing user data to third party developers. Because online social networks have huge amount of users' data, marketing can be targeted in a way never before thought possible. Advertisements are now targeting the users' individual interests. Advertising agencies can determine the specific interests of users based on their own generated content. Online social network providers get the majority of their revenue from advertising agencies that is why online social network providers have built their business model on providing highly focused advertisement space to advertisers. It is risky providing such complete profiles to third party corporations that have little interest in keeping users' data safe. A social network provider can see all data that flows through the network and it has access to all the data on the online social network, so it is considered as a threat. Nowadays, users share their location information with friends and applications. This raises privacy issues since location information is very sensitive [25].

1.6.2 Protection

Because of all the previous threats, many protection mechanisms have been developed to protect user data. Online social network providers enable the users to specify their privacy settings. Privacy settings are the first line of defense against malicious users. These privacy settings are straightforward and should be set carefully by users to protect their own data. For example, protect users’ data from advertisements it is better to block malicious or adversarial advertisements. Moreover, open standards and third party software development have made a partnership that provides users with the required tools to manage their privacy [25].

1.7 Thesis overview

The purpose of this thesis is to extend the integration of WSNs and OSNs, to improve social networking in area of healthcare to help people find motivation to exercise, for doctors to see specific information, to track and locate friends and family members to share certain parameters. Chapter two presents the literature review including integration examples, clustering and privacy concerns. In chapter three, the system architecture and design are discussed. Then, in chapter four, results and discussion are highlighted. In the last chapter, the conclusion presents an overview of the thesis, the main contributions and features, and future research directions.

Chapter 2

2.1 Literature Review

This chapter examines the integration examples of sensor networks and online social networks. First, it presents integration applications, their functions, security and privacy issues, risks, platforms, tools, protocols, sensors, how these applications attract users, and discusses whether they can be used easily. Then, it discusses clustering, related algorithms, what is a good cluster, clustering problems, and group stability. Finally, it analyses privacy and security challenges, ways to overcome them, how to measure users' privacy, and related leaks.

2.2 Integration of Sensor Network and Online Social Network

Nowadays sensors and social networks can fruitfully interface, from sensors providing contextual information in context-aware and personalized social applications, to using social networks as "storage infrastructures" for sensor information. The integration of sensor networks with social networks leads to applications that can sense the context of a user in much better ways and thus provides more personalized and detailed solutions. Obviously, different issues (such as privacy and personal-data access) are to be taken into account, every time specific integration is to be devised. This is because generic interfaces and protocols are still being researched. This area is still open to research as new drivers and requirements surface.

2.3 Integration Examples

Recently, a tremendous amount of research has been done on integrating sensors and social networks. The related works proposed a number of sensor applications to collect data that can be associated with human interactions.

2.3.1 Google Latitude Application

Google Latitude is a location-aware mobile app developed by Google as a successor to its earlier SMS-based service Dodgeball. Latitude lets a mobile phone user permit certain people to view their current location. The user's cell phone location is mapped on Google Maps, through their own Google Account [26]. The Google Latitude application shares the collected mobile position data of user among different users. The sharing of this data usually leads to significant events of interest. For example, proximity alerts may be triggered when two linked users are within geographical proximity of one another. This may lead to changes in user-behavior patterns, and therefore underlying sensor values. Usually the data on one sensor affects data in other sensors.

The Latitude application enables the creation of virtual friends, who are other users who use devices such as personal computers which can transmit approximate location data such as IP-addresses or other users who carry the same location-enabled device. A number of other applications enabled by the Google Latitude main application are:

- **Location Alerts**

This application enables the triggering of alerts when two latitude friends are near to each other. The alerts are triggered on the basis of both time and location when something interesting is happening. For example, an alert will be triggered when two friends are doing something at a scheduled time, but they are in an unusual place. Similarly, when two friends are doing something in a regular place, but at an unscheduled time.

- **Public Location Badge**

The user can post his/her location directly on blog or social network. This will increase the visibility of his/her information to other users of the site.

- **Use with Chat Applications**

Users who are chatting to each other can see the location of one another with the use of the embedded latitude functionality. This means that the mobile location can be used in conjunction with the Google Talk application which enables users to chat to one another [4].

2.3.1.1 Google Latitude security and privacy

The user can control the detail and the accuracy of what other users will be able to access and see. For example, the user can allow others to see the exact location or limit it to identifying the city only. For privacy, the user can turn off this application, or enter his location manually. Also, users have to opt in to Latitude, and so they can only see the location of the friends who want to share their location with them [26].

To ensure the users' privacy and security, Google latitude gives users three different options to share their location: they can allow Latitude to detect their location to the best of its ability and automatically share it; they can enter their location manually by entering their address or their city; or they can hide their location entirely. Users can select the option they prefer in the Privacy menu. In addition, they can change their options for each friend individually [27]. Google has built this application taking into consideration security and privacy issues. The service only stores for example, the last known location of a given user [30].

2.3.1.2 Google Latitude Risks

This application will increase privacy risks if it is used in a wrong way. It might be used by businesses to easily track the movements of their field force, and will have huge implications when integrated with social networks and blogging services. It is claimed by Google that they do not store any location information, but handle the data for sharing purposes. However, It is not easy to use this application correctly. Google Latitude app can provide great benefits, but misusing it can be damaging. Even if Google have built a ‘privacy-positive’ system, there are privacy risks which include:

1- Latitude installed on mobile devices without users’ knowledge - for example, a jealous partner installing it on partner’s phone could track a partner’s location.
2- Hacks could install Latitude on a handset without the need for physical verification, or suppress warning messages that the service is running.
3- Users could forget that their mobile phone is transmitting their location when they go somewhere that they wish to keep to themselves.
4- Unauthorized access to back-end services where an unauthorized user can track individuals’ locations.

The biggest concern is the inevitability of irresponsible sharing of location data by users who don’t really know other parties involved. It would only be a matter of time before frauds, burglaries and physical assaults emerged inspired by a user’s location. As we can see, Latitude is not a bad app, but we need to think very carefully about the implications of its location-based services. There should be legal protection for users, and law enforcement authorities need to have the understanding and resources to enforce protections. And the users should understand the implications of the services [29]. There is absolutely no doubt that this service is extremely beneficial, but it will be destroyed by privacy if the companies don’t get it right [30].

2.3.1.3 Google Latitude Platforms

Google Latitude is compatible with most devices running iOS, Android, BlackBerry OS, Windows Mobile and Symbian S60. On most platforms Latitude can continue to update the user’s location in the background when the application is in use. The Sony Ericsson W995, C905, C903, C510, Elm and Satio mobile phones support Google Latitude as part of its built-in Google Maps application. Although this is a Java ME application, it cannot be downloaded for use with other mobile phones [26].

2.3.1.4 Google Latitude Tools

The "opt-in" Latitude service uses data from GPS, Wi-Fi hardware, or mobile phone masts to update a user's location automatically [30]. With Google Latitude, the service has expanded to PC browsers (as it uses the Geolocation API as well as user-driven input) and automated location detection on mobile phones using cellular positioning, Wi-Fi positioning and GPS [26].

2.3.1.5 How Google Latitude attract users

It is clear that Latitude functions and techniques change the nature and dynamics of social interactions between users. For example, the triggering of alerts can lead to a different pattern of interaction among different users. The ability to mine the dynamics of such interactions is a useful challenging task for a variety of applications [4].

2.3.1.6 Ease of use

Google has always been known for offering simple to understand Google Maps. Google Latitude is no exception to that rule. The app is easy to use since the directions on screen of the app are self-explanatory. Many other cell phone tracking apps are very slow when location data is being shared and received. However, Google Latitude is not affected by this issue and it works well even when a large amount of location data is being exchanged [31].

2.3.1.7 Google Latitude Protocol

The Google Latitude API uses the RESTful calling style. The Google Latitude API allows for websites and programs to integrate with Google Latitude, authorizes users to read and update their current location and their location history.

2.3.1.8 Google Latitude Sensors

The Google Latitude Application uses GPS data collected from Google map users on mobile cell phones. It is also possible to collect more data using cell phones tower location data or the IP addresses of a computer which is logged into the personalized Google page called iGoogle. Latitude provides a mobile Google Maps service, but with a difference: the mobile uses GPS and GSM cell information to locate the user and position them on the map. Users can then choose to share the location or not [29].

2.3.1.9 Other Tracking Applications

Google Latitude is perhaps the most well-known tracking application. A number of tracking applications were designed for mobile devices tracking using GPS. These applications were designed purely for the purpose of tracking something which might be lost. Other apps might involve more complex social interactions.

Any software and hardware combination which enables tracking has the potential to be used for social sensing applications. Some examples of these applications are:

- **Navizon Application**

This application uses GPS to allow social interaction among people with mobile phones. Navizon app allows tracking of mobile friends, coverage of specific areas and trails followed by a particular user.

- **iLocalis Application**

This application is designed for particular mobile platforms such as the iPhone and it allows the tracking of family and friends. Using the web, it is also used by corporate applications to track employees' mobile. When the friendship links have been established, the application will enable the user to send messages to his/her friends, when they are nearby [4].

2.3.2 The CitySense and MacroSense Applications

The CitySense and MacroSense applications collect real-time data from different GPS-enabled cell phones, GPS-enabled cabs and cell phone tower triangulation. These two applications have similarities in this underlying methodology, but have different functionalities targeted to different kinds of audiences.

- **CitySense Application**

The CitySense application has a social networking version of a collaborative filtering application. This application uses the personal history for each user to determine where other similar users might be. It also provides recommendations to users about the possible places they would like to visit according to their stored history.

- **MacroSense Application**

The MacroSense application is similar to the CitySense application in terms of the data it collects and the functionality it provides. This application is focused towards the commercial segment in predicting consumer behavior. The application can predict what a particular customer may like based on his/her personal history which includes location and behavior. The main idea behind this application is to segment customers into marketing groups based on their stored history and behavior, and then this information is used in order to make predictions. For example, the popularity of a product with users who are most like the target can be used for predictive purposes. This approach is similar to collaborative filtering, except that it uses customer behavior instead of their feedback. The effectiveness of particular behaviors which predict the interests are also used. This analysis can be performed in real time, which provides great value in terms of predictive interactions. The analytics can also be used in order to predict group influences of the behaviors of the underlying subjects [4].

2.3.2.1 The CitySense and MacroSense security and privacy

Sense Networks (CitySense and MacroSense) have ways to function in a privacy-friendly manner. The location data can be used without personal information which will bring users a more relevant mobile browsing experience. Also, the sensitive details that might make people uncomfortable will be discarded [32].

Sense Networks has built its systems from scratch to support a paradigm of data ownership and privacy. Sense Networks respects the privacy and anonymity of its users and captures no personally identifiable information. Sense Networks uses best practices to ensure the safe keeping of the data it receives and deletes all raw data.

Sense Networks never shares specific user data with anyone. Access to data is strictly controlled [33].

2.3.2.2 The CitySense and MacroSense Risks

Sense Networks turns raw data into anonymous behavioral information, which can be used to increase the relevance of mobile offers and advertisements that users receive on their mobile phones. Sense Networks is committed to protecting consumer privacy by informing the consumer what information will be collected, how it is going to be used and with whom it might be shared. These applications might have risks if they are misused. Sense Networks takes reasonable measure to protect the information collected and stored. All data is stored in a secure facility and access to the data is strictly controlled [34].

2.3.2.3 The CitySense and MacroSense Platform

Sense Networks has a platform, called Macrosense (location analytics platform) that receives streaming location data in real-time, processes and analyzes the data in the context of billions of historical data points and stores it in a way that can be easily queried to better understand aggregate human activity. Citysense and Macrosense applications are built on top of the Macrosense platform. These applications were built for iPhone and Blackberry platforms [35].

2.3.2.4 The CitySense and MacroSense Tools

These applications currently access cell-phone and taxi GPS data from about four million GPS sensors, to find the local hot spots. They then link to Google and Yelp to display what places are operating as popular locations. The product is currently available in San Francisco only, but a New York version will be launched soon [35].

2.3.2.5 How CitySense and MacroSense applications attract users

These applications turn massive amounts of mobile location data into actionable, predictive behavioral data. They analyze and process data in real-time and their robust algorithms continuously learn as new data arrives. MacroSense's real-time models depend on state-of-the-art machine learning techniques and are supporting ad selection decisions that occur in milliseconds [32]. These applications eliminate the need for searching. CitySense improved searching to sensing. It passively "senses" the most popular locations based on actual real-time activity and shows a live heat map. The application intelligently provides the inherent wisdom of crowds without changing the existing user behavior, in order to guide people to the hottest spots in a city [37].

2.3.2.6 Ease of use

These applications are simple to understand and the user will not need a lot of time to learn how to use them as they are self-explanatory.

2.3.2.7 The CitySense and MacroSense Applications Sensors

The CitySense and MacroSense Applications use GPS sensors. The CitySense and MacroSense applications collect real-time data from different GPS-enabled cell phones, GPS-enabled cabs and cell phone tower triangulation.

2.3.3 CenceMe Application

CenceMe is a personal application that enables members of social networks to share their sensing data with their buddies in a secure manner. This application captures the user's status in terms of his/her activity (e.g., sitting, meeting friends, walking), disposition (e.g., sad, fine, happy), habits (e.g., at the coffee shop, work, gym) and surroundings (e.g., hot, noisy, bright, high ozone). CenceMe inserts a sensing presence into popular social networking applications such as MySpace, IM (Skype, Pidgin), and Facebook allows new levels of “connection” and implicit communication between friends in social networks. The CenceMe system is implemented on a number of standard and sensor-enabled cell phones which can be activated on a per-buddy basis. This application exposes different degrees of a user’s sensing presence; these services include life patterns, my presence, friend feed, social interaction, significant places and ‘buddy search’ and ‘buddy beacon’.

2.3.3.1 CenceMe security and privacy

Users’ raw sensor feed and inferred information are securely stored in the CenceMe database, but can be shared by CenceMe users according to group membership policies. The data becomes available only to users that are already part of a CenceMe buddy list. CenceMe buddies are defined by the combination of buddy lists imported by registered services (Pidgin, Facebook, etc.). Users can decide whether to be visible to other users via the buddy search service or via the buddy beacon service. CenceMe users are given the ability to further apply per-buddy policies to determine the level of data disclosure on per-user, per-group, or global level. In addition to user-specific data sharing policies, the system computes and shares aggregate statistics across the global CenceMe population. For this service, shared information is anonymized and averaged, and access to the information is further controlled.

2.3.3.2 CenceMe Application Risks

The CenceMe application allows members of social networks to share their sensing presence with their ‘buddies’ securely. CenceMe permits the collection of physical and virtual sensor samples, and the storage, presentation and controlled sharing of inferred human sensing presence. It is certain that this application is extremely beneficial, but it will be destroyed by privacy if it is misused.

2.3.3.3 CenceMe Platforms

The CenceMe application is compatible with cell phones like the Nokia N80 and N95; PDAs like the Nokia N800; phone/PDA hybrids like Apple iPhone; embedded sensor platforms like Nike+; recreational sensor platforms like Garmin Edge , SkiScape and BikeNet; and laptop/desktop computers.

2.3.3.4 CenceMe Tools

CenceMe exploits the availability of the following sensors: embedded cameras, laptop/desktop web cameras, microphone, accelerometer, GPS, radio (e.g., BlueTooth device contact logs, 802.15.4 ranging, 802.11 localization, temperature, light, humidity, magnetometer, button clicks, and device state (e.g., ringer off).

2.3.3.5 How CenceMe attract users

CenceMe provides the following services: Life Patterns, My Presence, Friends Feeds, Social Interactions, Significant Places, Buddy Search, Buddy Beacon and Above average. All these services attract users to the CenceMe application.

2.3.3.6 Ease of use

CenceMe application is self-explanatory, simple to understand and the users won’t need a lot of time to learn how to use it.

2.3.3.7 CenceMe Application Sensors

The sensing clients focus on gathering information from mobile user communication computing devices, including a number of sensors enabled cell phones. These cell phones have physical sensors (e.g., accelerometer, camera, microphone) embedded in off-the-shelf mobile user devices. CenceMe exploits the availability of the following sensors: embedded cameras, laptop/desktop web cameras, microphone, accelerometer, GPS, radio (e.g., BlueTooth device contact logs, 802.15.4 ranging, 802.11 localization, temperature, light, humidity, magnetometer, button clicks, and device state (e.g., ringer off) [38].

2.3.4 Green GPS

Green GPS is a participatory sensing navigation service that map fuel consumption on city streets, to allow drivers to find the most fuel-efficient routes for their vehicles between arbitrary end-points. Green GPS uses data collected by individuals from their vehicles and relies on data collected from mathematical models to find the most fuel-efficient routes. The most fuel efficient route doesn’t have to be the shortest or the fastest route, it may depend on the vehicle. This service uses the measurements of vehicular fuel consumption sensors at the OBD-II interface standardized in all vehicles sold in the US since 1996. OBD-II interface gives access to most gauges and engine instrumentation which monitors the health of the automobile. Fuel consumption, coolant temperature, engine RPM and vehicle speed are examples of monitored measurements [4, 39].

2.3.4.1 Green GPS security and privacy

As previously mentioned any beneficial service can be destroyed by privacy issues or by misuse. Green GPS is one of those services that has privacy challenges since a large class of participatory sensing systems monitor location information continuously. This monitoring may lead to significant privacy issues and simple anonymization of data will not be good enough for these situations. The GPS traces may lead to privacy breaches such as revealing the user identity by knowing their home location. To overcome privacy challenges and issues for this service, individuals can simply turn off data collection devices when they feel the need for privacy. Also, different techniques are used to preserve privacy such as data perturbation [39].

2.3.4.2 Green GPS Risks

This service will have increased security risk if it is used wrongly. Location data may be collected everywhere and at any time without user interaction. It can reveal personal information such as the user was at a specific medical clinic at a specific time. The omnipresence of location information may raise the risks of violence and stalking, if perpetrators are able to access location information. Moreover, the consistent use of this service can reveal patterns of the users' day-to-day life.

2.3.4.3 Green GPS Framework

2.3.4.3.1 A Participatory Sensing Framework

A participatory sensing framework called PoolView is utilized to implement Green GPS. This framework facilitates developing data collection applications. A client-side interface is provided by PoolView for data upload and all data is delivered to a central server called the aggregation server. GreenGPS was implemented by writing the aggregation server to PoolView and any individual who wants to share his/her OBD-II sensor data can download the client side software of PoolView. Then, the data can be uploaded to the GreenGPS aggregation server [39].

2.3.4.4 Green GPS Tools

Green GPS is possible, thanks to the On-Board Diagnostic (OBD-II) interface and the manufacturers of OBD-II scanners. Several commercial OBD-II scanner tools are available that can read and record these sensor values. Also, remote diagnostic systems such as, GM's OnStar, BMW's Connected Drive, and Lexus Link are capable of monitoring the car's engine parameters from a remote location (e.g. home of driver of the car). Green GPS uses a typical scanner tool and a vehicle's OBD-II system in conjunction with a participatory data collection framework to enable collection and upload of fuel consumption data [39].

2.3.4.5 How Green GPS attract users

Green GPS users might be attracted to the service benefits which include savings on fuel or reducing CO2 emissions and the carbon footprint. The increase in the use of Bluetooth devices (e.g., cell- phones) and in-vehicle Wi-Fi, Green GPS can be supported by inexpensive OBD-II-to-Bluetooth or OBD-II-to-Wi-Fi adaptors. This will enable uploading OBD-II measurements to applications on the driver's cell phone. Moreover, it can be supported by scanning tools which read and store OBD-II measurements on storage media such as SD cards [39].

2.3.4.6 Ease of use

Green GPS maintains the map of a given area as an Open Street Map (OSM). OSM is the equivalent of Wikipedia for maps, where data is collected from different free sources and an editable street map of the given area is created in an XML format. The OSM map is a directed graph that consists of nodes, ways, relations, and three basic object types. A node has stable coordinates and expresses points of interest (e.g. junction of roads, Marriott hotel). A way is an ordered list of nodes with tags to specify the meaning of the way, e.g. a road, a river, a park. The user will not face difficulties while using Green GPS because it is simple and easy to use [39].

2.3.4.7 Green GPS Sensors

Individuals who would like to contribute OBD-II data to Green GPS install, in their vehicle, a commercial OBD-II scanner along with a GPS sensor.

2.3.5 Microsoft SensorMap Application

The Microsoft SensorMap allows for a general framework where users can choose to publish any kind of sensor data. The sensor data published by a user can be their audio or video feed, location information or text which is typed on a keyboard. The goal of the Microsoft SensorMap is to store and index the data in a way that is efficiently searchable. The SensorMap application enables the users to index and cache data. The indexing and caching allow the users to issue spatio-temporal queries on the shared data. The Microsoft SensorMap is part of the SenseWeb project, which allows sharing and exploring of sensor streams over geo-centric interfaces [4].

2.3.5.1 Microsoft SensorMap security and privacy

Big concerns for sharing physical, real-time data include privacy and data ownership. Sensor data might disclose other information about users and their surroundings [40].

2.3.5.2 Microsoft SensorMap Risks

As with other applications the security risk increases, if there is unauthorized access to back-end services where an unauthorized user can track individuals' information. These issues were resolved by using an authentication framework and enabling the user to decide his privacy level.

2.3.5.3 Microsoft SensorMap Platforms

The SensorMap GUI is based on Windows Live Local and therefore provides features such as street maps, zooming, satellite images, panning, and 3D views. In addition, it lets end users pose queries on available sensors [40].

2.3.5.4 Microsoft SensorMap Tools

SensorMap is a centralized Web portal consisting of four components: the GeoDB database, the DataHub Web service, the Aggregator for creating icons, and the SensorMap (GUI). SensorMap uses many sensors that already exist on the Web. Some of these sensors are online traffic cameras and real-time stream-gauge information. SensorMap includes, crawler that addresses the problems in a narrow domain. It can automatically detect traffic cameras available on the Web and annotate them with their latitude and longitude [40].

2.3.5.5 How Microsoft SensorMap attract users

Microsoft SensorMap stores and indexes the data in a way that is efficiently searchable.

2.3.5.6 Ease of use

This application is easy to understand.

2.3.5.7 Microsoft SensorMap Sensors

This application depends on the following sensors cameras, microphone, accelerometer, GPS and radio sensors.

2.4 Clustering

Clustering is an important matter in the analysis and exploration of data. The tendency of people to come together and form clusters is inherent in the structure of society. Clustering consists of discovering natural groups of similar elements in data sets.

One of the following procedures is to convert sparse matrix into link-list data structure. Since a two-dimensional array to represent a sparse matrix leads to the wastage of a substantial amount of space, a different representation is used for sparse matrices. One such representation is to keep only non- zero elements with their column positions and row positions. This means that each non-zero element is symbolized by using triples (i, j, value), where *i* is a row position and *j* is a column position, and these triples will be stored in a linear list. These triples can be arranged in the increasing order of row indices, and for the same row index in the increasing order of column indices. Each triple (i, j, value) can be represented by using a node having four fields as shown in the following:

```
Struct snode{
Int row, col, val;
Struct snode *next;
};
```

As far as social network clustering analysis is concerned, one of the clustering algorithms is shown below [12]:

```
Input:
$Lu$ : Edge creation Lists
$Lp$ : Edge pointed Lists
Begin
$G = Lc$ * Swap ($Lu$ ) //perform swapping row column of Lu
for k=3 to m do
$Gk - 1 = Gk - 2 * G$
```

Lp : Edge pointed Lists

```
Begin
$G = Lc * Swap(Lu)$ //perform swapping row column of Lu
for k=3 to m do
$G_{k-1} = G_{k-2} * G$
$R = I \vee G \vee G2 \ldots \vee G_{m-1}$
$Q = R \wedge Swap(R)$ //perform swapping row column of R
$Q_k -> C$
$(C_k, Q)$->Relation $(C_k)$
End
```

where various variables stand for:

G: One step reachable matrix between objects

Lc : Edge creation Matrix

m: Number of objects

Gk: K step reachable matrix between objects

V: Boolean sum

I: Unit matrix

R: Reachable matrix

Q: Mutual reachable matrix

^: Boolean product

Qk− > C : generating clusters through mutual reachable matrix *Q*

(*C k* ,*Q*)->Relation(*Ck*): identifying relationships among clusters base on clusters and one-step reachable matrix

2.4.1 Addressing community representation problems

Exploring communities is a significant function in social network analysis. These communities are currently specified using clustering methods to group actors. This approach usually results in actors belonging to one and only one cluster, while in real life a person may belong to various communities.

This community representation suffers from two problems:

a. Clustering ambiguity

When an actor is connected to two or more communities: 1) most clustering algorithms place the actor in one of the communities 2) others place shared actors between their communities, solving the unique assignment problem but increasing link crossings when several nodes belong to several communities; 3) in a few clustering algorithms, communities that share actors are visualized as overlapping clusters, increasing the visual complexity of the graph by introducing node overlap and link crossings due to the tight space packing.

b. Readability:

When two communities share many connections, their links intersect and cross several nodes, hindering the identification of the particular actors connected. Layouts with numerous overlapping nodes and edge crossings are qualified as having a high visual complexity.

As a solution the authors in [41] propose duplicating actors in social networks. Actor duplication in social networks is used to assign actors to multiple communities without greatly affecting the readability. Based on paper [41] experimentation, the following general guidelines for node duplications in clustered graph representations are proposed:

When to duplicate?

- To minimize visual complexity in graphs that contain many actors shared between communities.
- To highlight central actors that connect multiple communities importance. These actors may be extracted from their communities when duplicated.
- To supply accurate community-centered views. It is a significant property of many social network analysis tasks.

How to duplicate?

- Using clone or split, but not both of them, since they are both complex representations.
- Clone requires less practice (at the expense of cluttering the network), so it can be used as base case.
- Split minimizes visual complexity, but interactive highlighting of the duplication links may be required for novice users.

How to visualize duplication?

- It is not enough to use simple colored nodes for representing duplications.
- It is more effective to use links between duplicates.
- Visual links used to connect duplicates should be easily distinguishable from other graph links in order to improve readability.
- It is preferred to use interactive highlighting of duplicated nodes and links.

2.4.2 Group stability

Groups, meanwhile, exist only as long as individuals are interested in becoming members of them. Much attention has recently been devoted to the task of identifying and understanding groups and communities in social networks [16]. The effectiveness of groups can be undermined when group members depart, taking with them, experience, resources and possibly other group members. The ability to predict the stability of groups is highly desirable, as it offers insights on factors that affect online group effectiveness. It also provides practical guidance to tasks such as risk management and customer retention.

In some online settings an individual can belong to only one group at any given point in time. In such settings the group serves as the main engagement platform for the individual. An individual who is not satisfied with his/her group will quit the group and join another one. The reasons for dissatisfaction can be plenty. In such cases the percentage increase/decrease in the number of group members over previous time periods is a good measure in determining whether a group is stable or shrinking. A group, though accumulating members over time, may still be a shrinking group, if most members do not participate in the group's activities. In addition, most previous studies treat all group members as equals when performing group evolution analysis. Groups are often led by a smaller set of leaders who have considerable influence over other group members. The authors in [42] study the problem of group stability, i.e., why some groups fall apart and disappear while others thrive. The analysis in [42] has shown that it is possible to predict group stability with high accuracy using a range of features that describes the group composition, activities within the group and structural aspects of a group. The study also shows that it is important to choose features from multiple perspectives; in fact, combining diverse features is essential to predictor performance.

2.5 Privacy

An online social graph is usually used to model data stored in an online social network. Users are modeled as nodes, while social connections between users are modeled as edges. Online social networks attract users by providing social interactions and communications which increases the privacy and security concerns.

Online social networks conserve a massive amount of private and sensitive information on users. Users are motivated to join an online social network in order to create a profile, and use various applications provided by the online social network. Revealing users' information is a double edged sword. On the one hand, revealing information improves the social interactions and relations. On the other hand, disclosing personal information may lead to identity theft, stalking, phishing, and spamming.

Network Security standards include confidentiality, integrity and availability. Confidentiality is a serious issue in online social networks. Confidentiality in the context of online social networks has four categories: (i) User's identity anonymity: The protection of a user's identity; (ii) User's personal space privacy: The visibility of a user's data; and (iii) User's communication privacy: The communication privacy has to be achieved. Confidentiality requirements are important. To achieve data privacy, access control is required to prevent unauthorized entities from accessing users' data.

Design conflicts exist between traditional design goals (usability and sociability) and security and privacy goals. These conflicts include: (i) Privacy vs. Social Space Exploring: There is a trade-off between privacy and search capabilities. More data should be revealed in order to achieve more efficient searches; (ii) Privacy vs. Social Interaction: The main functionality of online social networks is social interaction, but it might lead to a privacy leakage; (iii) Privacy vs. Data Mining: an adversary may access users' data on online social networks which will affect users' privacy. Even after users' identities are hidden, adversaries can recover most users' identities; and (iv) Client-Server vs. P2P Architectures: A client-server architecture has advantages over a P2P architecture in achieving traditional goals of OSNs.

The following are some research directions for reducing the design conflicts between the design goals of OSNs: (i) developing an Enriched Relationship Model for OSNs: by extending the relationship model to include relationships type categorized into bidirectional relationships such as friend or colleague or one-directional relationships such as fans or followers, identifying the trust strength, and using interaction intensity to measure the quantity and quality of interactions between users; (ii) Protecting Online Social Graphs: An online social network is a social graph that connect users, so it should be protected. Trust relationships and real life connection patterns embedded in social networks can be used to protect, the online social network graph; (iii) Defense against Social Link Forging Attacks: The nature of OSNs makes it difficult to forge social links; and (iv) Defense against Node Identity Forging Attacks [43].

Role-Based Access Control (RBAC) is used to control access to information. RBAC is extended in many researches to achieve specific requirements in such a way that a user will be able to make many collaborative groups and assign each user a collaborative relationship. Typically, close collaborative relationship users can share more information than others.

2.5.1 Measuring user privacy

The research in [44] proposes a technique to achieve users' privacy and increase privacy awareness. It proposes a framework that measures users' privacy with respect to some specific people of their choice. This framework ensures users privacy by comparing the user privacy with others in his/her friend list. The steps included in the calculation of privacy strength for a user in the proposed framework are:

1- Input to the framework: The user creates a Friend Set which is a set of users in the user's friend list with whom they want to compare their privacy.

2- Response matrix will be created: For n dichotomous variables for N users in the Friend Set a Nxn dichotomous response matrix will be created. If $1 \leq j \leq N$ and $1 \leq i \leq n$ then for an item i being shared by an individual j the value of jth row and ith column is marked as 1 otherwise is marked as 0.

3- The best model that will fit the response matrix will be selected. One of the below models will be selected:

   A. One Parameter Logistic Item Response Theory Model

   B. Two Parameter Model

   C. Naive Model Approach

4- Model Selection:

   The Naïve model is based on population and it cannot differentiate well between the sensitivities of two profile items. That is why it will be used only if the number of users is less than the number of profile items.

   To select the best model, the Information Criterion (IC) of the model should be calculated. Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) will be utilized for model selection. AIC and BIC values gives us the loss of information. AIC can be calculated as follows [44]:

$$\mathrm{AIC} = -2(\log - \mathrm{likelihood}) + 2\mathrm{K} \tag{1}$$

   BIC can be calculated as follows [44]:

$$\mathrm{BIC} = -2(\log - \mathrm{likelihood}) + \mathrm{K}\ln(\mathrm{N}) \tag{2}$$

   where K is the number of parameters used in the model. These values make sense only when it is compared with the other models. The model having the least value of AIC and BIC is preferred over all the models.

5- Calculation of Privacy Quotient and choosing the ranges:

To calculate the privacy quotient the below equation is used [44]:

$$PQ(j) = \sum_{i} \beta_i * P_{ij}(\theta_j) \qquad (3)$$

where βi is the sensitivity and Pij(θj) is the probability of sharing an item i by an individual j with an ability of θj .

6- Labelling the user privacy strength

It is easier to understand text than numbers. The range of privacy quotient will be labeled. Various labels will be assigned to the ranges such as ”High PQ”, ”Good PQ”, ”Average PQ”, ”Below Average PQ”, ”PoorPQ ”. The privacy strength is better when the privacy quotient decreases [44].

2.5.2 Measuring Privacy Leaks

Unstructured data is usually more vulnerable to attacks. Privacy Armor is a proposed model to alert users if they are sharing some sensitive data online intentionally or unintentionally. This model aims to achieve date protection and minimize the intrusion into a user’s privacy. It uses the Item Response Theory to maintain the privacy of unstructured data [45].

A. Crowdsourcing and Data Collection

Crowdsourcing method is used to collect information about the items being shared on the user’s profile. If the users have shared the data on purpose it will be stored as 1, otherwise it will be stored as 0. The outcome will be a N X n dichotomous response matrix where n will be the number of profile items and N will be the number of users.

B. Selecting a Model and calculating the Privacy Quotient

The Naïve approach is a simple and easy to follow, but the sensitivity values calculated are biased by the user population. If the user does not like to share data, then the calculated sensitivity will be high which is not accurate. The Item Response Theory model is used to calculate privacy scores because the real world data is too messy to fit effectively.

Perivacy leakage is calculated as [45]:

$$\vartheta = \frac{\sigma}{\beta} * 100 \tag{4}$$

Where σ is $\sum_k \sigma_i$ here k are number of sensitive items in the post and σ_i is the sensitivity of the i^{th} profile item.

β is the total sensitivity of all the n items.

Average privacy quotient [45]:

$$\mathrm{Avg(PQ)} = \frac{1}{N}\left(\sum_j \mathrm{PQ(j)}\right) \tag{5}$$

Privacy quotient using the mean value theorem [45]:

$$\mathrm{Req(PQ)} = \frac{1}{b-a}\int \sum_i \beta_i * V(i,j) \tag{6}$$

Where i varies from $1 \le i \le n$ a is the lowest Privacy Quotient obtained; b is the highest Privacy Quotient obtained.

The privacy armor will send an alert to the user showing the privacy leak percentage, If Avg(PQ) < Req(PQ). The privacy leak percentage can be calculated using the below equation [45]:

$$\vartheta = \frac{\mathrm{Req(PQ)} - \mathrm{Avg(PQ)}}{\mathrm{Req(PQ)}} * 100 \tag{7}$$

Chapter 3

3.1 System Architecture and Design

This chapter addresses the problem statement. Then, it discusses the design challenges and how to overcome them. It also presents the expected impact of this research and examines the contribution made in this research. Finally, it lays out the architecture design including the conceptual framework, framework implementation, privacy model and how to calculate and show burned calories to the user.

3.2 Problem Statement

In the United Arab Emirates, currently, more than 66 percent of men and 60 percent of women in the UAE are overweight or obese. Obesity is a growing health concern in because it is associated with health risks such as diabetes and cardiovascular disease. Health officials state that obesity is one of the major causes of preventable death [46].

Keeping active and exercising can help people to stay healthy and reduce health risks such as diabetes, high blood pressure, stroke, and cancers. Even a little exercise fights obesity and stress. In spite of all the benefits of physical activity and exercising, people generally don not exercise much.

In this research, an effort is made to extend sensing abilities of online social networks by injecting more sensing features and then processing and clustering some of this information for fruitful use using a locally developed application. Data privacy is enforced using data levels and user roles.

The solution is developed to improve social networking in the area of healthcare. This application is expected to motivate people to exercise, enable doctors to see specific information such as heart rate, and enable the users to track and locate their friends for sharing this specific data.

‘‘Care’’ is a sensing application that enables members of a typical social network platform to share their location information with their friends in a private manner. The goals of the development are: (i) to foster and extend the integration of WSNs and OSNs; (ii) to improve social networking; (iii) to create healthcare fun and motivate people to exercise; and (iv) to track and locate friends and family members and generate clusters based on a criterion. The application is expected to allow new levels of “connection” and implicit communication between contact groups in social networks, with the following services:

- Showing nearby places
- Sharing the user location
- Showing nearby friends
- Clustering nearby friends
- Calculating and sharing distance moved, calories burned and active time
- Getting and sharing weather temperature
- Sharing pictures
- Calculating, tracking and sharing the user heart rate

All of these services should motivate the user to exercise by competing with friends to see who has exercised more and burned more calories. In addition, the user can check if any of his friends are nearby so he/she can walk with. The application enables the users to track their heart rates and share the data with their doctor. This will help the user to take their fitness to a new level and track their results and progress.

As highlighted in the previous chapter, one of the major concerns of online social networks is data security. As disclosure of users’ personal details may lead to identity theft, embarrassment, and market benefits, this research proposes a privacy model to ensure individuals privacy. It aims to increase information sharing, increase social interactions and minimize the privacy threats. There is a trade-off between privacy and information sharing, that is why a privacy model is required to preserve the privacy of user information but not at the cost of information sharing. The privacy model applied in this research uses roles and relationships to enable the user to control his/her information sharing.

3.3 Design Challenges

Convenient design gives rise to other important challenges that need to be solved in order to enable the development of successful mobile sensing applications which meet user needs. Before describing the implementation, the challenges encountered while implementing the application will be discussed:

- Mobile limitations: Mobile phones have very good computational efficiency, but they are limited for developers in programming and resource usage control.
- Energy limitations: Application developers on mobile phone platforms should be aware of power consumption when developing an application that depends on using radio interfaces such as a GPS.
- Privacy issues: The collected data such as location data contains sensitive and personal data. It is highly important to apply a privacy model in order to maintain data security.
- Data management: The data collected is huge, so it is extremely important to efficiently process the huge amounts of data.
- Simplicity: the application should be easy to understand and use for the non-expert.
- Determining user location: A GPS only works outdoors, and it quickly consumes phone battery power. Android’s Network Location Provider helps to determine

user location using Wi-Fi signals and cell tower. This strategy helps to provide the location details indoors and outdoors and uses less battery power. The application developed in this research uses a combination of both strategies: GPS and Network Location Provider.

3.4 Expected Impact

The application uses various sensors to acquire relevant data and display it on a user device. The user can check-in to one of the displayed places, and this, in turn, is named as a single visit to a location. Thus, the user can get his/her friends location based on their last check-in. The user should also be able to see if any of his/her friends are checked in nearby. For this, the displayed nearby friends need to be clustered to nearby colleagues, family and so on. In addition the application should enable the user to calculate the distance walked, the duration spent during the walk and the walking related burned calories. The user can track his/her heart rate by placing his/her finger on the phone's camera to find his/her target heart rate in just seconds. This application will help the user to be aware of her/his health and be conscious of it. The user can share this information with friends. The solution enables the user to find nearby places to walk and nearby friends to walk with. It helps the user to determine a target heart rate so the user can work out without putting too much stress on his/her heart. Using this application, the user can not only track his/her heart rate but also how many calories he/she is burning while exercising. Moreover, it helps the users to optimize their exercise and track their progress. Social constraints such as privacy are also addressed in this application.

The developed application enables members of social network to share their information with their contacts in a private manner. A privacy model is developed to achieve users' privacy and security.

3.5 Contribution and Description of New features

The application is expected to be implemented on a number of standard and sensor-enabled android cell phones to offer a number of services, which can be activated on a per-buddy basis to expose different degrees of user data. Based on an application platform and sensors, a number of services that meet the goals in the previous section are detailed here. These are stated as follows:

3.5.1 Application Platform

There are several popular OSN platforms. The Facebook is the most popular OSN platform today. This application is developed for such a platform and is compatible with Android devices. For development, the Facebook supports different APIs for developers: (i) The Graph API, which is a simple HTTP-based API that gives access to the Facebook social graph, uniformly representing objects in the graph and the connections between them. Most other APIs are based on the Graph API; (ii) The Open Graph API allows applications to tell stories through a structured, strongly typed API; (iii) Facebook offers a number of dialogs for Facebook Login, posting to a person's timeline or sending requests; (iv) The Facebook Query Language (FQL) enables the developer to use a SQL-style interface to query the data exposed by the Graph API. It provides some advanced features not available in the Graph API such as using the results of one query in another; and (v) The Public Feed API lets the developer read the stream of public comments as they are posted to Facebook [47].

3.5.2 Application Sensors

The application uses the built-in GPS of the user mobile device to get current location coordinates. It also uses the temperature and humidity sensor to check the weather temperature. Furthermore, it uses the mobile device built-in camera to get the user heart rate.

3.5.3 Application Services

The services, which can be used using this platform, are:
Show Nearby Places: This application enables the user to use the built-in GPS of the mobile device to get the current location. It displays a list of nearby places as well.
Public Location Badge: The user can post his/her location directly on Facebook to increase visibility of information to other users.
Show Nearby Contacts: The user gets his contact location based on his/her last Check-in. The user can see if any of his/her friends are checked in nearby. The application displays a list of nearby friends, place and the time they checked in.
Cluster Nearby Contacts: Clustering is important in analysis and exploration of data. This application clusters nearby friends into groups based on colleagues, family and so on.
Tracking Distance Moved, Calories Burned and Active Time: This application tracks distance moved, calories burned and shows active time for the user. The application can also be used for running, cycling, walking and all other distance-based outdoor sports. Once data is shown on the network, the user can seek extra encouragement from friends and family to workout. The clinical staff from a healthcare center can also monitor shared clinical data.
Get weather temperature: This application enables the user to use the built-in sensors of a mobile device to check the weather temperature and share with friends.
Share pictures: The application enables image of a user’s specific injured body part or monitored body part once online, which can be seen and examined by a doctor.

*Monitor heart rate :*The application tracks the user heart rate, all he/she has to do is to place the tip of his/her index finger on the mobile’s camera and in few seconds, the user heart rate will be displayed. It will take between ten to thirty seconds to get an accurate result. The application uses the built-in mobile camera to track color changes on the fingertip that is linked directly to the user pulse.

3.5.4 Application Security and Privacy

At this level, the main focus is how users can control the detail and the accuracy of what other users will be able to access and see. In order to maintain, privacy, the user can turn off all or components of this application using the settings option. Another proposed option is to use different levels in data and the user roles based on connectivity. A role represents a group initiated by the user. The idea is that the users in the same cluster with same relationship can get same levels of information. Thus, the user assigns a relationship to each connection. Close relationships provide access to more information. The developed application provides the user with three roles: trusted, semi-trusted, and un-trusted. Each of these roles has predefined permissions. The user will assign role to each member from his contacts list. The permissions of each role will be applied to all of its members. Moreover the user can assign each member of each role a relationship as close or not close. Such a model is proposed in section 3.6.3.

3.6 Architecture Design

In the following section, a framework is developed that describes what is needed to build such an application, and which social parameters need to be linked. Then, the framework implementation and data flow model are discussed. Finally, how the platform addresses privacy will be discussed.

3.6.1 The Conceptual Framework

Based on the mentioned requirements and constraints, a framework is displayed as shown in Figure 3.

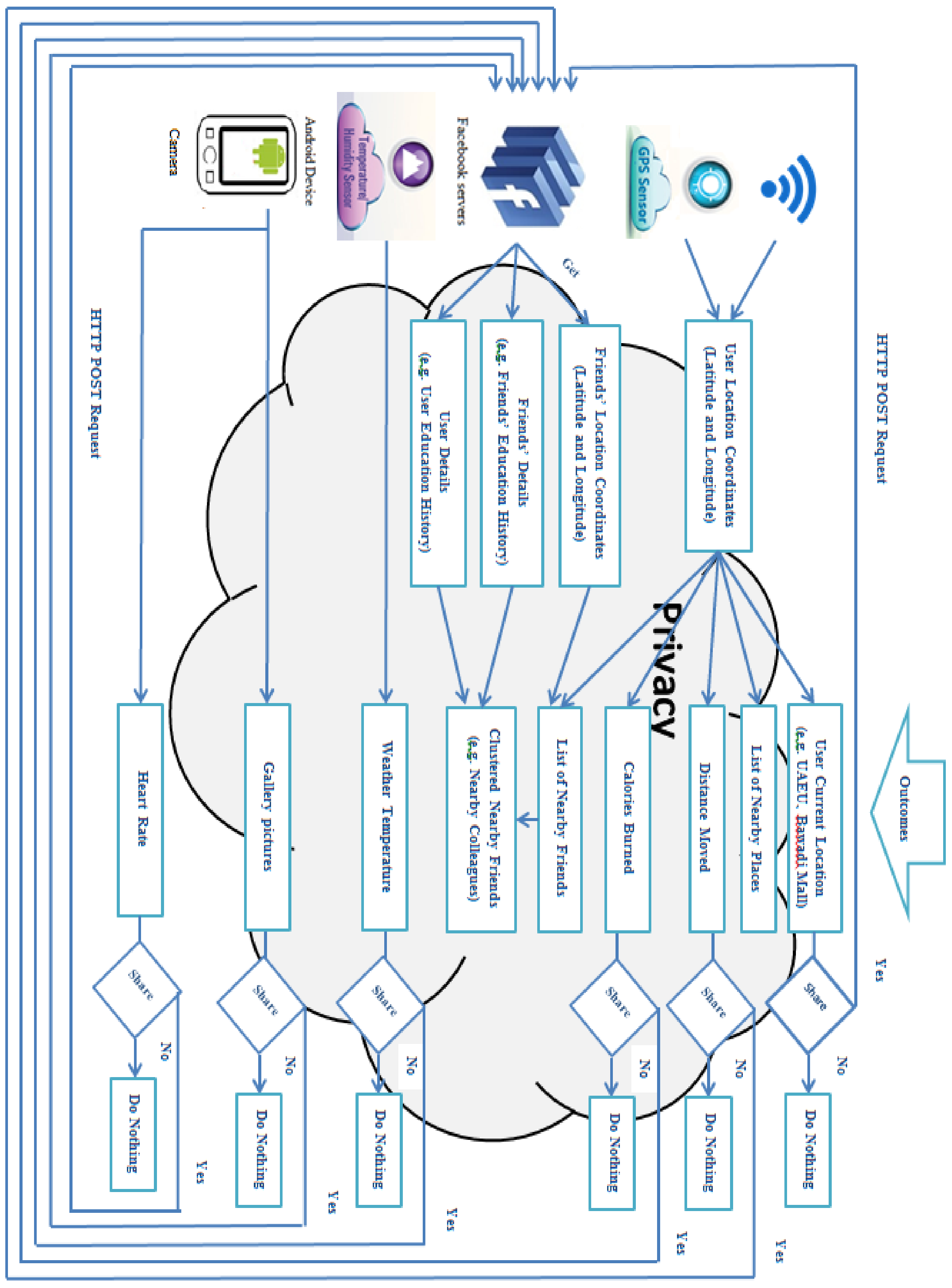

Outcomes
HTTP POST Request
GPS Sensor
Facebook servers
Get
Temperature/ Humidity Sensor
Android Device
Camera
HTTP POST Request
User Location Coordinates (Latitude and Longitude)
Friends' Location Coordinates (Latitude and Longitude)
Friends' Details (e.g. Friends' Education History)
User Details (e.g. User Education History)
Privacy
User Current Location (e.g. UAEU, Bawadi Mall)
List of Nearby Places
Distance Moved
Calories Burned
List of Nearby Friends
Clustered Nearby Friends (e.g. Nearby Colleagues)
Weather Temperature
Gallery pictures
Heart Rate
Share
Yes
No
Do Nothing

Figure 3: The Conceptual framework

3.6.2 Data Flow Model

Based on the framework, the information and process flow inside ‘Care’ can be easily visualized, based on which software architecture of ‘Care’ is shown in Figure 4. For the purpose of implementation, the various stages in information and process flow are discussed below, and displayed in Figure 5.

A. Sensing: Most Android-powered devices have built-in sensors that measure motion, orientation, and various environmental conditions. These sensors prove raw data with precision and accuracy. The platform supports three broad categories of sensors: (i) position sensors to measure the physical position of a device. This category includes orientation sensors and magnetometers; (ii) environmental sensors to measure various environmental parameters, such as ambient air temperature and pressure, illumination, and humidity. This category includes barometers, photometers, and thermometers; and (iii) motion sensors to measure acceleration and rotational forces along three axes. This category includes accelerometers, gravity sensors, gyroscopes, and rotational vector sensors. In this application, position and environmental sensors are used.

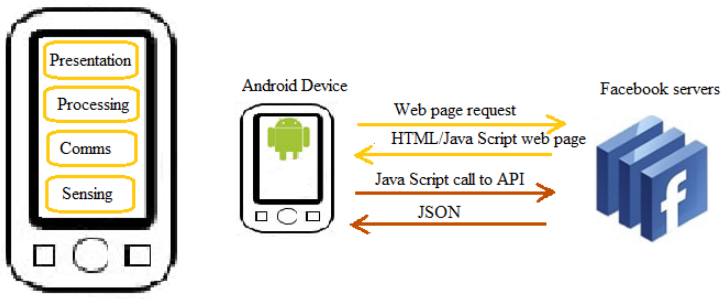


Figure 4: Software architecture of Care

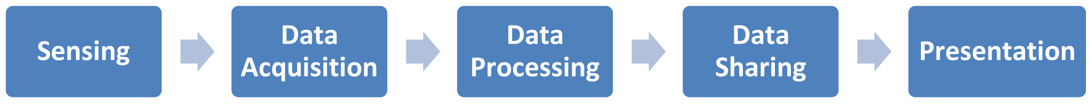


Figure 5: Information and process flow in the ‘Care’ system

B. Data Acquisition: Data acquisition is the process of gathering information in an automated fashion from analog and digital measurement sources such as sensors. Apart from using position and environmental sensors to get position (i.e., latitude and longitude information) and weather information respectively, the application uses Facebook APIs such as Graph API and Facebook Query Language to extract the user education history, work history, friend's last check-in coordinates, friends education history, friends work history and nearby places.

C. Data Processing: The solution analyses and processes the extracted data to produce meaningful information. After getting the user location coordinates and his/her friends' location coordinates, the application measures the distance between the user and each one of his friends to produce a list of nearby friends. The application compares the user education-history and other details with friends' education-history to cluster the nearby friends into groups such as colleagues, family and so on. Additionally, the application uses the acquired location coordinates taken frequently to measure the distance walked, the related duration and the calories burned.

D. Data Sharing: The application enables the user to share location, the distance walked, burned calories, the weather temperature and pictures. The Graph API updating is done simply with an HTTP POST request to a relevant endpoint with the updated parameters. To publish and share new data, the application uses POSTs HTTP requests to appropriate URLs.

E. Presentation: After processing the data, the application displays a list of nearby places. The user will be able to check in to any of these places by clicking on one of the places. It also has a nearby friends icon, by pressing on this icon, the user will get a list of his/her nearby friends based on their last check-in. 'Care' organizes nearby friends into groups of colleagues, work friends, family and others. Moreover, it displays the distance the user walked, total calories burned and the weather temperature.

3.6.3 Privacy Model

Data privacy is a major concern. In order to protect users' shared data, data can be handled in many ways: (i) by bifurcating table into shareable and non-shareable columns i.e. data partitioning, and then assigning the rights. This will work only when you know exactly which field is to be shared and your table entries are fixed (i.e. not dynamic). However for data porting and scaling up the options, you may face problems; (ii) by using MongoDb. MongoDB is a no-sql database and works on documents rather than entries. It is also hierarchical and supports dynamic data (i.e. table entries cannot be fixed); (iii) (Jugar option) by inserting one column or rights to all your data rows. In the first part of the query, a check is done to see whether rights are correct, then further processing of the query is allowed, otherwise it is rejected; and (iv) by using Role Based Access Control (RBAC) mechanism. In RBAC, permissions are associated with roles, and users are to be assigned to appropriate roles. This helps to simplify management of permissions. Roles are similar to the group's concept in access control. A role is defined as a set of users on one side and a set of permissions that will be applied to the users on the other side, while groups are defined as a set of users only. The most appropriate technique for 'Care' is RBAC. The privacy model applied to 'Care' is derived from Role Based Access Control (RBAC).

In the proposed model, the roles are generated for various trust levels and contacts are assigned roles based on their relationship with the user. Contacts can be easily reassigned from one role to another. The developed application has predefined role-permission relationships, which makes it simple to assign contacts to the predefined roles. It is difficult, without the new privacy model, to determine what permissions have been assigned to what users. The proposed privacy model helps to perform large-scale authorization management. The basic concept of the proposed privacy model is that the user assigns his contacts to roles, roles have predefined permissions, and contacts acquire permissions by being members of roles. User-role can be many to many, which means that the same user can be assigned to many roles.

The application provides the user with three roles: (i) trusted, (ii) semi-trusted and (iii) un-trusted. The user will set the members of each role according to his/her relationships with his/her contacts. Trusted role has predefined permissions that will enable its members to view and comment on most of the user posts. Semi-trusted will enable its members to view and comment on some of the user posts, while untrusted role members will not be able to view or comment on many of the posts.

Moreover, the user can assign relationships to his trusted and semi-trusted contacts. The user can assign his trusted contacts a close or not-close relationship. Assigning a close relationship to contacts will give them the option to view more posts, while not-close relationship will not give them the permission to view more details. Using this privacy model, the users can allow their contacts to share certain levels of their information. This model helps in managing different levels of information sharing. Contacts will not know what role or relationship the user has assigned them. The example of such a model is shown in Figure 6.

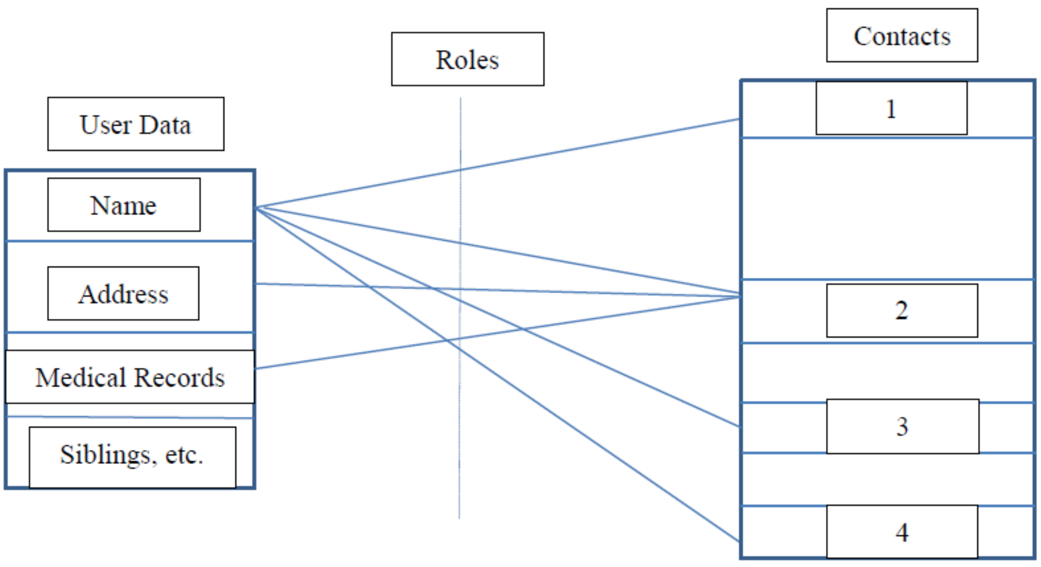


Figure 6: User-data model

It is clear from Figure 6 that each contact has some level of access to the user data. For example, the contacts 1, 3, and 4 (i.e., friend, colleague, office staff) may only access name of the user, while contact 2 (a healthcare unit staff) may also access address and medical records, though all contacts may be part of same social network.

Consider set of users, roles, objects, operations, collaborative relationships, access levels, and conditions be represented by *U*, *R*, *Objs*, *Oprs*, *CR*, *AL*, *Cond* respectively. Thus, the assignment relations among elements of the privacy-aware model are:

2^R: the power set operations

1. Many to many mapping user-role assignment relation: $URA \subseteq UxR$
2. The set of permissions: $Perms = 2^{(OprsxObjs)}$. This can also be stated as:

$Perms = \{(Objs, Oprs) | Objs \in Objs, Oprs \in Oprs\}$

3. The set of sharing and privacy-aware permissions: $PA\_perms = (Perms, CR, Cond, AL)$

   This can also be stated as:

$PA\_perms = \{(perms, cr, al, cond) | perms \in Perms, cr \in CR, al \in AL, and\ cond \in Cond\}$

4. Many to many mapping permission-role assignment relation: $PRA \subseteq PermsxR$
5. Many to many mapping sharing and privacy-aware permission-role assignment relation: $Prv - PRA \subseteq PA\_perms\ xR$

3.6.4 Burned Calories Calculation

The application will request the user to enter his/her weight in kilograms in order to calculate the calories burned walking. After that, it uses the following equation to calculate the rate of calories burned per pound of body weight [48].

Rate per Pound (Cal/lb-min) = $A+BV+CV^2+KDV^3$ where:

V=Walking Speed (mph) – Limited to a minimum of 1 mph and a maximum of 5 mph

A= 0.0195

B= - 0.00436

C= 0.00245

D= $[0.000801(W/154)^{0.425}]/W$

W=Weight (lbs)

K= 0 or 1 (0=Treadmill; 1=Outdoors)

Chapter 4

4.1 Results & Discussion

This chapter presents the simulation environment and then discusses the simulation steps. After that, it analyses the development results and describes the implementation of a privacy model. Finally, it presents the comparative analysis.

4.2 Simulation Environment

The application is developed for Android operating system devices. Also, it can be developed for iOS by using the Facebook iOS SDK. Android is an open source operating system available to all developers with various expertise levels. Android is a Linux-based operating system created for touchscreen mobile phones and tablets. The Android platform allows users to develop, install and use their applications. The application is primarily developed in Java programming language by using the Android software development kit (SDK). The SDK provides the developers with all the tools they need including software libraries, debugger, sample code, emulator, and tutorials. The integrated development environment (IDE) for developed applications is Eclipse using the Android Development Tools (ADT) plugin. Facebook SDK for Android was used to integrate this application with Facebook's platform.

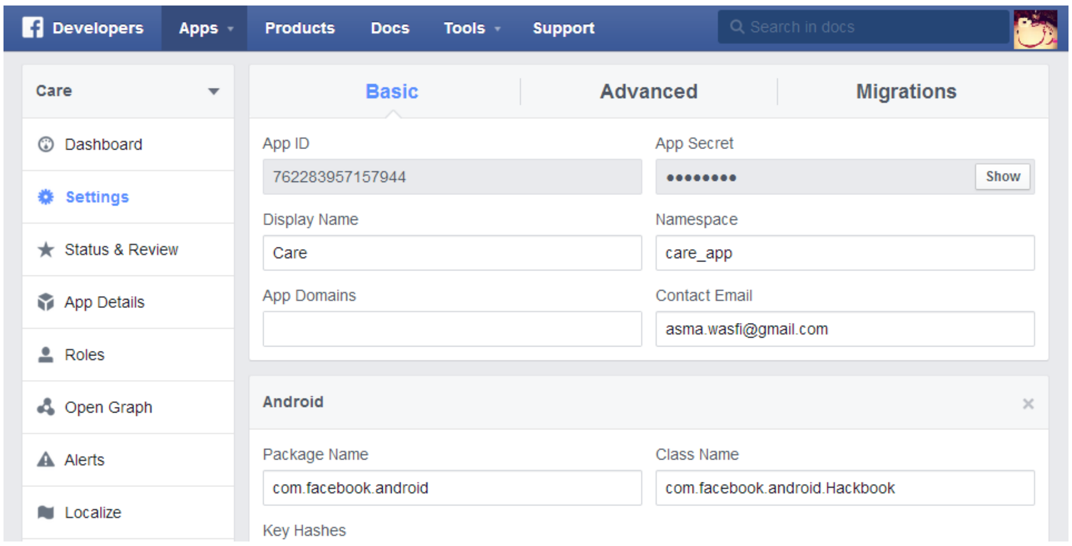
4.3 Simulation Steps

In order to start the simulation, prerequisites including Eclipse, Android SDK, **Android Developer Tools (ADT) Plugin**, and Facebook SDK were installed. Then, Facebook SDK was imported to Eclipse. Every Android app that is developed was signed, as it was required to register each application’s key hash with Facebook as a security check for authenticity. To generate a key hash on a local computer, the

Java’s keytool utility is run, which should be on the consol’s path against the Android debug keystore. On windows the following command was used for to get release key hash:

```
Keytool -exportcert -alias androidreleasekey -keystore %HOMEPATH%\.android\release.keystore | "C:\Users\Asma\bin\openssl" sha1 -binary | "C:\Users\Asma\bin\openssl" base64
```

This resulted in a key hash of 30 characters and was placed on ‘Care’ Facebook profile shown in Figure 7. After registering on the Facebook Developer Site as a developer, ‘Care’ Facebook profile was created and details such as app name, category, and key hash for ‘Care’ were entered. After creating ‘Care’ profile, ‘Care’ app ID was generated and appeared in the profile. ‘Care’ app ID was added to ‘Care’ project files on Eclipse.



App ID: 762283957157944
App Secret: ••••••••
Display Name: Care
Namespace: care_app
App Domains:
Contact Email: asma.wasfi@gmail.com

Android
Package Name: com.facebook.android
Class Name: com.facebook.android.Hackbook
Key Hashes

Figure 7: Care Facebook profile

In order to test ‘Care’ on a real device (Galaxy S4) after developing it on Eclipse: (i) Galaxy S4 is connected to the development machine with a USB Cable and an appropriate USB driver is installed; (ii) USB debugging is enabled on Galaxy S4; and (iii) Run on Android application option is selected to enable Eclipse to install the app on the connected device.

4.4 Development Results

In this section, the main focus is how all the functions, components and services mentioned in chapter 3 are implemented. Each of these is discussed below.

4.4.1 Login

This is implemented in an easy way for people to log in to the application such as ‘Care’. ‘Care’ uses Android, JavaScript and Facebook SDKs to speed up the process and builds login systems quickly. For secure authorization Facebook uses the OAuth2.0 open protocol for confirming a person’s identity and giving them control over right of access to their information. ‘Care’ main list including the login icon is shown in Figure 8.

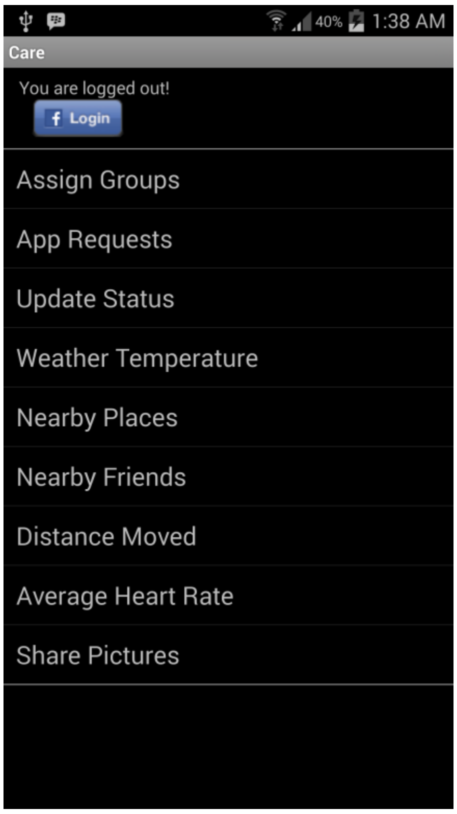
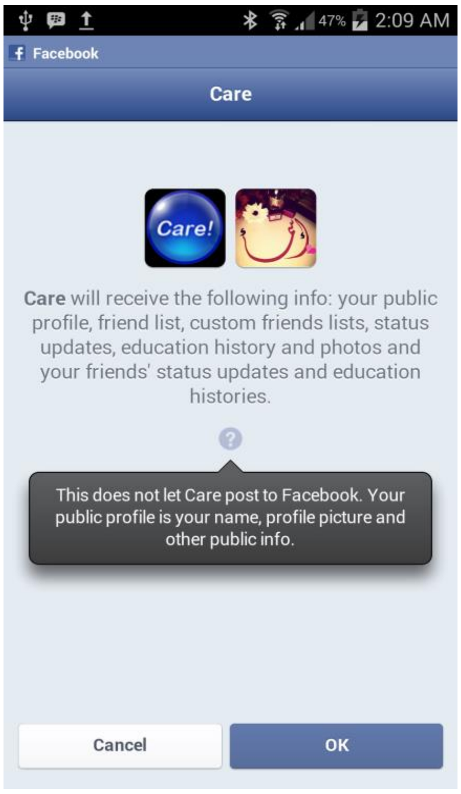


Figure 8: ‘Care’ main list

4.4.2 Permissions

The permissions enable developers to request access to information about someone using their application. ‘Care’ asks for the following permissions offline access, publish stream, publish check-ins, photo upload, user status, user education history, user work history, friends' status, friends' education history and friends' work history. To gain access 'Care' requests the permissions transparently through the Login dialog as shown in Figure 9. The login process uses the following steps: (i) it determines whether someone is already logged in; (ii) if they aren't logged in, it prompts them to do so; (iii) exchanges secure codes to confirm identity and (iv) generates an access token. An access token is a random string that gives an application temporal and safe access to Facebook APIs. In the last step of the login flow the token is generated. The token saves information about which application generated it and when the token will expire as well as information about permissions that have been granted. To maintain information security, almost all API calls at Facebook need to have an access token passed in the parameters of the request [49].



Care will receive the following info: your public profile, friend list, custom friends lists, status updates, education history and photos and your friends' status updates and education histories.

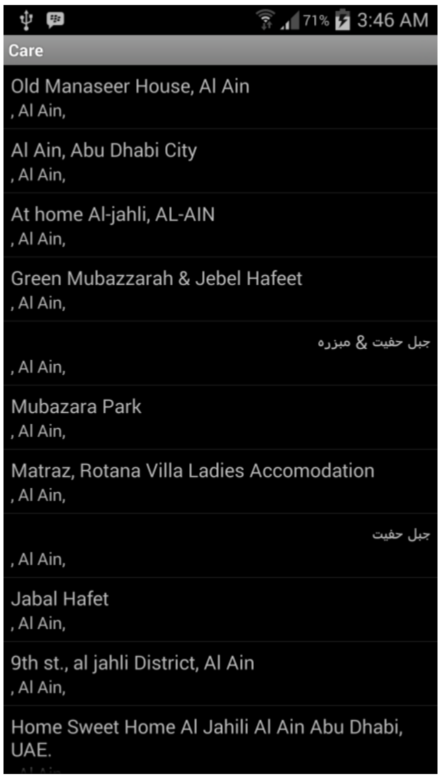
This does not let Care post to Facebook. Your public profile is your name, profile picture and other public info.

Figure 9: Care permission request

4.4.3 Show Nearby Places

The following steps outline how to get a user’s current location, display a list of nearby places, as shown in Figure 10 and check in to one of these places with the Facebook SDK for Android.

1. Set Up the Place Picker Item: This step includes defining a BaseListElement class to represent an item in the list. This class contains member variables that define the user interface (UI) as well as methods that are subclassed to implement the behavior around click events, storing and restoring state info, in addition to notifying observers about data changes.



Old Manaseer House, Al Ain
, Al Ain,

Al Ain, Abu Dhabi City
, Al Ain,

At home Al-jahli, AL-AIN
, Al Ain,

Green Mubazzarah & Jebel Hafeet
, Al Ain,

جبل حفيت & مبزره
, Al Ain,

Mubazara Park
, Al Ain,

Matraz, Rotana Villa Ladies Accomodation
, Al Ain,

جبل حفيت
, Al Ain,

Jabal Hafet
, Al Ain,

9th st., al jahli District, Al Ain
, Al Ain,

Home Sweet Home Al Jahili Al Ain Abu Dhabi, UAE.

Figure 10: Nearby places list

2. Show the Places Picker: The Facebook SDK provides a placePickerFragment class that displays a list of nearby places. This fragment is hosted in the PickerActivity class. This activity launches when the user clicks on a place in the list. The PlacePickerFragment is used if the incoming intent data matches a pre-defined

place picker Uri. Before loading the data, the PlacePickerFragment is configured to specify search criteria like radius, query and maximum results to return.

3. Display the Selected Place: In this step, the place will be displayed when the place picker activity is dismissed [50].

4.4.4 Public Location Badge:

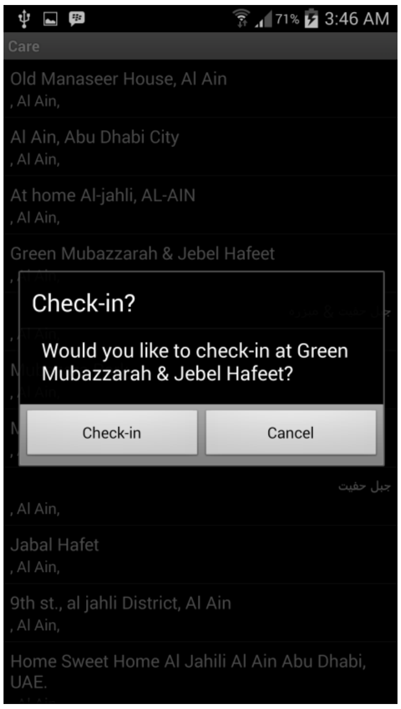
The following steps outline how to publish a story to share the user location with friends. A request will be published by using Request(Session session, String graphPath, Bundle parameters, HttpMethod httpMethod). GraphObject and OpenGraphAction interfaces are used to set up a Graph object representation of the POST parameters. Facebook SDK is used to publish the user location by performing the following steps:

1. Construct a new Request for a currently active session that is an HTTP POST to the me/checkins Graph API path.
2. Set a GraphObject for the Request instance. The GraphObject represents location parameters, like the selected place ID, message and location coordinates.
3. For best practices, the user is asked for publish_actions write permission in context, when the app is about to publish the user location as shown in Figure 11.

The code shown below, publishes the user location to his/her timeline and on the user friends' news feeds. An example of the user shared location is shown in Figure 12.

```
public void onClick(DialogInterface dialog, int which) {
Bundle params = new Bundle();
params.putString("place", placeID);
params.putString("message", message);
params.putString("coordinates", location.toString());
```

Utility.*mAsyncRunner*.~~request~~("me/checkins",params,"POST",**new** placesCheckInListener(), **null**); }



3:46 AM 71%

Care

Old Manaseer House, Al Ain
, Al Ain,

Al Ain, Abu Dhabi City
, Al Ain,

At home Al-jahli, AL-AIN
, Al Ain,

Green Mubazzarah & Jebel Hafeet

Check-in?

Would you like to check-in at Green Mubazzarah & Jebel Hafeet?

Check-in | Cancel

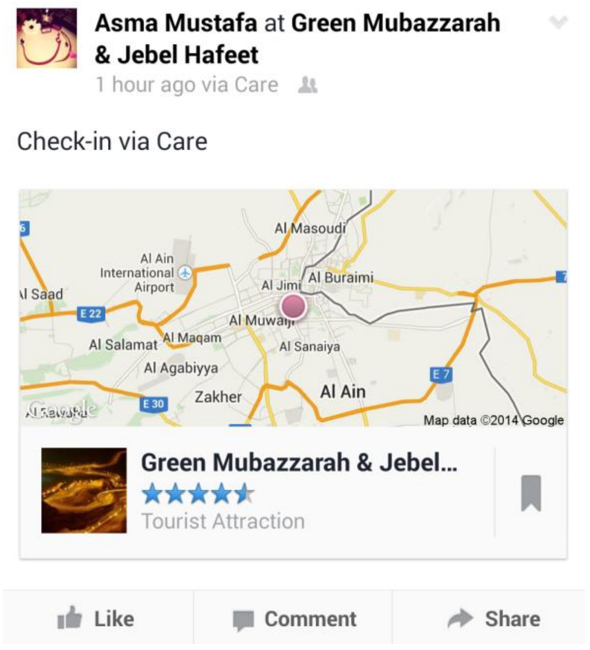
جبل حفيت
, Al Ain,

Jabal Hafet
, Al Ain,

9th st., al jahli District, Al Ain
, Al Ain,

Home Sweet Home Al Jahili Al Ain Abu Dhabi, UAE.

Figure 11: Publish_actions request



Asma Mustafa at **Green Mubazzarah & Jebel Hafeet**
1 hour ago via Care

Check-in via Care

Green Mubazzarah & Jebel...
★★★★☆
Tourist Attraction

Like | Comment | Share

Figure 12: User shared location

4.4.5 Show Nearby Contacts

To show nearby friends, Facebook Query Language (FQL) is used. FQL enables a SQL-style interface to query the data exposed by the Graph API. The steps are described below, that display nearby friends as shown in Figure 13.

1. Issue a HTTP GET request to /fql?q=query where query is a JSON-encoded dictionary of queries. The following code uses FQL to get the friends details and location according to their last check-in:

```
Bundle params = new Bundle();
params.putString("method", "fql.query");
params.putString("query", "SELECT author_uid,timestamp,coords,checkin_id FROM checkin WHERE author_uid IN (SELECT uid2 FROM friend WHERE uid1 = me())  ");
String response = Utility.mFacebook.~~request~~(params);
response = "{\"data\":" + response + "}";
```

2. Store the friends’ details and locations from JSONObject into the following arrays latitude, longitude, author_uid_array, timestamp and checkin_id. After that, the distance between the user and each one of his/her friends is calculated and stored in distances array.

The following code uses the response for the FQL query from the previous step and extracts the friends’ details and locations from the response, and then it stores them in arrays. After that, the distance is calculated and stored in distances array.

```
JSONObject json = Util.parseJson( response );
JSONArray data = json.getJSONArray( "data" );
JSONObject coords;
Long author_uid=(long)0;
```

```
for ( int i = 0, size = data.length(); i < size; i++ ){
JSONObject friend = data.getJSONObject( i );
if(author_uid!=friend.getLong("author_uid"))
{
coords = data.getJSONObject( i ).getJSONObject("coords");
latitude[counter] = coords.getDouble( "latitude" );
longitude[counter] = coords.getDouble( "longitude" );
author_uid = friend.getLong("author_uid");
author_uid_array[counter]=friend.getLong("author_uid");
timestamp[counter] = friend.getString( "timestamp" );
checkin_id[counter]=friend.getLong("checkin_id");
loc. distanceBetween (loc.getLatitude(), loc.getLongitude(),latitude[counter],
longitude[counter], results);
distances[counter]=results[0];
counter++;}}
```

3. Find out nearby friends by comparing distance between the user and each one of his/her friends with a predefined distance, then store nearby friends' name in an array.

The following code uses the distances array from the previous step to compare the distance between the user and each one of his/her friends with a predefined distance-threshold in order to determine nearby friends. Additionally, the following code uses FQL to get nearby friends academics history details to be used in clustering.

```
for ( int i = 0, size = distances.length; i < size && distances[i]!=0 ; i++ ){
if( distances[i]<(float)11500)
{
```

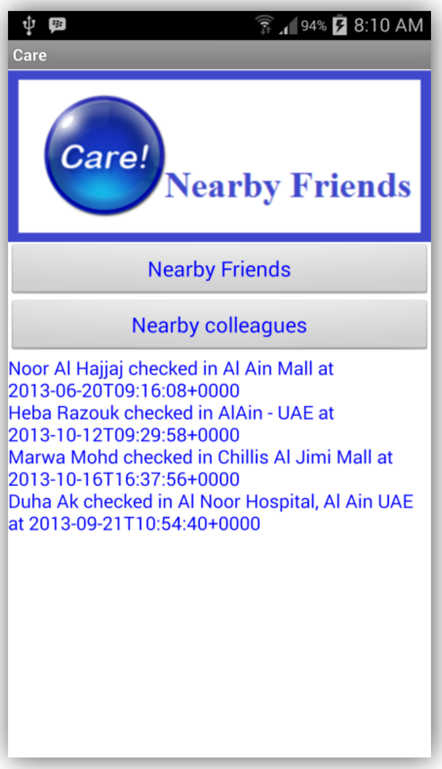
```
neededIndex[counter3]=i;
Nearby_friend_id[counter3]= author_uid_array[i] ;
counter3++;
}}
for ( int i = 0, size = counter3; i < size ; i++ ){
if(i!= size-1)
{
query+= "uid="+Nearby_friend_id[i]+" or ";
}
else
{
query+= "uid="+Nearby_friend_id[i];
}}
Bundle params2 = new Bundle();
params2.putString("method", "fql.query");
params2.putString("query", "SELECT uid,name,education FROM user WHERE "+query);
String response2 = Utility.mFacebook.~~request~~(params2);
response2 = "{\"data\":" + response2 + "}";
JSONObject json2 = Util.parseJson( response2 );
data2 = json2.getJSONArray( "data" );
for ( int i = 0, size2 = data2.length(); i <size2; i++ ){
JSONObject friend2 = data2.getJSONObject( i );
Nearby_friend_Name[i]= friend2.getString("name");}
```

4. Display nearby contact names, their exact location and when they checked in. For example: Dr. Noor Checked in Al-Ain Hospital at 2014-02-10 T09:16

In the following code, when user presses nearby friends’ button, a list of nearby friends is displayed with their location and when they checked in.

```
Button mGetNearbyFriends = (Button) findViewById(R.id.get_nearby_friends);
mGetNearbyFriends.setOnClickListener(new View.OnClickListener() {
public void onClick(View v) {
try{
TextView friends_Locations = (TextView) findViewById(R.id.friends_Locations);
friends_Locations.setText("" );
String jsonUser=null;
for ( int i = 0, size = counter3; i < size ; i++ ){
jsonUser= Utility.mFacebook.~~request~~(""+checkin_id[neededIndex[i]]);
obj = Util.parseJson(jsonUser);
placeName[i]=obj.optJSONObject("place").getString("name");
created_time[i]=obj.getString("created_time");
friends_Locations.append(Nearby_friend_Name[i] +" checked in "+placeName[i] +"
at "+ created_time[i]+"\n");} }
catch (MalformedURLException e) {
e.printStackTrace();}
catch (IOException e) {
e.printStackTrace();}
catch (FacebookError e) {
e.printStackTrace();}
catch (JSONException e) {
```

e.printStackTrace();} }});



Noor Al Hajjaj checked in Al Ain Mall at 2013-06-20T09:16:08+0000
Heba Razouk checked in AlAin - UAE at 2013-10-12T09:29:58+0000
Marwa Mohd checked in Chillis Al Jimi Mall at 2013-10-16T16:37:56+0000
Duha Ak checked in Al Noor Hospital, Al Ain UAE at 2013-09-21T10:54:40+0000

Figure 13: Nearby friends list

4.4.6 Cluster Nearby Contacts

The following steps show, how clustering can be used to group nearby friends.

1. Issue a HTTP GET request, shown below, to /fql?q=query to get user academics history:

Bundle params3 = **new** Bundle();

params3.putString("method", "fql.query");

params3.putString("query", "SELECT name,education_history FROM user WHERE uid =me()");

String response3 = Utility.*mFacebook*.~~request~~(params3);

response3 = "{\"data\":" + response3 + "}";

JSONObject json3 = Util.*parseJson*(response3);

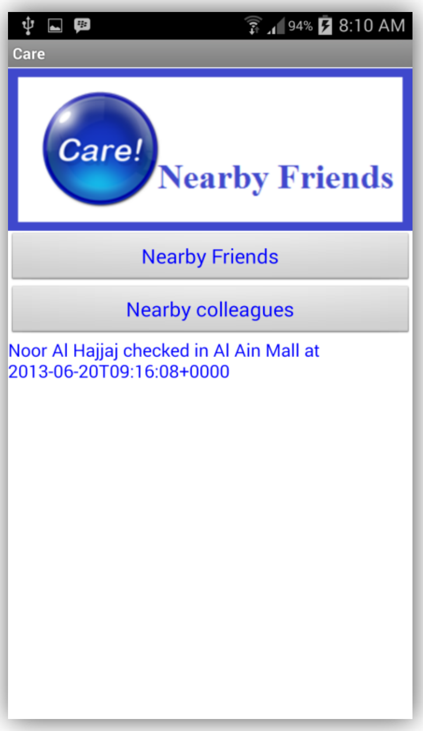
String MyCollege=json3.getJSONArray("data"
).getJSONObject(0).getJSONArray("education_history").getJSONObject(0).getStrin
g("name");

2. Issue a HTTP GET request, shown below, to /fql?q=query to get nearby colleagues after placing the user education history in the FQL query and display the nearby colleagues for the user.

params3.putString("query", "SELECT name,uid,education_history FROM user
WHERE ("+query+ ") AND '"+ MyCollege +"' IN education_history");
response3 = Utility.*mFacebook*.~~request~~(params3);
response3 = "{\"data\":" + response3 + "}";
json3 = Util.*parseJson*(response3);
data3=json3.getJSONArray("data");
for (**int** i = 0, size2 = data3.length(); i <size2; i++)
{
Nearby_college_id[i]=data3.getJSONObject(i).getLong("uid");
}
int counter4=0;
for (**int** i = 0, size = counter3; i < size ; i++){
if(Nearby_friend_id[i].equals(Nearby_college_id[counter4]))
{
friends_Locations.append(Nearby_friend_Name[i]+" checked in "+placeName[i] +"
at "+ created_time[i]+"\n");
counter4++;   } }
}
catch (MalformedURLException e) {

```
e.printStackTrace();
} catch (IOException e) {
e.printStackTrace();
}
catch (FacebookError e) {
e.printStackTrace();
} catch (JSONException e) {
e.printStackTrace();} }  }); }
```

When the user presses a nearby colleagues' button, the user's nearby colleagues will be displayed as shown in Figure 14.



Noor Al Hajjaj checked in Al Ain Mall at
2013-06-20T09:16:08+0000

Figure 14: Nearby colleagues list

4.4.7 Tracking Distance Moved, Calories Burned and Active Time

The following steps show, how tracking distance moved, calories burned and active time are calculated.

1. ‘Care’ tracks the user moved distance by capturing user location coordinates periodically, calculating the distance between each two locations and summing the distances from the time the user presses the start button till the time the user presses the stop button.

The following code stores the user location coordinates periodically till the stop button is pressed.

```
public void onLocationChanged(Location loc) {
dialog.dismiss();
if (loc != null) {
try {
location.put("latitude", new Double(loc.getLatitude()));
location.put("longitude", new Double(loc.getLongitude()));
} catch (JSONException e) {
}
showToast("Location acquired: " + String.valueOf(loc.getLatitude()) + " "
+ String.valueOf(loc.getLongitude()));
lm.removeUpdates(this);
if(counter==0)
{
fetchPlaces();
latitude[counter]=loc.getLatitude();
longitude[counter]=loc.getLongitude();
}
else if (counter==1)
{
```

```
latitude[counter]=loc.getLatitude();
longitude[counter]=loc.getLongitude();
}
else
{
latitude[counter]=loc.getLatitude();
longitude[counter]=loc.getLongitude();
loc.distanceBetween (latitude[counter-1], longitude[counter-1],latitude[counter],
longitude[counter], results);
distances[counter]=results[0];
TotalDistance+=distances[counter];}}
```

2. 'Care' will request the user to enter his/her weight in kilograms in order to calculate the walking burned calories. 'Care' uses the below equation to Calculate the rate of calories burned per pound of body weight [48].

Rate per Pound (Cal/lb-min) = $A+BV+CV^2+KDV^3$ where:

V=Walking Speed (mph) – Limited to a minimum of 1 mph and a maximum of 5 mph

A= 0.0195

B= - 0.00436

C= 0.00245

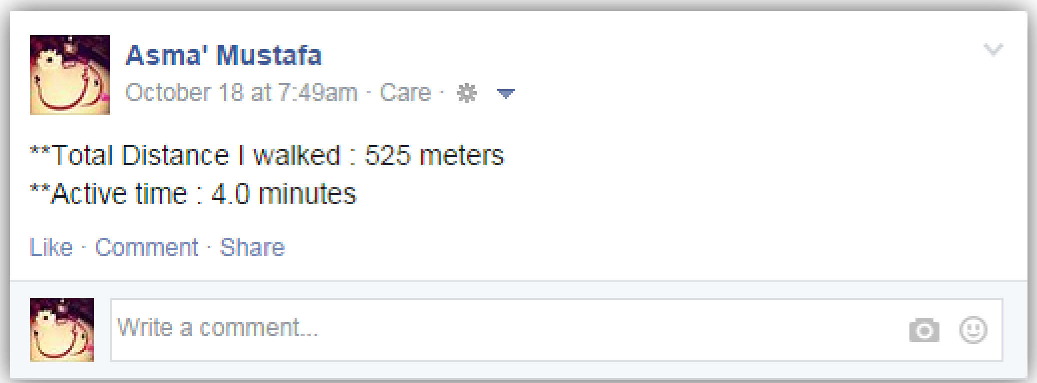
D= $[0.000801(W/154)^{0.425}]/W$

W=Weight (lbs)

K= 0 or 1 (0=Treadmill; 1=Outdoors)

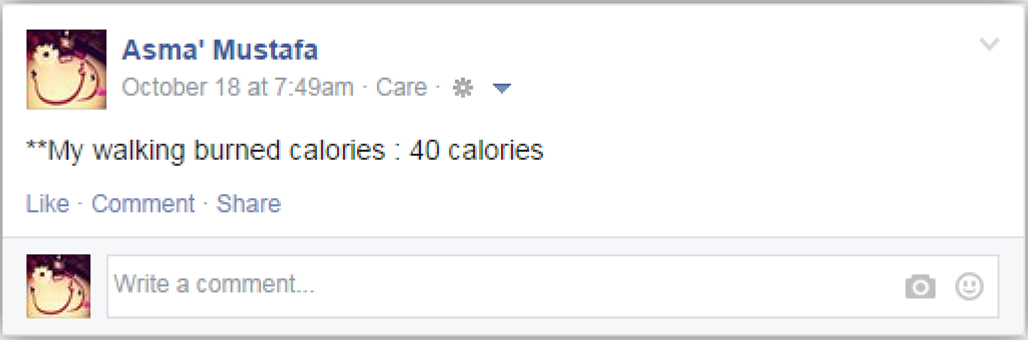
The code, shown below, uses the above equation to calculate the walking burned calories. When the user presses the stop button, 'Care' displays the distance walked, the duration spent during the walk and the walking burned calories as shown in Figure 17.

```
private void getDistanceAndCalories() {
TextView DistanceMoved = (TextView) findViewById(R.id.distance_moved);
TextView Activetime = (TextView) findViewById(R.id.active_time);
TextView WalkingBurnedCalories = (TextView)
findViewById(R.id.burned_calories);
EditText Weight = (EditText) findViewById(R.id.weight);
DistanceMoved.setText("Total Distance you walked : "+ TotalDistance );
activeTime= (counter-1)*30/60; //minutes
Activetime.setText("Active time : "+ activeTime + " minutes \n");
activeTimeInHours=activeTime/60;
totalDistanceInMiles=TotalDistance/(float)1609.344;
A=(float) 0.0195;
B= (float)-0.00436;
C=(float)0.00245;
K= 1;
weightInKgs=(float)Double.parseDouble(Weight.getText().toString());
weightInPounds=weightInKgs*(float)2.20462;
D= (float)((Math.pow(weightInPounds/145, 0.425)*0.000801)/weightInPounds);
V=(totalDistanceInMiles/activeTimeInHours); //Walking Speed (mph) – Limited to a
minimum of 1 mph and a maximum of 5 mph
ratePerPound= (float)(A+(B*V)+(C*Math.pow(V,2))+(K*D*Math.pow(V,3)));
walkingBurnedCalories= ratePerPound*weightInPounds*activeTime;
WalkingBurnedCalories.setText ("Walking burned calories : "+
walkingBurnedCalories + " calories \n");      }
```


Asma' Mustafa
October 18 at 7:49am · Care ·

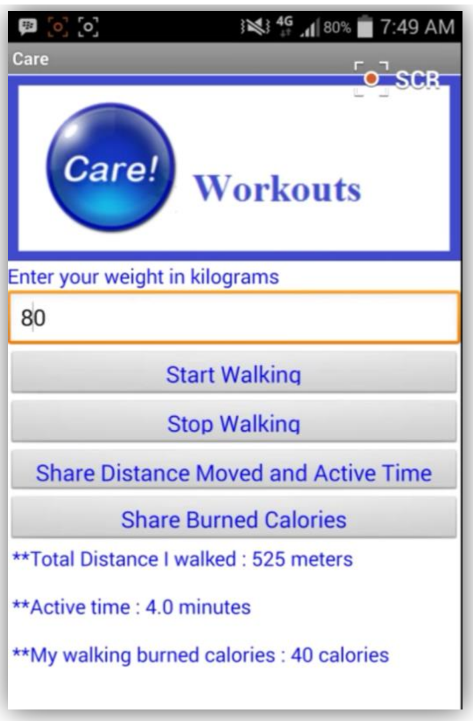
**Total Distance I walked : 525 meters
**Active time : 4.0 minutes

Figure 15: User shared workouts


Asma' Mustafa
October 18 at 7:49am · Care ·

**My walking burned calories : 40 calories

Figure 16: User shared walking burned calories



Enter your weight in kilograms

80

**Total Distance I walked : 525 meters

**Active time : 4.0 minutes

**My walking burned calories : 40 calories

Figure 17: ‘Care’ workouts

When the user presses the share distance moved and active time button, the user shared data will appear on his/her facebook profile as shown in Figure 15. When the user shares his burned calories, his/her facebook post will appear as shown in Figure 16.

4.4.8 Get weather temperature

The following steps show, how weather information is collected.

1. To acquire data from temperature and humidity sensor, an instance of the SensorManager class is created. This instance is used to get the physical sensor.
2. Register a sensor listener in the onResume() method, and start handling incoming sensor data in the onSensorChanged() callback method.
3. Implement onAccuracyChanged() and onSensorChanged() callback methods. The sensor is unregistered when an activity pauses to prevent the sensor from continually sensing data and draining the battery.

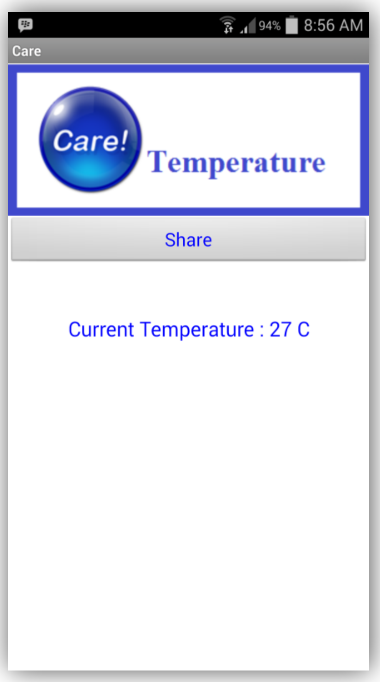
The code shown below uses the temperature and humidity sensor to get the weather, room temperature and then displays them for the user.

```
private SensorManager mSensorManager;
private Sensor mTemperature;
public class Temperature extends Activity implements SensorEventListener {
private SensorManager mSensorManager;
private Sensor mTemperature;
private TextView displayTemperature;
@Override
public final void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
```

```
 // Get an instance of the sensor service, and use that to get an instance of
 // a particular sensor.
mSensorManager = (SensorManager)
getSystemService(Context.SENSOR_SERVICE);
mTemperature =
mSensorManager.getDefaultSensor(Sensor.TYPE_AMBIENT_TEMPERATURE);
displayTemperature = (TextView) findViewById(R.id.friends_Locations);
  }
  @Override
public final void onAccuracyChanged(Sensor sensor, int accuracy) {
  }
  @Override
public final void onSensorChanged(SensorEvent event) {
float Temperature = event.values[0];
displayTemperature.setText("Temperature : "+ Temperature + " C");
 }
  @Override
protected void onResume() {
 // Register a listener for the sensor.
super.onResume();
mSensorManager.registerListener(this, mTemperature,
SensorManager.SENSOR_DELAY_NORMAL);
}
@Override
protected void onPause() {
```

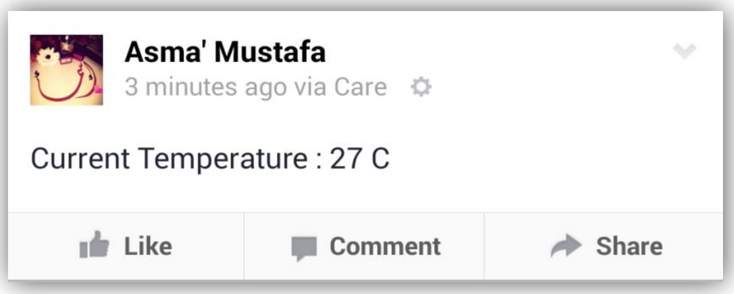
```
// Be sure to unregister the sensor when the activity pauses.
super.onPause();
mSensorManager.unregisterListener(this);
}}
```

When the user presses weather temperature in ‘Care’ main list, he/she will get the temperature as shown in Figure 18 while the temperature shared post will appear on the user profile as shown in Figure 19.



Current Temperature : 27 C

Figure 18: Current temperature



Current Temperature : 27 C

Figure 19: Shared temperature

4.4.9 Share pictures

In order to share pictures, the following code issues a HTTP POST request to share a photo with friends or healthcare professionals.

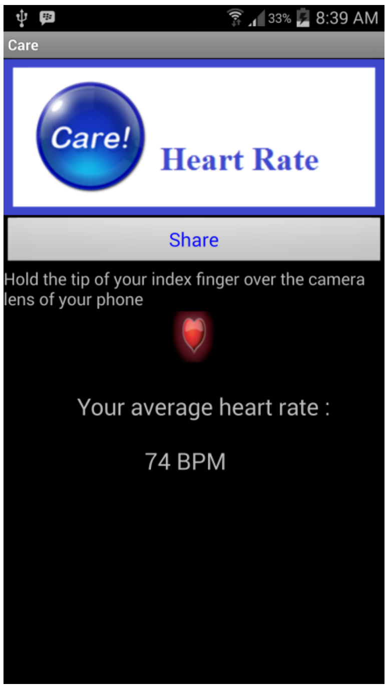
Utility.*mAsyncRunner*.~~request~~("me/photos", params, "POST", **new** PhotoUploadListener(), **null**);

4.4.10 Heart rate monitor

Heart rate monitor uses the camera and its flash to find the user heart rate in beats per minute. The user has to hold the tip of his/her index finger over the camera lens of his/her phone. The application takes between ten to thirty seconds to get an accurate heart rate. Heart rate monitoring is based on using the camera with as little focus as possible. When the user puts his/her finger on the camera lens, it will not be focused. The resulted image will only be shades of light and dark RGB. The code looks at a single channel (red) and tries to find out when the channel goes from light to dark red.

The application uses the PreviewCallback mechanism to capture the last image from the preview frame. Then the YUV420SP data will be processed to get all the red pixel values. YUV420 semi-planer format is the input format for codec like H.264. It is the standard picture format on an Android camera preview. YUV420sp is the default and expected Preview format for onPreviewFrame callback function. Data smoothing in an Integer array is used to figure out the red pixel average value in the image. The heart beat is detected when the average red pixel value in the latest image is greater than the smoothed average. The application collects data during ten seconds, and then adds the beats per minute to an integer array which will be used to smooth the beats per minute data [51]. The resultant heart rate will be displayed for the user as shown in Figure 20.

As in the illustration, the following is the resulting display of user heart rate using 'Care' and a medical device at the same time. Both showed similar results.



Hold the tip of your index finger over the camera lens of your phone

Your average heart rate :

74 BPM

Figure 20: Care heart rate

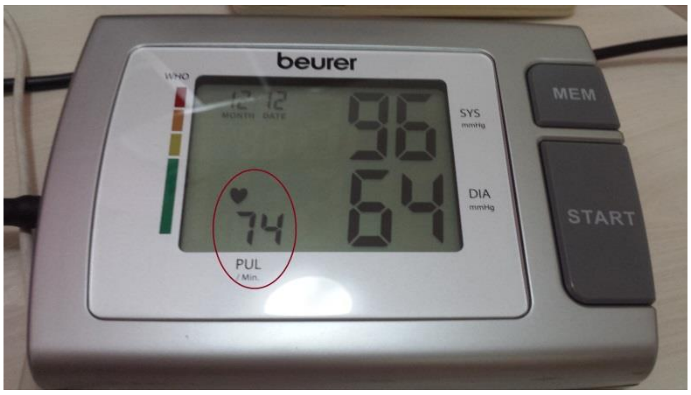
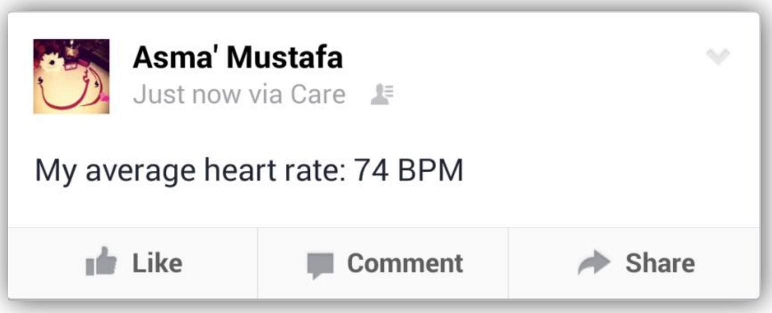


Figure 21: Heart rate using a medical device



My average heart rate: 74 BPM

Figure 22: User shared heart rate

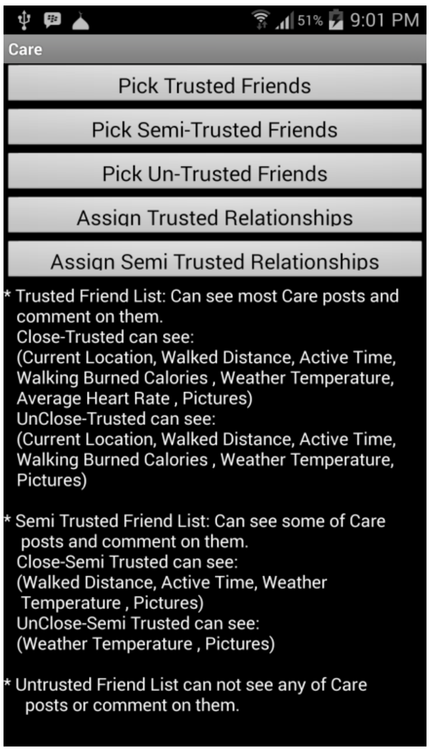
4.5 Implementing Privacy Model

In order to maintain privacy, ‘Care’ provides the user with three roles: trusted, semi-trusted, and un-trusted as shown in Figure 23. Predefined permissions are assigned to each one of the roles. The members of each role will be assigned by the user from his contact list.

When the user presses Pick Trusted Friends, the following steps will occur:

1. Issue a HTTP GET request, shown below, to /fql?q=query to get user friends list as shown in Figure 24:

```
String query = "select name, current_location, uid, pic_square from user where uid in
(select uid2 from friend where uid1=me()) order by name";
Bundle params = new Bundle();
params.putString("method", "fql.query");
params.putString("query", query);Utility.mAsyncRunner.request(null, params, new
TrustedFriendsRequestListener());
```

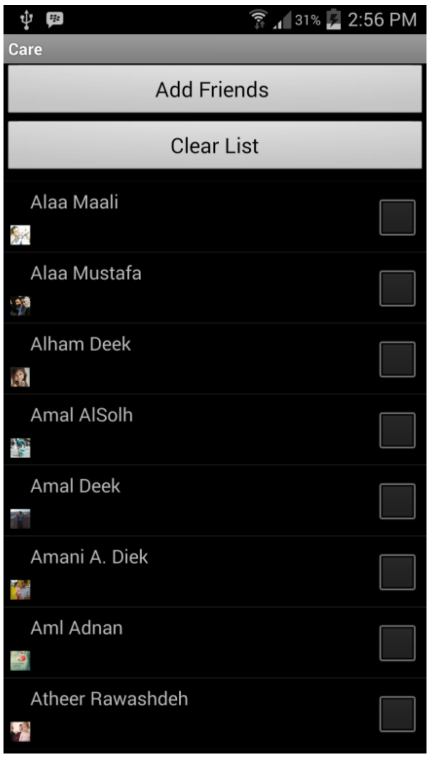

51% 9:01 PM
Care
Pick Trusted Friends
Pick Semi-Trusted Friends
Pick Un-Trusted Friends
Assign Trusted Relationships
Assign Semi Trusted Relationships

* Trusted Friend List: Can see most Care posts and comment on them.
Close-Trusted can see:
(Current Location, Walked Distance, Active Time, Walking Burned Calories , Weather Temperature, Average Heart Rate , Pictures)
UnClose-Trusted can see:
(Current Location, Walked Distance, Active Time, Walking Burned Calories , Weather Temperature, Pictures)

* Semi Trusted Friend List: Can see some of Care posts and comment on them.
Close-Semi Trusted can see:
(Walked Distance, Active Time, Weather Temperature , Pictures)
UnClose-Semi Trusted can see:
(Weather Temperature , Pictures)

* Untrusted Friend List can not see any of Care posts or comment on them.

Figure 23: ‘Care’ privacy



Alaa Maali
Alaa Mustafa
Alham Deek
Amal AlSolh
Amal Deek
Amani A. Diek
Aml Adnan
Atheer Rawashdeh

Figure 24: User friend list

When the user press Add Friends, after choosing the ones he wants to select as trusted, the following steps will occur:

1. Issue a HTTP POST request, shown below, to /fql?q=query to create Trusted Friend list:

```
private void createFriendsList() {
//create empty friendlist
Bundle params = new Bundle();
params.putString("name", "Trusted");
/* make the API call */
new Request(
Utility.mFacebook.~~getSession~~(),
"/me/friendlists",
params,
HttpMethod.POST,
new Request.Callback() {
public void onCompleted(Response response) {
/* handle the result */
try {
addFriendstoList();
} catch (JSONException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}}}
).executeAsync();
}
```

2. Issue a HTTP GET request, shown below, to /fql?q=query to get Trusted Friend list id, then, a POST request to add the selected friends to the list:

```
private void addFriendstoList() throws JSONException {
//Get friendList ID
Bundle param = new Bundle();
param.putString("method", "fql.query");
param.putString("query", "SELECT flid, name FROM friendlist WHERE
owner=me()and name='Trusted'");
String response="";
try {
response = Utility.mFacebook.~~request~~(param);
} catch (MalformedURLException e) {
// TODO Auto-generated catch block
```

```
e.printStackTrace();
} catch (IOException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
response = "{\"data\":" + response + "}";
JSONObject json = null;
try {
json = Util.parseJson( response );
} catch (FacebookError e) {
// TODO Auto-generated catch block
e.printStackTrace();
} catch (JSONException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
String result=json.getJSONArray( "data" ).getJSONObject(0).getString("flid");
Bundle params1 = new Bundle();
params1.putString("members", selectedFriends);
// make the API call */
new Request(
Utility.mFacebook.~~getSession~~(),
result+"/members",
params1,
HttpMethod.POST,
new Request.Callback() {
public void onCompleted(Response response) {
Toast toast = Toast.makeText(FriendsListTrusted.this,selectedFriendsDisplay + " are
added to Trusted Friend List" , Toast.LENGTH_LONG);
toast.show();
}}
).executeAsync();
}
```

When the user presses Clear List, the following steps will occur:

1. Issue a HTTP GET request, shown below, to /fql?q=query to get Trust Friend list id, then, a Delete request to empty it:

```
private void deleteFriendList() {
// TODO Auto-generated method stub
//get Trusted friendlist id
Bundle param = new Bundle();
param.putString("method", "fql.query");
param.putString("query", "SELECT flid, name FROM friendlist WHERE
owner=me()and name='Trusted'");
String response="";
String result="";
try {
response = Utility.mFacebook.~~request~~(param);
} catch (MalformedURLException e) {
// TODO Auto-generated catch block
e.printStackTrace();
```

```
} catch (IOException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
response = "{\"data\":" + response + "}";
JSONObject json = null;
try {
json = Util.parseJson( response );
result=json.getJSONArray( "data" ).getJSONObject(0).getString("flid");
} catch (FacebookError e) {
// TODO Auto-generated catch block
e.printStackTrace();
} catch (JSONException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
// TODO Auto-generated method stub
//delete Trusted friendlist
/* make the API call */
new Request(
Utility.mFacebook.~~getSession~~(),
"/"+result,
null,
HttpMethod.DELETE,
new Request.Callback() {
```

```
public void onCompleted(Response response) {
Toast toast = Toast.makeText(FriendsListTrusted.this,"Trusted Friend list is empty",
Toast.LENGTH_LONG);
toast.show();
/* handle the result */
}}
).executeAsync();}
```

The same process will be executed when the user presses Pick Semi-Trusted Friends or Pick Un-Trusted Friends.

When the user presses Assign Trused Relationships:

The user will get the layout as shown in Figure 25. If he/she presses Close Friends or Un Close Friends, he/she will get a list of the trusted friends only to choose the close or not close friends from them.

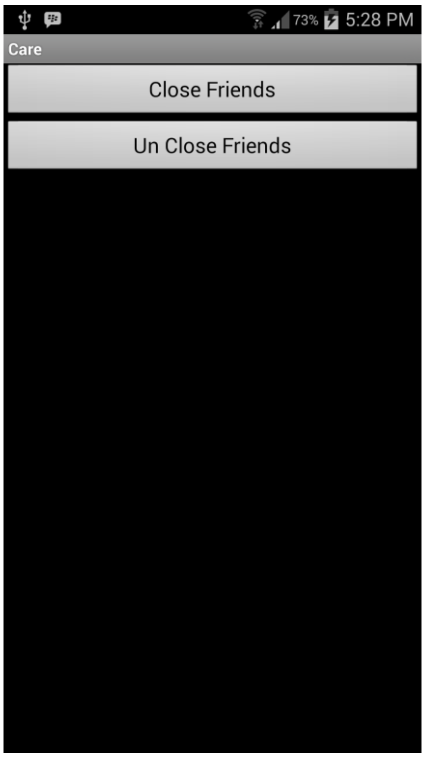
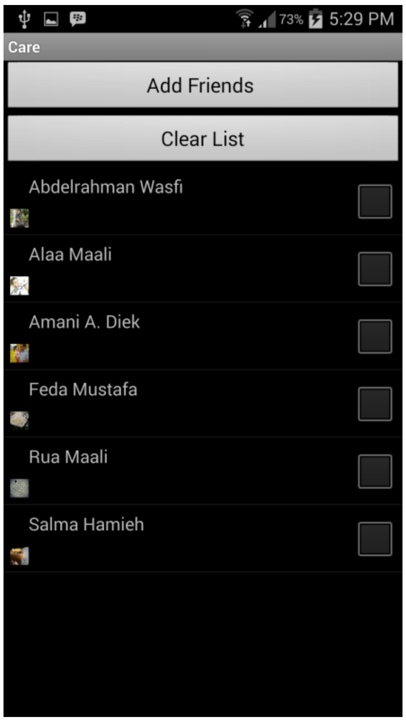


Figure 25: Assigning trusted relationship

When the user presses Close Friends, the following steps will occur:

1. Issue a HTTP GET request, shown below, to /fql?q=query to get user trusted friends only as shown in Figure 26:

```
graph_or_fql = "fql";
String query = "select name, current_location, uid, pic_square from user where uid in (SELECT uid FROM friendlist_member WHERE flid ="+ trustedFriendListId +") order by name";
Bundle params = new Bundle();
params.putString("method", "fql.query");
params.putString("query", query);
Utility.mAsyncRunner.request(null, params,
new assignCloseTrustedRequestListener());
```



Abdelrahman Wasfi

Alaa Maali

Amani A. Diek

Feda Mustafa

Rua Maali

Salma Hamieh

Figure 26: Trusted friend list

Adding Friends to close or un-close will be performed the same way as mentioned in the trusted friend list.

Each time a post will be shared such as ‘distance moved’, ‘calories burned’, ‘active time’, ‘weather temperature’, ‘heart rate or pictures’, the following steps will be performed to specify who can see and comment on the post and who can’t. Below is an example for sharing weather temperature. As we can see trusted and semi trusted friends are allowed to see and comment on the weather post while untrusted friends are denied.

```
private void share() {
String s="";
Bundle params = new Bundle();
params.putString("message", temperature);
JSONObject privacy = new JSONObject();
try {
privacy.put("value", "CUSTOM");
privacy.put("friends", "SOME_FRIENDS");
privacy.put("allow", trustedListId+","+semiTrustedListId);
privacy.put("deny", unTrustedListId);
} catch (JSONException e) {
 // TODO Auto-generated catch block
e.printStackTrace();
}
params.putString("privacy", privacy.toString());
try {
Utility.mFacebook.request("me/feed", params, "POST");
Toast toast;
toast = Toast.makeText(Temperature.this,"Your status has been updated",
Toast.LENGTH_LONG);
toast.show();
} catch (FileNotFoundException e) {
// TODO Auto-generated catch block
e.printStackTrace();
} catch (MalformedURLException e) {
// TODO Auto-generated catch block
e.printStackTrace();
} catch (IOException e) {
// TODO Auto-generated catch block
e.printStackTrace();}
```

As a part of demonstration of these results, preliminary results of this application have been presented at International Conference on Robotics, Automation, and Cybernetics held in Singapore, August 18-19, 2014.

4.6 How to improve 'Care' security

HTTP POST and GET requests are insecure, and all data is transferred in clear text. This leads to a problem because anyone can sniff this data. In order to avoid this problem HTTP traffic should be encrypted by using HTTP + SSL or HTTPS. Anything transferred over HTTPS is encrypted. SSL (Secure Socket Layer) is a protocol layer that exists between the Network Layer and Application Layer. SSL provides a mechanism for encrypting all kinds of traffic such as HTTP traffic.

4.7 Integrate Care with other social networks

There are various ways to integrate Android apps with social networks such as Facebook, Twitter and LinkedIn. One of them is using the SDKs available from Facebook, Twitter etc. which provide the API to share updates. The second is using an embedded browser control and use OAuth for authentication and finally the REST APIs is provided to post the update. These two ways require downloading and using the different APIs or implementing the complete protocol. The third and the best way is to use an open source SDK which is available and easy to use. It provides integrating with several social networks. It is known as SocialAuth Android SDK. It allows posting status updates, getting user profiles from Twitter, Facebook and LinkedIn using the API.

First the developer has to register the developed application with the social provider and get the API keys and secrets. Then he/she must integrate the SDK which contains the java libraries that will perform the heavy lifting of OAuth, as well as the REST calls for the social providers. The SDK provides the developer with multiple ways to show the user interface for selecting the social providers on which the user may want to share updates [53].

4.8 Discussion (Comparative Analysis)

A number of recent applications designed in the context of integrating wireless sensor networks with online social networks can be examined for the purpose of comparison. The existing platform applications such as Google Latitude shares the collected mobile position data of the user among different users, and then it generates proximity alerts when two linked users are within geographical proximity of one another. These applications are limited and target specific service only. In this work, ‘Care’ not only uses the built-in sensors of the user mobile device to get current location coordinates, view user current location, share location, show nearby places, and show nearby friends, but also provides more services such as clustering nearby friends, tracking distance moved, calories burned and active time, getting weather temperature, tracking heart rate and sharing pictures. This helps to improve healthcare awareness and prompt quicker and safe advice from healthcare professional. The data privacy is enforced by two options: The first option is available on all available social network platforms, while the second option is using user assigned roles versus data levels.

Chapter 5

5.1 Conclusion

This chapter represents an overview of the thesis, the main contributions and features, and the future work that can be done for further improvement.

In this thesis, the distinctive features of wireless sensor networks and online social networks, the interconnection of both networks and the privacy concerns were discussed. The framework and implementation of ‘Care’ was presented. A number of sensors were exemplified to build a sensing application that enables members of a social network to share their specific healthcare related information with their contacts in a private manner. The user can see if any of his/her contacts are checked-in nearby.

It was shown that contacts can be clustered based on colleagues, family or any other criterion. The clustering process takes place in the user device. ‘Care’ is a great tool for fitness, weight loss, calorie counting, and so forth, and can facilitate quicker monitoring of, for example, pulse/heart rates of the user through social network by concerned healthcare units. ‘Care’ enables the user to monitor their heart rates and share the data with their doctor. This will help the user to track their fitness in a new level. ‘Care’ encourages users to stay active and exercise which helps people to stay healthy, reduce health risks and lose weight.

Social constraints such as privacy were addressed in two ways: simple, like turning application ON or OFF; and the other by privacy aware data connectivity based on user roles. ‘Care’ privacy model was proposed to ensure individuals privacy. This model uses roles and relationships to enable the user to control his/her information sharing.

Some future work can be performed in order to extend and improve ‘Care’. Some aspects can be improved and more functionality added. Below are some suggestions for further improvement:

- Health support for elders by using sensor information to send alerts if there is abnormal activity pattern. Request for attention can be sent to doctors and nearby friends based on the collected information from sensors such as body position and health measurements.
- Suggesting nearby friends based on common interests.
- Adding more features such as monitoring and tracking the user blood pressure: storing, analyzing and sharing the user blood pressure measurements.

Bibliography:

[1] Kumari, J. ; Patel N. ; Anand S. ; Bhattacharya P.P., "Designing Low Power Wireless Sensor Network: A Survey", *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering* , vol. 2, Issue 9, September 2013, pp. 4447-4456.

[2] Bhattacharyya, D. ; Kim, T.-h. ; Pal, S., "A Comparative Study of Wireless Sensor Networks and Their routing Protocols," *Sensors*, vol. 10, pp. 10506-10523, 2010.

[3] Breslin, J.G. ; Decker, S. ; Hauswirth, M. ; Hynes, G. ; Le Phuoc, D. ; Bojars, U. ; Passant, A. ; Polleres, A. ; Rabsch, C. ; Reynolds, V. , "Integrating Social Networks and Sensor Networks", *W3C Workshop on the Future of Social Networking*, Barcelona, Spain, 15 January 2009.

[4] Aggarwal, C.C. ; Abdelzaher, T.F., " Integrating Sensors and Social Networks". *Social Network Data Analytics*, Springer,2011.

[5] Lewis, F.L., "Wireless Sensor Networks", *Smart Environments: Technologies, Protocols, and Applications* ed. D.J. Cook and S.K. Das, John Wiley, New York, 2004.

[6] Kumar, R. ; Novak, J. ; Tomkins, A., "Structure and evolution of online social networks", *In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM Press, New York, 2006.

[7] Blackstock, M. ; Lea, R. ; Friday, A., "Uniting online social networks with places and things," *in Proceedings of the Second International Workshop on Web of Things*, New York, NY, USA, 2011, pp. 5:1–5:6.

[8] Moshman, J. (2009): *On line Social Networking and NGOs. (Non-profit Social Network Survey Report),* retrieved from http://www.wango.org/resources.aspx?section=news&sub=2009 (18-07-2012).

[9] Breslin, J.G. ; Bojars, U. ; Passant, A. ; Fernández, S. ; Decker, S., "SIOC: Content Exchange and Semantic Interoperability Between Social Networks", *W3C Workshop on the Future of Social Networking*, Barcelona, Spain, 15 January 2009.

[10] "OAuth," Wikipedia. [Online]. Available: http://en.wikipedia.org/wiki/OAuth. [Accessed 1 12 2014].

[11] Owyang, J., ''Integrate Social Networks with your Corporate Website with 'Social Sign On'", September, 2010.

[12] Alcaraz, C. ; Najera, P. ; Lopez, J. ; Roman, R., "Wireless Sensor Networks and the Internet of Things: Do We Need a Complete Integration?", *In 1st International Workshop on the Security of the Internet of Things (SecIoT'10)*, December, 2010.

[13] Sharma S. ; Gupta R. K., "Improved BSP clustering Algorithm for Social Network Analysis", *International journal of grid and Distributed Computing,* Vol. 3, September, 2010.

[14] Mishra, N. ; Schreiber, R. ; Stanton, I. ; Tarjan, R.E., "Clustering Social Networks", *in WAW*, 2007.

[15] Yu, G., "Social Network Analysis Based on BSP Clustering Algorithm", *Communications of the IIMA,* Volume 7, Issue 4, (2007).

[16] Brown, C. ; Nicosia, V. ; Scellato, S. ; Noulas, N. ; Mascolo, C., "Where Online Friends Meet: Social Communities in Location-Based Networks", *ICWSM*, 2012

[17] Geard, N. ; Bullock, S., "Group formation and social evolution - a computational model", *ALIFE* 2008: 197-203

[18] Brauer, S. ; Schmidt, T.C., "Group Formation in eLearning-enabled Online Social Networks", *Proc. of the International Conference Interactive Computer aided Learning* (ICL'12), (Michael E. Auer Ed.), Piscataway, NJ, USA: IEEE Press, Sep. 2012.

[19] Memon, Q.A. ; Khoja, S.A., "Semantic Web for Program Administration", *International Journal of Emerging Technologies in Learning*, Vol. 5, No. 4, 2010.

[20] Moravejosharieh, A. ; Modares, H. ; Salleh, R., "Overview of Mobile IPv6 Security", *Third International Conference on Intelligent Systems, Modelling and Simulation*, 2012, pp. 584-587, DDI: 10.1109/ISMS.2012.9.

[21] Memon, Q.A., "A New Approach to Video Security over Networks", *International Journal of Computer Applications in Technology*, Vol. 25, 2006, pp. 72-83.

[22] Altshuler, Y. ; Elovici, Y. ; Aharony, N. ; Pentland, A., "Security and privacy in social networks." Springer, 2013.

[23] Huber, M. ; Mulazzani, M. ; Weippl, E. ; Kitzler, G. ; Goluch, S., "Exploiting social networking sites for spam," *Proceedings of the 17th ACM conference on Computer and communications security,* ACM, 2010, pp. 693–695.

[24] Fung, B. ; Wang, K. ; Chen, R. ; Yu, P. S., "Privacy-preserving data publishing: A survey of recent developments," *ACM Computing Surveys*, (CSUR), vol. 42, no. 4, p. 14, 2010.

[25] Novak, E. ; Li, Q., “A Survey of Security and Privacy in Online Social Networks”, *College of William and Mary Computer Science Technical Report*, WM-CS-2012-2, April, 2012.

[26] "Google Latitude," Wikipedia. [Online]. Available: http://en.wikipedia.org/wiki/Google_Latitude. [Accessed 13 9 2014].

[27] "Google Latitude: An In-Depth Look," Robert Strohmeyer, 6 2 2009. [Online].Available:http://www.pcworld.com/article/159137/google_latitude_look.html. [Accessed 13 9 2014].

[28] " Application programming interface," wikipedia. [Online]. Available: http://en.wikipedia.org/wiki/Application_programming_interface. [Accessed 13 9 2014].

[29] "Google's Latitude - Are location services a privacy risk?," Toby Stevens, 5 2 2009. [Online]. Available: http://www.computerweekly.com/blogs/the-data-trust-blog/2009/02/googles-latitude---are-locatio.html. [Accessed 13 9 2014].

[30] "Privacy fears over Google tracker," BBC News, 5 2 2009. [Online]. Available: http://news.bbc.co.uk/2/hi/technology/7872026.stm. [Accessed 13 9 2014].

[31] "Google Latitude review,". [Online]. Available: http://cellphonetracker.net/cell-phone-tracker-iphone/google-latitude-review. [Accessed 13 9 2014].

[32] "MacroSense ® Technology Platform," Sense Networks. [Online]. Available: https://www.sensenetworks.com/products/macrosense-technology-platform/. [Accessed 13 9 2014].

[33] "Privacy Principles," Sense Networks. [Online]. Available: https://www.sensenetworks.com/principles/privacy-principles/ .[Accessed 13 9 2014].

[34] "Privacy Policy," Sense Networks. [Online]. Available: https://www.sensenetworks.com/principles/privacy-policy/ .[Accessed 13 9 2014].

[35] "Sense Networks," Richard MacManus, 6 4 2009. [Online]. Available: http://readwrite.com/2009/04/06/sense_networks_citysense.[Accessed 13 9 2014].

[36] “MacroSense® Transforms Location into Behavior," Sense Netowrks. [Online]. Available: https://www.sensenetworks.com/macrosense-transforms-location-into-behavior/.[Accessed 13 9 2014].

[37] “CitySense™," Sense Netowrks. [Online]. Available: hhttps://www.sensenetworks.com/products/macrosense-technology-platform/citysense/.[Accessed 13 9 2014].

[38] Miluzzo, E. ; Lane, N.D. ; Eisenman, S.B. ; Campbell, A.T., “CenceMe - Injecting Sensing Presence into Social Networking Applications (Invited Paper)",

2nd European Conference on Smart Sensing and Context (EuroSSC '07), October 2007

[39] Ganti, R.K. ; Pham, N. ; Ahmadi, H. ; Nangia, S. ; Abdelzaher, T.F., "GreenGPS: A Participatory sensing fuel-efficient maps application", *in Proc. ACM MobiSys*, 2010.

[40] Nath, S. ; Liu, J. ; Zhao, F., "SensorMap for Wide-Area Sensor Webs", in *IEEE Computer*, vol. 40, no. 7, pp. 90–93, IEEE Computer Society Press, Los Alamitos, CA, USA, 2007

[41] Henry, N. ; Bezerianos A. ; Fekete. J-D., "Improving the Readability of Clustered Social Networks using Node Duplication", *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1317-1324, 2008.

[42] Patil, A. ; Liu, J. ; Gao, J., "Predicting Group Stability in Online Social Networks". In *WWW'13, International Conference on World Wide Web,* Rio de Janeiro, Brazil, May 13-17, 2013.

[43] Zhang, C. ; Sun, J. ; Zhu, X. ; Fang, Y., "Privacy and Security for online Social Networks: Challenges and Opportunities" , *IEEE Network* , July/August, 2010, pp 13-18

[44] Malik, A.K. ; Dustdar, S., "Sharing and Privacy-Aware RBAC in Online Social Networks". *The Proceedings of the Third International Conference on Social Computing (IEEE SocialCom 2011),* October 9-11, 2011, MIT, Boston, USA. pp. 1352-1355

[45] Srivastava, A. ; Geethakumari, G., "Measuring Privacy Leaks in Online Social Networks", *Proceedings of the IEEE ICACCI-2013*: *International Symposium on Women in Computing and Informatics (WCI-2013)*, August 22 - 25, Mysore, 2013.

[46] "Obesity in the Gulf: disturbing new survey," Lindsay Carroll, 29 5 2014. [Online]. Available: http://www.thenational.ae/uae/health/obesity-in-the-gulf-disturbing-new-survey. [Accessed 13 9 2014].

[47] "Application Programming Interfaces," Facebook. [Online]. Available: https://developers.facebook.com/docs/reference/apis/. [Accessed 1 2 2014].

[48] Karkanen, K.M., "Walking/running heart rate monitoring system," U.S. Patent 6013009, Jan 11, 2000

[49] "Facebook Login Overview," Facebook. [Online]. Available: https://developers.facebook.com/docs/facebook-login/overview/v2.1. [Accessed 13 9 2014].

[50] "Show Nearby Places,” Facebook. [Online]. Available: https://developers.facebook.com/docs/android/scrumptious/show-nearby-places. [Accessed 13 9 2014].

[51] "android-heart-rate-monitor," Justin Wetherell. [Online]. Available: https://code.google.com/p/android-heart-rate-monitor/. [Accessed 13 9 2014].

[52] Wasfi, A. ; Memon, .A., “On extending the Sensing of Privacy-aware Online Social Networks”, *Proceedings of International Conference on Control, Robotics and Cybernetics*, Singapore, August 18-19, 2014.

[53] "How to integrate Facebook, Twitter, Linkedin in Android Apps," Code Project. [Online]. Available: http://www.codeproject.com/Tips/457153/How-to-integrate-Facebook-Twitter-Linkedin-in-Andr /. [Accessed 1 12 2014].

List of Publications:

[1] Wasfi, A. ; Memon, .A., “On extending the Sensing of Privacy-aware Online Social Networks”, *Proceedings of International Conference on Control, Robotics and Cybernetics*, Singapore, August 18-19, 2014.