# An Architecture for M2M Enabled Social Networks

Ashis Kumar Bhowmik

A Thesis

in

The Department of Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science at

Concordia University

Montreal, Quebec, Canada

December 2014

**CONCORDIA UNIVERSITY**

**SCHOOL OF GRADUATE STUDIES**

This is to certify that the thesis prepared

By: Ashis Kumar Bhowmik

Entitled: "An Architecture for M2M Enabled Social Networks"

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

**Master of Applied Science**

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

Signed by the final examining committee:

_____ Chair
Dr. M. Z. Kabir

_____ Examiner, External
Dr. L. Narayanan (CSE) To the Program

_____ Examiner
Dr. A. Agarwal

_____ Supervisor
Dr. F. Khendek

_____ Supervisor
Dr. R. Glitho (CIISE)

Approved by: _____
Dr. W. E. Lynch, Chair
Department of Electrical and Computer Engineering

_____20_____ _____
Dr. Amir Asif, Dean
Faculty of Engineering and Computer Science

# ABSTRACT


An Architecture for M2M Enabled Social Networks

**Ashis Kumar Bhowmik**

Social Networks (SNs), such as Facebook, Twitter, Google+, are becoming more and more popular nowadays. People are now more connected than before. They share information, pictures, videos and news with their family and friends. However, sharing physical phenomena in SNs is still a manual process done by people themselves. For instance, people would like to share current health status, feelings, thoughts, weather or riding information with friends. The sharing of ambient information automatically in SNs can promote independent living. Moreover, it can enhance the autonomy and confidence of elderly people via continuous monitoring and health support. A set of biometric sensors, for example, placed within a patient body can inform a doctor about patient's health status; hence the doctor can perform a remote diagnosis.

Nowadays people are surrounded by devices like smartphone, sensors, cameras, computers and many other devices known as machines. These devices can automatically collect contextual information from the neighborhood. This thesis proposes an architecture for posting contextual information in SNs to support the automatic sharing of physical phenomena. In the proposed architecture, machines collect the contextual data through an overlay-based gateway to support scalability in terms of number of devices. Considering the resource-constrained devices, the architecture makes use of the Constrained Application Protocol (CoAP), a lightweight standard protocol. An SN processes that data into shareable information and disseminates it as appropriate within the users' Community of Interests (COIs) (e.g., family, friends).

A proof of concept prototype is developed to verify the feasibility of the proposed architecture and its performance has been partially evaluated.

# ACKNOWLEDGEMENTS

I would like to express my heartiest gratitude to my supervisors, Dr. Ferhat Khendek and Dr. Roch Glitho, for their continuous support. This thesis would not have been possible without their inspiration, valuable advices and comments. Their constructive ideas helped me to accomplish this thesis. Their patience and helping attitude kept me motivated throughout my research work.

I would like to give special thanks to Majid Hormati for his inspiration, help and comments. It has been an honor and pleasure to work with the members of the Telecommunication Service Engineering research team.

I am grateful to my supervisors and Concordia University for the financial support. This work also has been partially supported by Natural Sciences and Engineering Research Council of Canada (NSERC) and Ericsson Canada.

Finally, I would like to thank my beloved parents, sisters and friends for trusting and encouraging me throughout my study and my career.

# Table of Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| 3GPP | 3rd Generation Partnership Project |
| AAL | Ambient Assisted Living |
| API | Application Programming Interface |
| BSN | Body Sensor Network |
| COI | Community of Interests |
| CoRE | Constrained RESTful Environment |
| DHT | Distributed Hash Table |
| ETSI | European Telecommunications Standard Institute |
| GB | Gigabyte |
| GPS | Global Positioning System |
| HTML | Hypertext Mark-up Language |
| HTTP | Hypertext Transfer Protocol |
| IETF | Internet Engineering Task Force |
| IP | Internet Protocol |
| JAXB | Java Architecture for XML Binding |
| JSON | Java Script Object Notation |
| M2M | Machine to machine |
| P2P | Peer-to-Peer |
| PC | Personal Computer |
| PDA | Personal Digital Assistant |
| PHP | PHP: Hypertext Preprocessor |

| | |
|---|---|
| REST | Representational State Transfer |
| ROA | Resource Oriented Architecture |
| SN | Social Network |
| SN | Social Network |
| SQL | Structured Query Language |
| TCP | Transmission Control Protocol |
| UDP | User Datagram Protocol |
| URI | Uniform Resource Identifier |
| URL | Uniform Resource Locator |
| WADL | Web Application Description Language |
| Wi-Fi | Wireless Fidelity |
| WiMAX | Worldwide Interoperability for Microwave Access |
| WSN | Wireless Sensor Network |
| WWW | World Wide Web |
| XHTML | Extensible Hypertext Mark-up Language |
| XML | Extensible Markup Language |

# Chapter 1: Introduction

This chapter starts by presenting the research domain. It then states and discusses the motivations, followed by the thesis contributions. Finally, it presents the thesis organization.

## 1.1 Research Domain

Machine-to-Machine (M2M) [1] refers to the communication between different devices (e.g. smartphone, IP camera, personal health device, and computers) without or with a limited user intervention. The purpose of the M2M technology is to share information between autonomous electronic systems. This promising technology enables new possibilities of smart services and applications. M2M communication has its application in the smart power grid, home networking, e-health and vehicular networks [2]. A smart power grid application allows users to monitor and control their power consumptions and costs [3]. Another example of M2M is E-health applications like remote patient monitoring. E-health applications are used to monitor and collect vital signs like blood pressure or heart rate of the patients [4].

Social Networks (SNs), like Facebook [5], Twitter [6], Google+ [7], are gaining popularity from the early period of this century. SNs help us to remain connected with our family and friends. They allow us to create profiles with personal details and connect to other people. An SN enables the sharing of information among its users and thereby elevate the level of social interactions within the community [8]. This sharing is done for different purposes. It can be general social interactions as in Facebook, news sharing among the users as in Reddit [9], or professional network building and recruiting as in LinkedIn [10]. According to an online

statistics [11], 73% of the online users use social networking sites in recent days. Moreover, 42% of those SNs users use multiple SNs; whereas these numbers are rapidly increasing.

## 1.2 Problem Statement & Motivation

SNs are being used for more than social exchanges and connections. For instance, SNs are being exploited in the E-health domain [12]. Elderly and disabled people can be assisted by Ambient Assisted Living (AAL) that can improve their autonomy and confidence [13]. Current SNs depend on users who update their information manually. These include textual contents, images, videos, link to web resources or news. These all are user-initiated processes. For instance, Facebook allows users to share "What's on your mind" [5].

Devices can produce contents automatically [14]. For example, a Global Positioning System (GPS) sensor of a mobile can update the location of a user in an SN, or a body sensor placed with an elderly person can post information about his health condition to his Community of Interests (COIs) (e.g., family, friends). LiveLens [15] is a video streaming platform that uses Google Glass [16] to enable live streaming and share with friends. Additionally, an online social gaming platform can enhance the virtual reality by using the automated feature of a sensor device. A sports trainer can monitor the routine activity and health statuses of his athletes [17].

Researchers predict that about 50 billion devices will be connected by the end of 2020 [18]. These devices will certainly generate a huge amount of data. Certain people might be interested in various types of these auto-generated information.

In this thesis, we propose an architecture for enabling M2M communication in SNs to handle these M2M devices, collect the data and share them among the SN users. Smartphones are an integral part of our daily lives. They have the processing capability to realize different

services. A smartphone can be used as a gateway to connect M2M devices with SNs. We propose a detailed architecture for this gateway. The gateway is capable of managing these M2M devices in a scalable manner. Additionally, it controls the data flow by filtering to provide a better user experience. This thesis also proposes a detailed architecture for the SN to map M2M devices with the SNs users and analyze the M2M data to find potential events. The SN relies on the adaptation of that data into useful information to share them among the users. Though some data might be sensitive, privacy and security are not addressed in this research work.

## 1.3 Thesis Contributions

This thesis aims to propose a novel Constrained Application Protocol (CoAP) [19] based overlay gateway architecture and OpenSocial API [20] based SN architecture to feature the automatic sharing of physical phenomena. The main contributions of this thesis are as follows:

- A derived set of requirements for enabling the M2M communication in the SN to automate the information sharing. Two sets of requirements have been derived. First, a set of general requirements for enabling the sharing of the M2M related posts in the SN. These requirements led us to the second set of requirements for the M2M gateway to support a large number of M2M devices.
- Review of existing research works relevant to automatic information sharing in the SN and works relevant to overlay based gateway.
- Evaluation of the related works with respect to the derived set of the requirements for the general architecture and the gateway to enable the M2M sharing in the SN.
- A novel architecture of the M2M enabled SN based on an overlay gateway for the M2M and M2M extension of OpenSocial API for the SN. The architecture includes a set of CoAP based overlay gateway components and a set of SN server components.

- A proof of concept prototype of the proposed M2M enabled SN and its partial performance evaluation.

## 1.4 Thesis Organization

The rest of the thesis is organized as follows:

- Chapter 2 provides the background information on M2M, SNs, Point-to-Point (P2P) [21] overlay network, Representational State Transfer (REST) [22] & RESTful web services [23] and CoAP, which are required to understand the concepts and ideas related to this thesis.
- Chapter 3 introduces the motivating scenarios, the requirements and the related works relevant to the sharing of M2M information in the SN. It then evaluates the related works with respect to the derived requirements.
- Chapter 4 elaborates on the architecture of the M2M enabled SN. First, it describes the overall architecture of the system. Then, it discusses the detailed architecture of the SN Server and the M2M gateway. It also describes the components of the SN Server and the M2M gateway and their functionalities.
- Chapter 5 describes the software architecture of the SN Server and the gateway. Additionally, it presents a proof of concept prototype. It concludes with a partial performance evaluation of the architecture.
- Chapter 6 concludes the thesis by summarizing its contributions and proposes some future research directions.

# Chapter 2: Background Information on SNs, M2M, P2P Overlay, REST and CoAP

This chapter presents the relevant background information of the research domain. It discusses Social Networks (SNs) [8], Machine-to-Machine Communications (M2M) [1], Representational State Transfer (REST) [22] & RESTful web-services [23], Point-to-Point (P2P) overlay networks [21] and Constrained Application Protocol (CoAP) [19].

## 2.1 Social Networks

Social Networks (SNs) are a popular medium to connect with friends and acquaintances nowadays. This section presents the general definition of an SN and the history of various SN sites. It also presents the SN structure, some existing SNs and their applications.

### 2.1.1 Definition of Social Networks

SNs are known as the web-based internet services that enables the human relationship formation on the internet among the people with common interests [24]. It allows a user to connect with the other users and share information within the network. The SN helps to build a strong social relationship amongst its users. All the SNs let users perform at least three common functionalities [8], which are as follows:

1) Users are able to construct a public or a semi-public profile within the system
2) Users are able to build and manage a list of shared connection with other users (friends) bounded within the same system
3) Users can share information with their connections by posting and getting updates from the connected users

Though, some SNs help to maintain existing social relationship, most of them allow connecting and sharing with strangers with common interests. An SN allows users to visit their friend's connection. This feature makes it possible to renew old connections and convert an offline connection to an online one, which might not be possible otherwise. Additionally, recent improvements allow SNs more sophisticated features like photo sharing, video sharing, text chat, voice and video chat.

### 2.1.2 History of Social Networks

Users could create profiles on the dating sites and community sites even before the first recognized SN, SixDegrees.com [25]. A user profile contains user's personal information and interests. SixDegrees.com was launched in 1997. It also introduced the friend-list for the users. A friend-list is a list of user's connections. A feature to browse a friend's friend-list was added in SixDegrees.com in 1998. AIM [26] and ICQ [27] also had buddy list, which is similar to a friend-list of an SN, but the list was not visible to other buddies (users). SixDegrees.com helped people to connect and send messages to connected users. This SN could not sustain, and they closed their services in 2000, in spite of being a popular SN to the millions of users. Users had very little to do in SixDegree.com after accepting friend requests [8].

AsianAvenue [28], BlackPlanet [29] & MiGente [30] were launched between 1997 and 2001. These websites allowed people to create personal, professional and dating profiles. The users could connect with the other users without their approval. LiveJournal [31], launched in 1999, allowed unidirectional connection amongst the users. LinkedIn [10], Ryze.com [32], Tribe.net [33] provide services for business and professional network. Only LinkedIn became popular among professionals and is a success. Friendster [34] was launched in 2002 as a dating site to compete with Match.com [35]. It soon became a famous SN and started attracting people,

although it only allowed friends-of-friends to connect with each other. Later, it faced technical difficulties to handle the user-base and failed to continue its service. Flickr [36], Last.FM [37] and YouTube [38] emerged as media sharing SN. Flickr allows users to share photos whereas Last.FM allows users to share music. YouTube started their service in 2005 for video sharing. MySpace [39] was launched in 2003 and attracted Friendster users as Friendster had already failed. Twitter, Google+ and Facebook are recent SNs. Twitter is a microblogging service. It became popular by attracting celebrities and their fans. Facebook is the most dominant SN, serving 71% of the total SN users [11]. An online survey shows that 42% of the SN users use multiple SN. Figure 2.1 shows the timeline of various SNs from 1997 to 2006. More details about the SNs history can be found in [8].

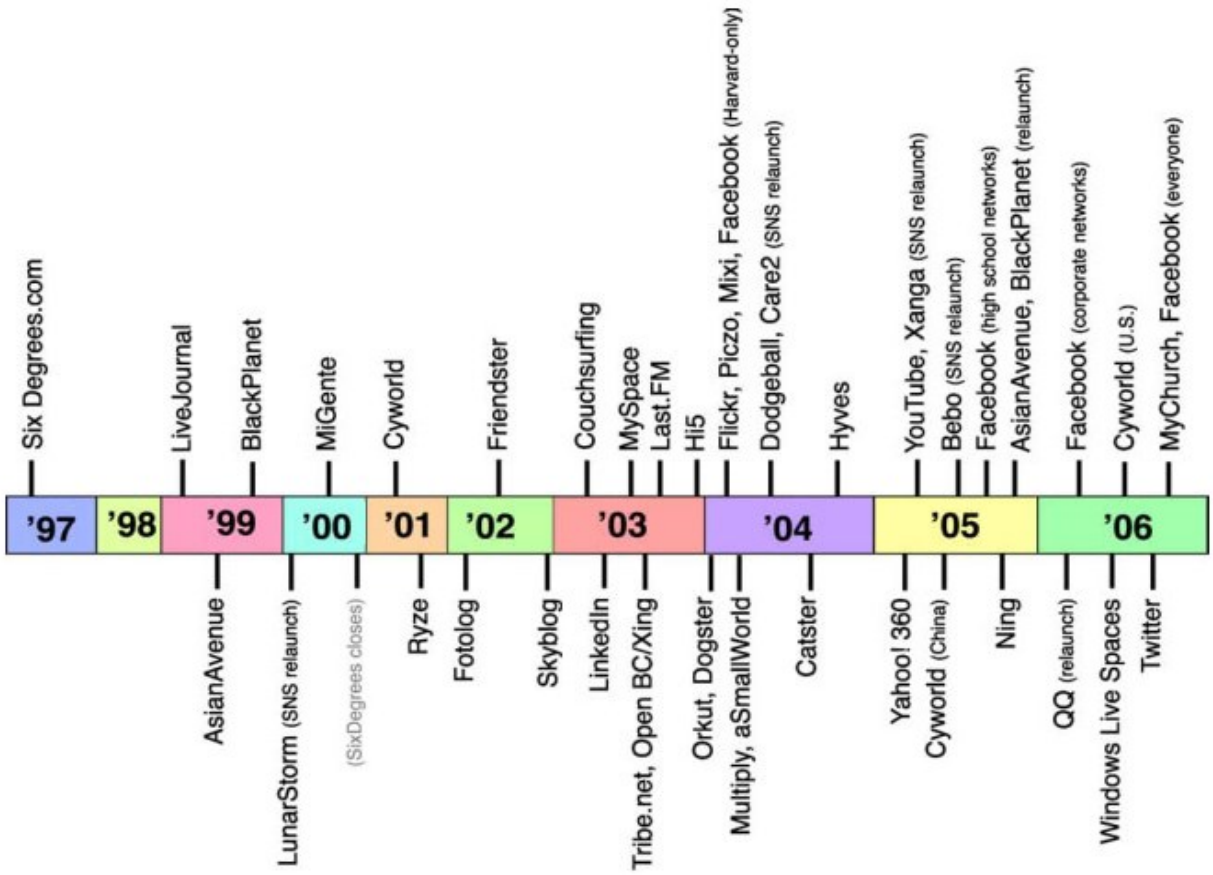


**Figure 2.1 – The timeline of the launch dates of major SNs in** [8]

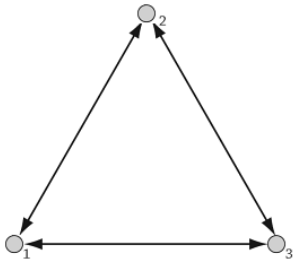


**Figure 2.2 – The sociogram of mutually related three nodes in** [40]

### 2.1.3 The Social Network Structure & Social Graph

A network is defined as a set of connections among some nodes. In the SN perspective, SN users are the nodes in the network. The relationship among two users is the connection between them [40]. This relationship can be a simple relationship, a directed relationship, or a symmetric relationship. A symmetric relationship is a mutually directed relationship. Figure 2.2 shows a mutually related graph, called sociogram, of three nodes. The term "sociogram" was invented by Jacob Moreno, a key founder of modern network studies, in 1953 [40].

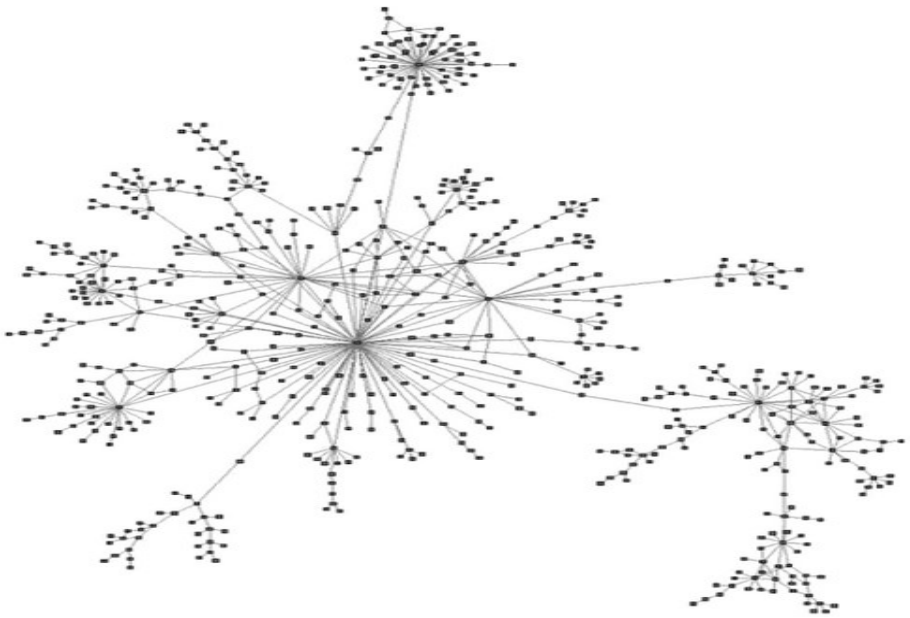

**Figure 2.3 – The social graph of Krebs and his followers in Twitter in** [40]

The SN structure is different from a user perspective and the SN perspective. A user's SN structure only includes the nodes that are connected to the user. This structure is called the social graph. It is different for every user of an SN and only considers his/her connections. Figure 2.3 shows a social graph of an SN user, Valdis Krebs, and his followers in the Twitter. The central node in the graph represents Krebs himself. Other nodes in the social graph are his followers and the followers of his followers (friends-of-friends). On the other hand, the overall SN structure includes all the nodes within an SN. Figure 2.4 shows the SN structure of the members of a Karate club. In this SN structure, the nodes are representing the members and the connections are representing the established friendship between them.

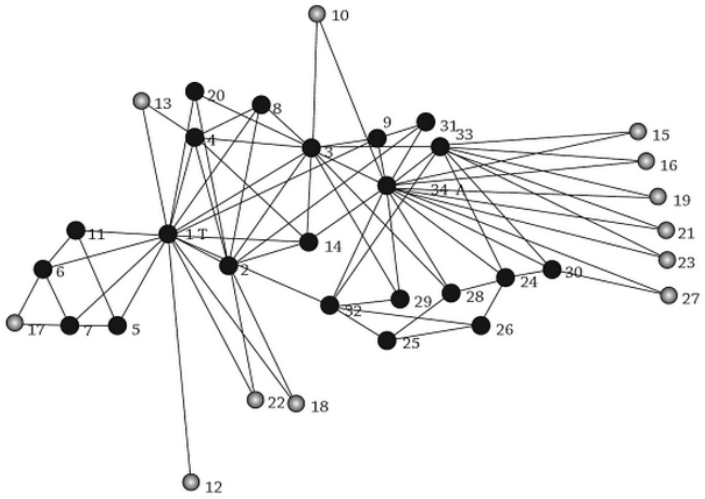


**Figure 2.4 – The SN structure of a karate club from SN perspective in** [40]

### 2.1.4 **Different types of Social Networks and the Standards**

The scope of the SNs is very wide on the internet world. New SNs are arriving with special features every day. These SNs are based on different types of user's interest and target the users or community accordingly. For example, a network of drivers shares road traffic and road conditions. A network of patients and doctors shares health issues, experiences, drug history and side-effects among the other users within the network.

More than a million people are killed in the world every year in traffic accidents. In 2007, around 4000 people were killed, and 2.5 million were injured in US [41]. Causes for these accidents include but no limited to bad road conditions, mad drivers and traffic jam. An SN for drivers enables sharing of road conditions and traffic information with the other drivers [42] to improve this condition.

E-Health based SNs are designed considering patients and health practitioners. They enable connecting them with each other with certain privacy. CureTogther [43] and PatientsLikeMe [44] are the SNs that promotes telehealth and telemedicine opportunities [45]. Third parties also develop applications for existing SNs by accessing information through a set of available APIs [46], especially for social games or adding a new feature to an existing SN.

Google launched the OpenSocial [20] in 2007 as a process of SN standardization. OpenSocial proposes a set of APIs for the SN application development. Google+, Hi5 [47], LinkedIn, Myspace and the many other SNs support the OpenSocial [48]. OpenSocial supports a great range of functionalities, which includes Profiles, Relationships, Activities, Shared applications, Authentication and Authorization. The primary goal of the OpenSocial is to provide a framework for developers to develop applications, which are interoperable throughout various OpenSocial supported SNs.

OpenSocial enables the client programs (third party applications) to access the SN resources through REST APIs [49]. The REST allows data exchange in various formats like JavaScript Object Notation (JSON) [50], Extensible Markup Language (XML) [51] and ATOM [52]. OpenSocial supported SNs use the OAuth [53] for authentication and authorization of the

access requests from a third party SN applications. We discuss REST and RESTful web services later in this chapter.

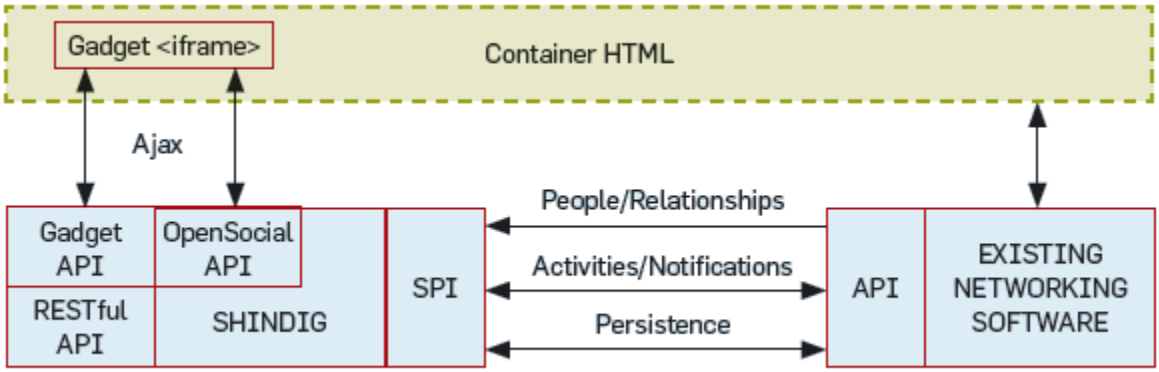


**Figure 2.5 – The Shindig reference architecture in** [49]

Apache Shindig [54] is an open-source reference implementation of the OpenSocial. It provides the code to render gadgets, proxy requests and handle the REST and the RPC requests. A gadget is an XML document that contains HTML, JavaScript and metadata. The gadgets are the base of OpenSocial based applications. Shindig allows the application developers to build and test OpenSocial based applications in a short period. Shindig implementation currently exists in Java and PHP languages. The details about the Shindig implementation is available in [54]. Shindig communicates with existing SNs through the APIs. The SNs access Shindig through the Service Provider Interface (SPI). Shindig implements a gadget container, an OpenSocial container, and a gadget server. The gadget server transforms the XML gadgets into HTML and JavaScript, which is included within an HTML file. Figure 2.5 depicts the Shindig reference architecture.

## 2.2 Machine-to-Machine Communication

This section introduces Machine-to-Machine (M2M) communication. It also discusses the M2M system architecture. Next it shows the relationship between the M2M and REST. It ends with examples of M2M applications.

### 2.2.1 Definition of Machine-to-Machine Communication

M2M is a promising technology that enables millions of smart objects to communicate with each other. It enables the communication between different devices (e.g., smart phone, IP camera, personal health devices, sensors, actuators) with little or no human intervention to share information. This technology started developing from pretty old days. The use of the M2M was found in Supervisory Control and Data Acquisition systems in 1980s [55].

### 2.2.2 M2M System Architecture

A high level M2M system architecture is depicted in Figure 2.6. It consists of two domains, the device domain, and the network and applications domain. The device domain includes the M2M devices, an M2M gateway and an M2M area network. Some M2M devices are capable of hosting M2M service capabilities and applications. An M2M gateway hosts the M2M service capability and applications, where the M2M devices are not capable of doing so. An M2M area network is responsible for providing the connectivity between the M2M devices and the M2M gateway. The transport network, M2M applications and the management functions are in the network and applications domain. M2M service capabilities are used for developing various M2M applications. The access network connects the device domain and the core network. The users interact with the M2M applications through the user interfaces. The European Telecommunications Standard Institute (ETSI) [56], Internet Engineering Task Force

(IETF) [57], 3rd Generation Partnership Project (3GPP) [58] and some other key telecommunications and networking standardization bodies are working on the standardization process of the M2M communication [55].

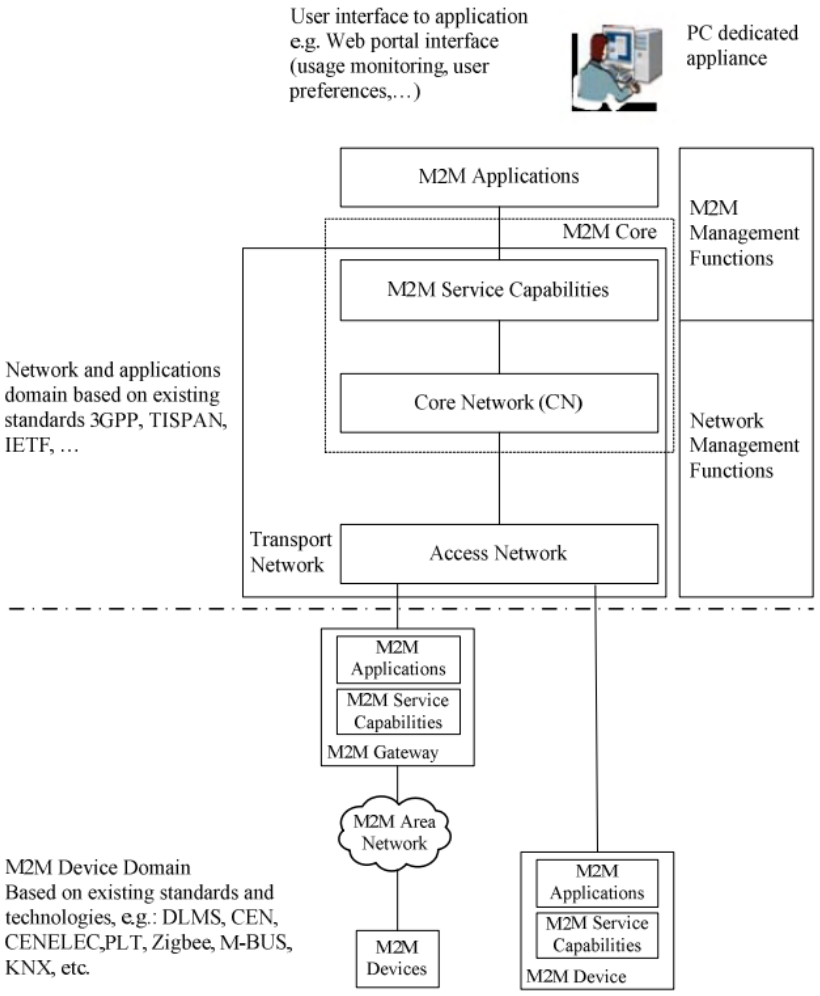


**Figure 2.6 – The ETSI M2M system architecture in** [55]

REST [22] is chosen for the M2M system architecture by the ETSI, for communications between the M2M devices and the M2M gateway as well as between the M2M gateway and the M2M applications [59]. REST is discussed in the Section 2.4 in more details.

### 2.2.3 M2M Applications

There is a great possibility for a variety of applications to emerge in the future with the daily increase of smart objects. These objects will communicate to serve humanity to provide a smart way of living. Domains that can be benefited from M2M communication are logistics, healthcare, smart grid and surveillance [59]. Smart grid applications are used for saving resource consumptions by dynamically matching the demand with the supply. Some applications of smart grid are smart metering, wide area monitoring control, and equipment diagnostics [1].

In vehicular scenarios, M2M applications are about the safety and security, information and navigation, diagnostics, or entertainment [1]. For example, a voice call can be initiated to report a crash, by sensing through various onboard crash sensors. Information or navigation services provide the location and navigation information to the users. As an example, the Google Map provides road maps and street views with the GPS navigation capability on the smartphones.

M2M applications in the healthcare can improve the quality of patient care and reduce the healthcare costs. Remote patient monitoring is one of the key application in the field of M2M healthcare. Biosensors, placed with the body of a patient, can connect to a hospital's network to provide more accurate and timely report to the health specialists. In emergency conditions, physicians can prepare the treatment before the patients arrival [1] by accessing these remotely available medical data.

## 2.3 Peer-to-Peer Overlay

A Peer to peer (P2P) overlay network is a distributed application architecture, where each peer can act as a client and/or a server to perform a particular task. In the traditional client/server

architecture, the servers are computation and network bottleneck. The traditional client/server architecture faces issues like scalability, efficiency and availability. The P2P is proposed to address these issues [60].

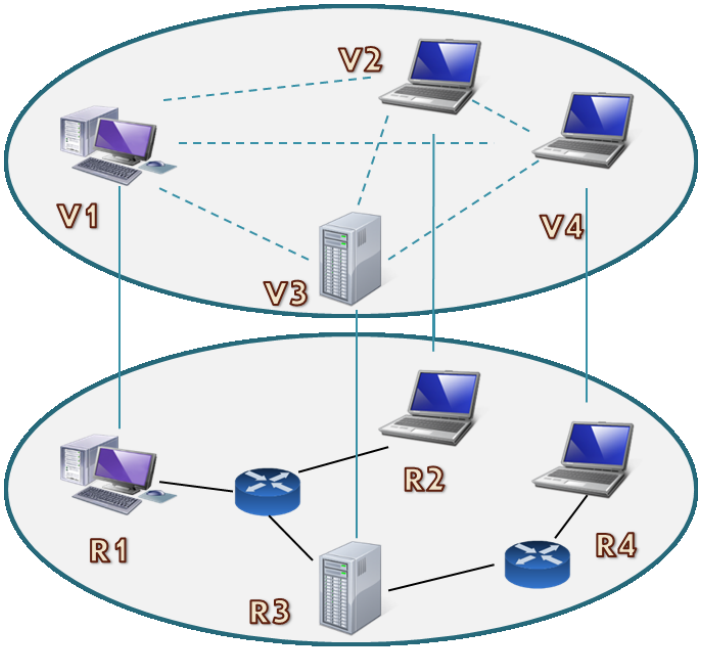


**Figure 2.7 – A P2P overlay network on top of a real network**

P2P relies on a network of peers. These peers communicate via a P2P network to offer storage and processing that is required by the applications. A peer on a P2P overlay network can join or leave anytime. The P2P overlay network should be able to self-organize, handle fault tolerance for departures of peers. The type of network is called an overlay as they are virtual nodes that are built on top of existing network. Figure 2.7 depicts a P2P overlay network. A P2P overlay network has the following characteristics:

- A P2P overlay network topology may differ from the topology of the real network
- It may use a different protocol than the protocol of the real network.

- It can either be embedded in an application (e.g. Skype) or an infrastructure that can be used by the other applications to build a P2P network on top of it (e.g. Chord [60]).

### 2.3.1 Peer-to-Peer Overlay Network Architecture

Figure 2.8 shows the abstract architecture of a P2P overlay. It consists of five layers. The Network Communications layer is the real transport network. The Overlay Nodes Management layer deals the routing, peer discovery and lookup. The Feature Management layer makes the security, reliability and fault resiliency features available to the network participants. The Service Specific layer provides the building blocks for the application developers. The Application-level layer provides various services (e.g. file sharing, voice chat) to the end users [21].

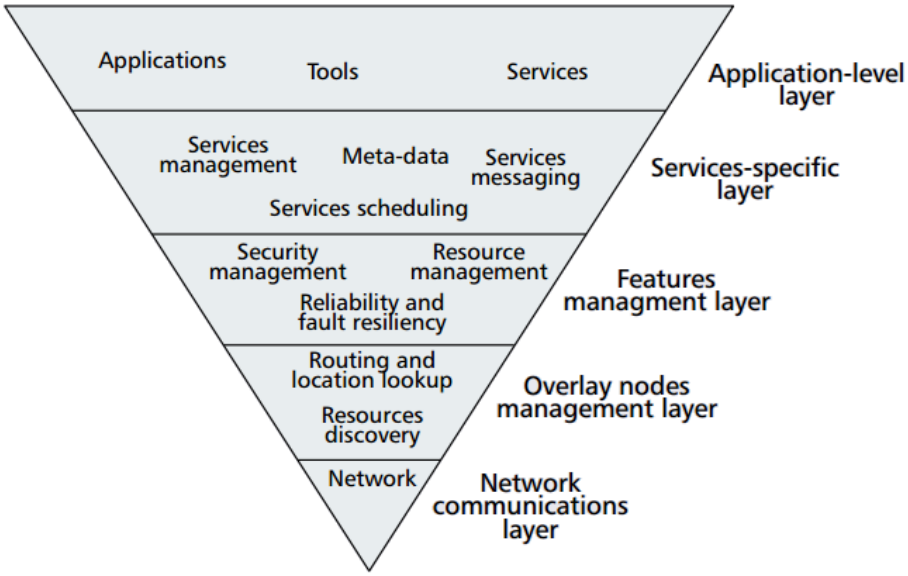


**Figure 2.8 – An abstract P2P overlay network architecture in** [21]

### 2.3.2 Peer-to-Peer Overlay Network Categories

A P2P overlay network can be structured or unstructured, based on the control of the network topology. The network topology in structured P2P overlay is tightly controlled. The

contents in a structured P2P overlay are placed in a specific location to make an efficient retrieval query. The structured P2P overlay network uses the Distributed Hash Table (DHT) [60] to identify and locate nodes or resources in the network. An example of structured P2P overlay is Chord [60].

In contrast, an unstructured P2P overlay has a loosely controlled topology. The contents in an unstructured P2P overlay are placed at a random location. Those contents are accessed by using a flooding technology. The unstructured P2P overlay networks are ad-hoc in nature. These types of P2P overlays are efficient for highly replicated contents but are inefficient for rare contents. An example of an unstructured P2P overlay network is the BitTorrent [61]. More details about P2P overlay networks and comparison between the structured and the unstructured P2P overlay network can be found in [21].

## 2.4 RESTful web services

This section discusses REST [22], REST principles [62], Resource Oriented Architecture (ROA) [23] and RESTful web services [23].

### 2.4.1 Introduction to Representational State Transfer

Representational State Transfer (REST) is an architectural style for the distributed hypermedia system. Different types of media, (e.g., text, images, and audio) can be stored and retrieved by a hypermedia system. REST was first introduced by Roy F. in his Ph.D. dissertation [22]. REST enables the design and development of the distributed network applications, using well known web technologies and protocols (e.g. HTTP, XML). REST uses the client/server architecture of the web. REST does not depend on any specific communication protocol [62]. A RESTful web service is described using the Web Application Description Language (WADL)

[63]. A WADL file describes a service, the Uniform Resource Identifier (URI) of the service, and the data that the service can process and generate.

### 2.4.2 REST Principles

REST is based on three main design principles, which are addressability, uniform interface and statelessness [62].

**Addressability:** REST models the information as resources. A resource is identified via a URI. A resource is any form of information that is important enough to be named and referenced.

**Uniform Interface:** A REST resource can be accessed and manipulated through a uniform and standard interface. This principle gives REST the advantages like familiarity and interoperability, regardless of the used technologies.

**Statelessness:** A REST request is self-sufficient, which means that the request has all the information that is necessary to process the request. No client information needs to be saved on the server to generate a response to a request. This property helps the applications to be scalable, easily implementable, and load-balance.

A resource representation is known as the information that represents the current state of the resource. It can be done in different ways (e.g., plain text or XML) which depends on the implementation of the resource. This representation is sent to the client in a response to a request. An example of a resource is a temperature sensor. Upon request from the client, the server reply with the current temperature, as a plain text.

### 2.4.3 **Resource Oriented Architecture**

The Resource Oriented Architecture (ROA) [23] is based on the concept of resources. A resource is anything important to be named and referenced as a "thing" itself (e.g. a document, a sensor, an item, etc.). It can be a physical object (e.g., a sensor), or an abstract concept (e.g., a search result). A resource should be associated with at least one URI to identify it on the web and that URI should be unique. ROA is a RESTful architecture. It provides a set of guidelines to develop RESTful web-services. ROA associates the REST principles with the web technologies and protocols. A URI is used to address a REST resource in ROA. It supports various representations such as JavaScript Object Notation (JSON), Extensible Markup Language (XML), HyperText Markup Language (HTML) and plain text. ROA uses four of the Hypertext Transfer Protocol (HTTP) methods to manipulate the resources. Table 2.1 shows the HTTP methods and the operations to manipulate the REST resources.

**Table 2.1 – The methods for manipulating REST resources**

| Method | Operation |
|---|---|
| POST | To create a new resource in the server |
| PUT | To modify an existing resource in the server |
| GET | To read a resource from the server |
| DELETE | To delete a resource from the server |

### 2.4.4 **Steps to Design RESTful Web Services**

A set of procedures is applied to design the RESTful web services [62]. The first step is to figure out the dataset for a specific service. Then, that dataset is split into several resources. Later, the following steps are applied to each of the resources:

1) Naming the resource with a URI

2) Identifying the subset of the uniform interface that is exposed by the resource
3) Designing the representations to accept from the client and to serve to the client.
4) Integrating this resource with the existing resources
5) Considering the typical course of events, like what should happen or what might go wrong

A detailed description about RESTful web services can be found in [23].

## 2.5 Constrained Application Protocol

Constrained Application Protocol (CoAP) [19] is based on the REST architecture and proposed by IETF. The Constrained RESTful Environment (CoRE) Working Group [64] has defined this protocol. CoRE also works for the CoAP standardization. The goal of the CoAP is to introduce the web-service paradigm in the constrained-network of smart objects [65].

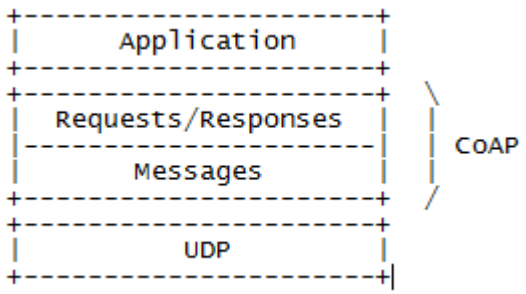


**Figure 2.9 – The abstract layering of CoAP in** [19]

### 2.5.1 Introduction to CoAP

CoAP is an application protocol for resource-constrained devices [19]. It is a web transfer protocol. It's HTTP like behavior makes it a good replacement for the HTTP in a constrained M2M environment. It is easily translatable to HTTP and from HTTP. Thus, CoAP can be used by any web-service. CoAP uses a similar client/server interaction model as of HTTP. A CoAP implementation in a node may act as both client and server for the M2M interactions. The CoAP

abstract layer is shown in Figure 2.9. Logically, CoAP is seen in two layers; the messaging and the request/response layer. The Messaging layer deals with the asynchronous nature of the interaction and interacts with the User Datagram Protocol (UDP) [66] layer. It also deals with the request/response interactions using a method and a response code.

### 2.5.2 **Relation between CoAP and HTTP**

CoAP makes use of four methods of HTTP [19], which are GET, POST, PUT and DELETE. Among these messages, GET is safe, and PUT is idempotent. These methods are used in the same manner as used in HTTP. This allows the easy translation of messages between these two protocols. This translation is usually done in a border node, called a proxy node. Following two features made CoAP a right choice for the M2M communication.

- Multicast messages
    - CoAP supports group communication or multicasting. A multicast message is used for discovery of a service.
- Synchronous and asynchronous messaging
    - CoAP supports separate response besides request/response model. A CoAP request can also trigger multiple responses (subscribe/notify).

CoAP also transfers significantly smaller amounts of data and causes less energy consumption for the same transaction compared to HTTP [65]. These features made the CoAP a popular protocol for the constrained-network.

### 2.5.3 **CoAP Request/Response Model**

A CoAP request or response is carried in a CoAP message. A CoAP message either carries a method code or a response code. A request is carried in a Confirmable (CON) or a Non-

confirmable (NON) message. An Acknowledgment (ACK) message carries the response to a CON request, which is known as piggybacked response. Figure 2.10 shows two requests with piggybacked responses.

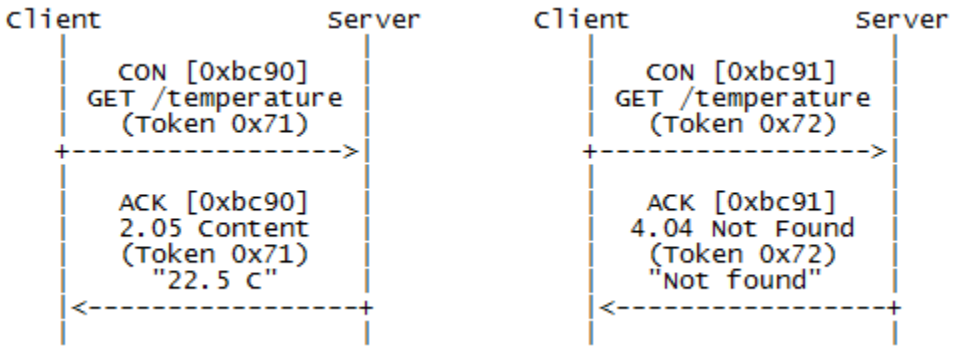


**Figure 2.10 – Two GET request with Piggy-backed responses in** [19]

The ACK message does not need any further acknowledgment. CoAP uses the UDP, a transport layer protocol. UDP uses a simple retransmission mechanism for the reliability of the transmission, whereas the Transmission Control Protocol (TCP) uses complex congestion control mechanism [67]. If a server cannot respond immediately, it replies with an empty ACK to stop the retransmission from the client. The server sends a response message to the client a later time when the response is ready. This process is called a “separate-response”. Figure 2.11 shows the message flow for a separate-response.

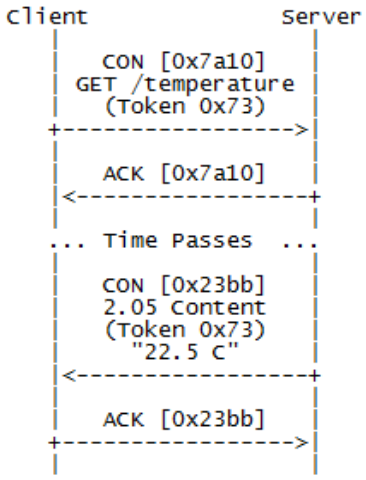


**Figure 2.11 – A Separate-Response to a GET request in** [19]

The response is sent using a NON-message if the request is also NON-type. A NON-request and the response are shown in Figure 2.12.

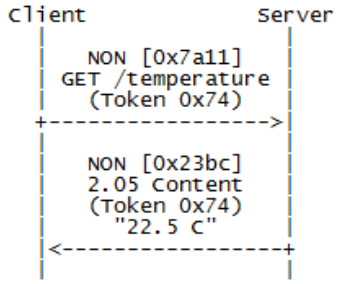


**Figure 2.12 – The response to a non-confirmable message in** [19]

Following is a summary of the types of CoAP messages.

### *2.5.3.1 Confirmable message (CON)*

A CON message is a request message. It requires an acknowledgment (ACK) message in reply with the same message ID. A client resends a CON message until it receives an ACK reply. A server replies with an empty ACK message if the response is not ready at the time of the request.

#### *2.5.3.2 Non-confirmable message (NON)*

A NON-message is also a request message. Unlike CON message, a NON-message does not require any ACK response. However, it is recommended to use only safe methods (e.g. GET) with the NON-type requests.

#### *2.5.3.3 Acknowledgement message (ACK)*

An ACK message acknowledges that a CON message was received and processed. An ACK message may carry the piggybacked response. The server sends an empty ACK if the response is not ready at the moment of request and sends the response separately (a separate-response).

#### *2.5.3.4 Reset message (RST)*

An RST message acknowledges an earlier request message (CON or NON), but saying that the recipient could not process the request. An RST message causes the requesting node to resend the request. This process applies for both the CON and NON messages.

### 2.5.4 **CoAP OBSERVE**

A CoAP client can observe a resource change in a CoAP server through subscribing to that resource. This publish/subscribe mechanism is based on an extension of the CoAP called Observe protocol [68], which is based on the Gang of Four [69] observer model. A client subscribes to a resource in the server. The server notifies the client upon changes of the resource with the latest representation. Figure 2.13 depicts an example of message exchanges between a client and a server for observing a temperature resource.

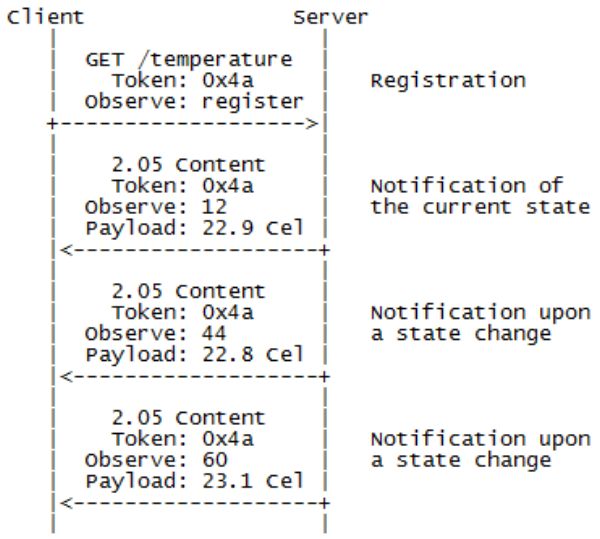


**Figure 2.13 – Observing a resource in CoAP in** [70]

## 2.6 Chapter Summary

This chapter discussed the thesis related background information. It introduced the SN and its example usages. It also discussed OpenSocial, an SN standardization. Moreover, it presented the M2M and P2P architecture and their applications. Finally, it explained the REST and CoAP with their principles and examples. The following chapter will present the motivating scenarios for the M2M enabled SN, and the requirements derived from those scenarios. It will also discuss the existing works for automatic sharing of the M2M information in SNs.

# Chapter 3: Motivating Scenarios, Requirements and State of the Art of the M2M Enabled SNs

This chapter presents the motivating scenarios of this thesis. It is followed by a set of derived requirements for enabling the M2M communication in SNs. It then discusses the related works. This chapter concludes with a summary of the evaluation of the related works.

## 3.1 Motivating Scenarios

There are many domains where we can benefit from automatic sharing of machine collected information in the SNs. Let us consider the first scenario where John is an elderly person and a member of an SN. He is connected to his family physician, family members and friends through the SN. His physician and family members remotely monitor his health condition from periodic posts in the SN. John has the risk of falls, as he is elderly. The SN can detect the fall by analyzing the sensory data. The SN notifies John's COIs by posting about fall and health condition to his profile.

As a different scenario, Mike is an active biker and a member of an SN. When he goes for biking, the SN posts the information received from the sensors on his body and his bike. The SN posts the distance, start point and end point, and the distance of his ride. It also posts his current heart rate and blood pressure. His coach subscribes to his biking related posts in the SN and may advise him on his training.

## 3.2 Requirements for M2M enabled Social Network

### 3.2.1 General Requirements

Based on the representative scenarios above, we derived a set of requirements for the potential solutions enabling M2M communication in SNs. The first requirement is that the architecture should support common features like authentication, posting, retrieving of SNs. M2M has its application in various domains (e.g., e-health, smart-grid). The proposed architecture should be able to incorporate all application domains. Therefore, the second requirement is that this architecture should be application domain independent. Third, the architecture should be scalable in terms of the number of M2M devices. As a fourth requirement, the architecture should follow standards and reuse existing SN infrastructure as much as possible. The rationale behind this requirement is interoperability and ease of deployment. Most of the M2M devices are resource-constrained, especially in terms of processing power. Therefore, the fifth and last requirement for the architecture is to support resource-constrained devices. Table 3.1 gives the summary of the general requirements.

**Table 3.1 – General Requirements**

| Index | Requirement |
|---|---|
| **1** | Supports common features of SN |
| **2** | Application Domain Independent |
| **3** | Scalability |
| **4** | Reuse existing SN Infrastructure and Follow Standard |
| **5** | Support Resource Constrained Devices |

### 3.2.2 Gateway Requirements

An M2M gateway is required for the resource-constrained devices (e.g., sensors) to aggregate and to process data. Also, the gateway connects the M2M devices with the internet and

SN. However, a single node gateway will certainly be a bottleneck for the M2M devices (e.g. sensors) to reach the SN server. It hence plays a vital role for scalability in terms of number of sensors that can connect through the gateway. The P2P overlay-based solutions are scalable [21]. We, therefore, choose a P2P overlay-based gateway on top of the smartphones to address the scalability issue. We set a set of requirements for the M2M gateway to address scalability and other general architectural requirements.

We derived five requirements that a gateway should meet to enable M2M in the SN. First, the gateway should be independent of the lower layer M2M technology and supports various types of M2M devices. Second, the gateway should minimize the communication overhead between M2M end-devices, the gateway and the SN. In details, the gateway has to support a lightweight communication mechanism between (i) M2M end-devices and the gateway, and (ii) the gateway and the SN, in order to support resource-constrained devices. The third requirement is that the gateway should be capable of self-organization to handle node joining and node leaving. Node leaving can be either voluntary or involuntary. Our fourth requirement is that the gateway should not have any permanent centralized point. The rationale behind the third and fourth requirements is to avoid a single point of failure that may cause a system failure. The fifth and last requirement is to support synchronous and asynchronous communications, as the system has to notify when changes happen. Table 3.2 contains a summary of the gateway requirements.

**Table 3.2 – Gateway Requirements**

| Index | Requirement |
|---|---|
| **1** | Independent of the lower layer |
| **2** | Lightweight Communication |
| **3** | Self-Organizing |
| **4** | No Permanent Centralized Point |
| **5** | Supports Synchronous and Asynchronous Communications |

## 3.3 State of the Art

The first idea of integrating the M2M with the SN is found in [71]. The authors talk about utilizing the sensory information in SNs, e.g., people's precise location. They also propose using SNs as a "storage infrastructure" for the sensor networks. However, in this thesis, we only take into account the solutions that enable automatic event sharing in the SN. Furthermore, we evaluate them with respect to the requirements. We also discuss existing P2P overlay gateway solutions and evaluate them with respect to the gateway requirements.

### 3.3.1 Event Sharing in SNs

Solutions that enable M2M information sharing in SNs can be categorized into two groups, based on whether the M2M devices send data directly to the SNs or use a gateway.

#### *3.3.1.1 Direct Posting in SNs*

The authors in [72] merge SNs with M2M networks to support the exchange of information. The intention is to provide services to both human users and machine users in pervasive computing environments. When sensors located next to plants tweets about the

humidity, the human users are informed. After receiving the information on Twitter, users water the plants if the humidity is low. Resource constrained devices do not have enough capability to connect directly to the internet. This approach does not support constrained devices and therefore does not satisfy our last requirement.

#### *3.3.1.2 Gateway-based Approaches*

SenseFace [73], is a framework for capturing sensory data (e.g., blood pressure, heart rate) from users' Body Sensor Network (BSN). It shares the processed information with users' COIs over variety of SNs such as Facebook and Twitter, and also via fax, emails and phones. Figure 3.1 shows the 4-tier architecture of the SenseFace. BSN is in the first tier. SenseFace users can be static or mobile depending on the environment they are in (e.g., in a car or a shopping mall). It is assumed that a user carries a mobile device which acts as a gateway for the BSN. The mobile network connects the BSN with the internet. Mobile devices act as personal gateway that analyzes the sensory data for any possible health issues. The personal gateway pushes the sensory data to a webserver in the third tier using the cellular network. This architecture considers an overlay SN that is extended over the generic SNs such as Facebook, YouTube, and Twitter in fourth-tier. SenseFace also considers existing communication facilities such as fax, email, voice mail, etc. as a part of one's overlay SN.

This architecture is application domain independent as the architecture can be used for any BSN integration with the SNs. This framework also supports resource-constrained devices. Sensors connect to the mobile device using Bluetooth. It uses existing social network infrastructures to share information. However, the SNs have no control over the sensory data and the data flow. It uses SN for information sharing rather than enabling the SN to analyze data.

Moreover, this architecture does not meet the scalability requirement in terms of M2M devices because of the single node gateway on users' mobile devices.

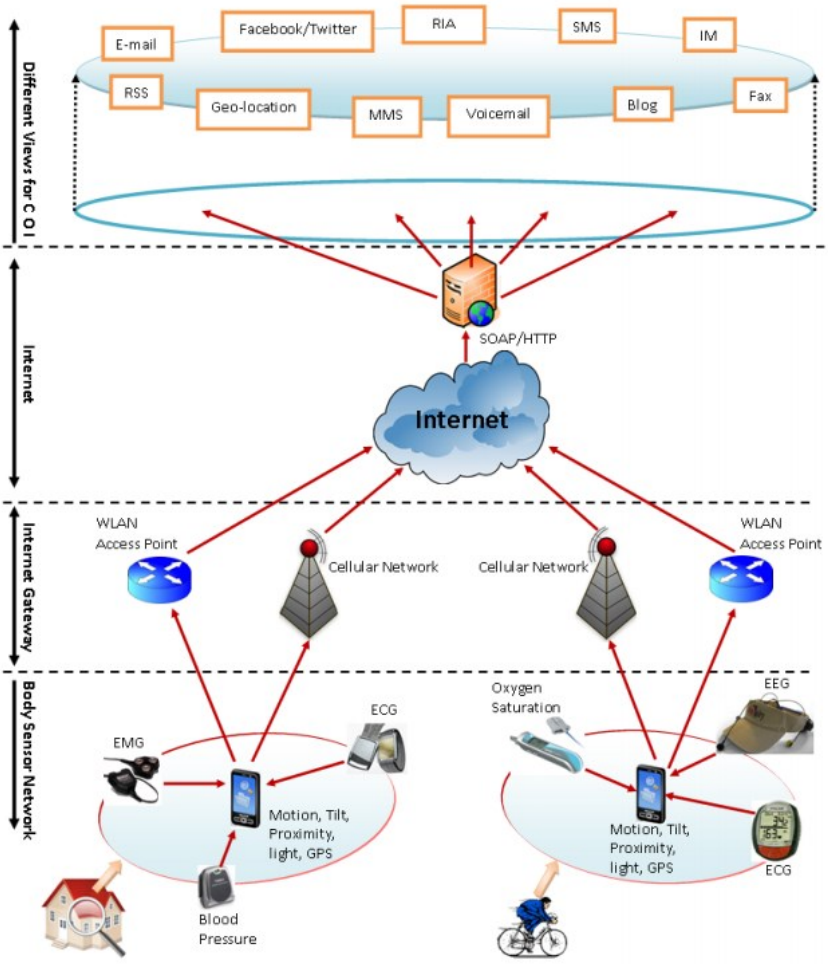


**Figure 3.1 – The SenseFace framework in** [73]

The author introduces a context-aware service architecture in [17] for integrating BSN and SNs through the IP Multimedia Subsystem (IMS) [74]. The proposed architecture is shown in Figure 3.2. Wearable sensors in the BSN monitor the vital signs or movements of the human body. A wireless smart device (e.g., a smartphone, a tablet pc) works as a monitoring station. This monitoring station forwards the sensory data to the IMS. An IMS-WEB 2.0 gateway is responsible for connecting social networks with the IMS using Session Initiation Protocol (SIP)

services (e.g. SIP presence). The architecture enables services like pervasive gaming and wireless healthcare applications. The author does not consider SN's internal structure to share BSN information and makes use of existing SNs to share. The monitoring station for the M2M network forms a bottleneck, and therefore the architecture is not scalable. This architecture does meet all the other requirements.

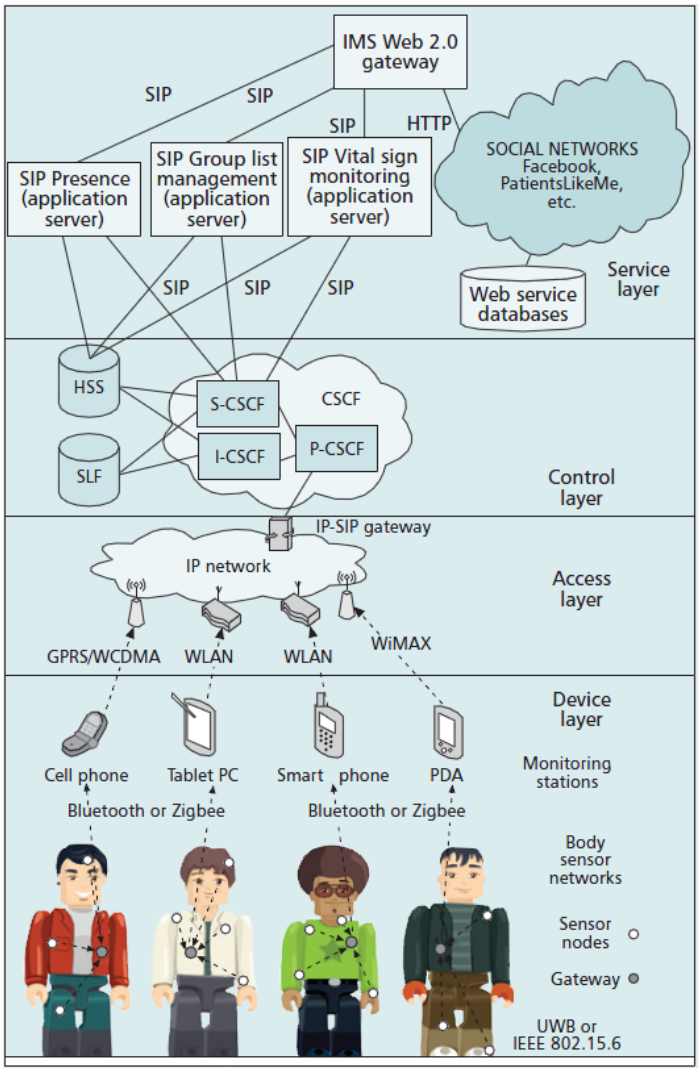


**Figure 3.2 – The architecture for integration of BSN and SN through the IMS in** [17]

To facilitate disease management, [75] proposes a pervasive health system. Chronic patients need their health information on a daily basis to manage their disease. This system

enables self-management by chronic patients, by letting users continuously monitor vital signs, like blood pressure, heart rate or temperature. Users also log and share health information with health care specialists and friends in SNs through the user interface on their smartphones, which enables them getting feedback or help. The smart phone also acts as a gateway for the vital sign monitor sensors. It is called a Mobile Base Unit (MBU).

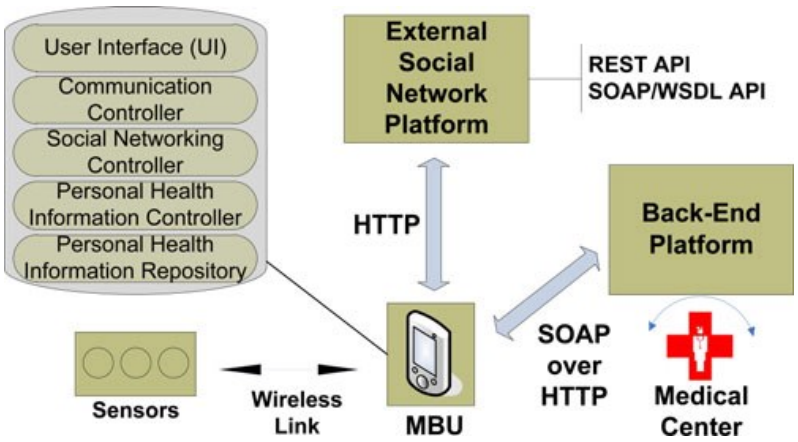


**Figure 3.3 – The proposed system architecture for integrative self-management in** [75]

The MBU is the core part of the architecture and consists of five layers. The Personal Health Information Controller works as the orchestrator of the application. The Personal Health Information Repository works as a local storage. Simple Object Access Protocol (SOAP) [76] is proposed for communication between the MBU and the backend. The Communication Controller handles the data transfer with the backend platform. The Social Network Controller uses appropriate API to publish health information in the external SN. This architecture supports resource constrained devices and enables publishing sensory information in the SN. As shown in Figure 3.3, the MBU is a bottleneck for the architecture, as it is therefore not scalable in terms of the number of M2M devices. This solution is specifically for health applications and hence is not application domain independent.

Drive and share (DaS) [42] is an SN that facilitates the sharing of traffic and personal information among drivers in vehicular scenarios. A web application provides the service. Users share events (e.g., an accident on the way) from smartphone via user interface. The backend infrastructure also auto generates events (e.g., bumpy road ahead) by analyzing the data received from the on-board sensors. The data reception and analysis by backend infrastructure allows the system to monitor vehicular behavior with the help of tracking functionalities. This vehicular monitoring feature helps to detect incidents like high traffic for suggesting alternative routes to the users. The smartphones use a cellular network or WiFi to connect to the backend infrastructure. DaS is specially designed for vehicular information sharing from smartphone browsers. This vehicular SN does not satisfy the application domain independence requirement. A smart phone or onboard computer acts as a gateway for onboard sensors to connect to the SN. This gateway allows the architecture to support resource-constrained devices. This single-node gateway makes the architecture not scalable in terms of number of M2M devices, thus not meeting the third requirement.

To the best of our knowledge, there is no solution that solely proposes an M2M enabled SN that addresses all of the aforementioned general requirements. Especially, the scalability requirement is not addressed by any of these solutions. As mentioned earlier in Section 3.2.2, P2P overlay based architectures are scalable. A P2P based overlay gateway can address the requirement of the scalability in terms of M2M devices requirement. The following subsection discusses existing P2P overlay based gateway solutions.

### 3.3.2 **P2P Overlay-based Gateway**

The authors in [77] have proposed a scalable P2P overlay based gateway architecture to integrate IMS and mobile sink based Wireless Sensor Networks (WSN). This gateway is located

on top of the users' cell phones. Cell phones are also used as sink nodes for the M2M devices. The gateway collects the data from the M2M devices through sink nodes and publishes it into the IMS presence server. The proposed gateway architecture is shown in Figure 3.4. The gateway is made up of peers. There are five groups of peers in the gateway. The first group works as Sink Entry Points (SEP) and interacts with the mobile sinks. The second group that interacts with the IMS Presence Service is known as Presence Service Entry Point (PSE). The gateway storage and processing is handled by the third and fourth group respectively. The super-peers from all four groups make the fifth group and enable interaction between all of them. The Super Sink Entry Point (SSEP) is in charge of the first group. It also decides whether the data received from a SEP should be stored, processed or published through the SPSE. The Super Presence Service Entry Point (SPSE) is the leader of the PSEs. The Super Data Management (SDM) manages Storage and Processor peers. The sensory data transfer is done using SIP in XML format between the gateway peers. The gateway is capable of self-organizing and self-recovery by exchanging the SIP messages.

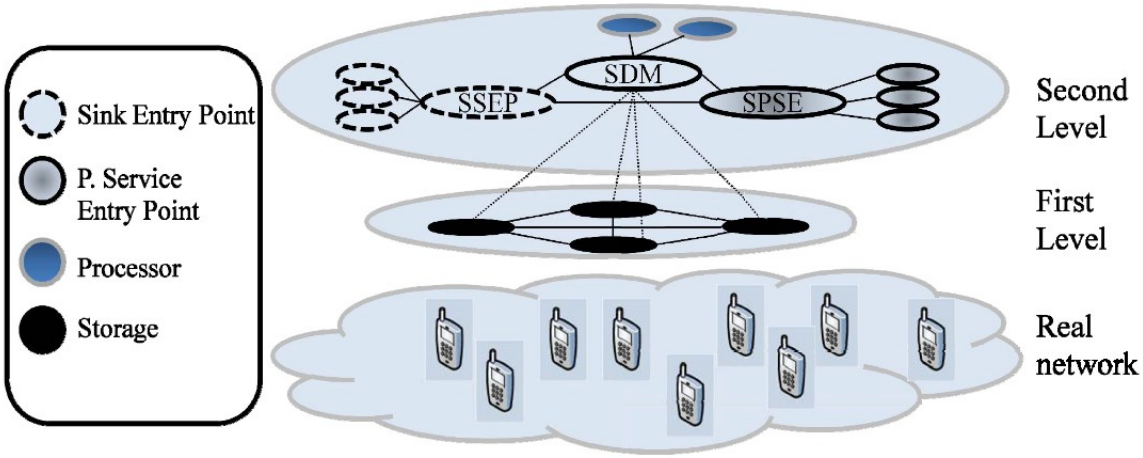


**Figure 3.4 – The overlay gateway architecture in** [77]

A similar gateway architecture is proposed in [78] to integrate the Vehicular Ad-hoc Networks (VANETs), IMS and WSNs. Both [77] and [78] use SIP for the communications between the nodes. Both the architectures meet the requirement of being independent of the lower layer M2M technology. They also satisfy the requirement of having no permanent centralized point. These gateways support both synchronous and asynchronous communications. However, the overhead due to SIP is higher than what is suitable for a resource-constrained device. The communication overhead can be minimized by using appropriate communication protocol that is much suitable for the M2M. Therefore, these architectures do not meet the requirement of minimizing communication overhead, i.e., they do not offer a light-weight communication mechanism. They also do not propose any solution for the involuntary departure of a gateway node. Both gateways meet all the other requirements.

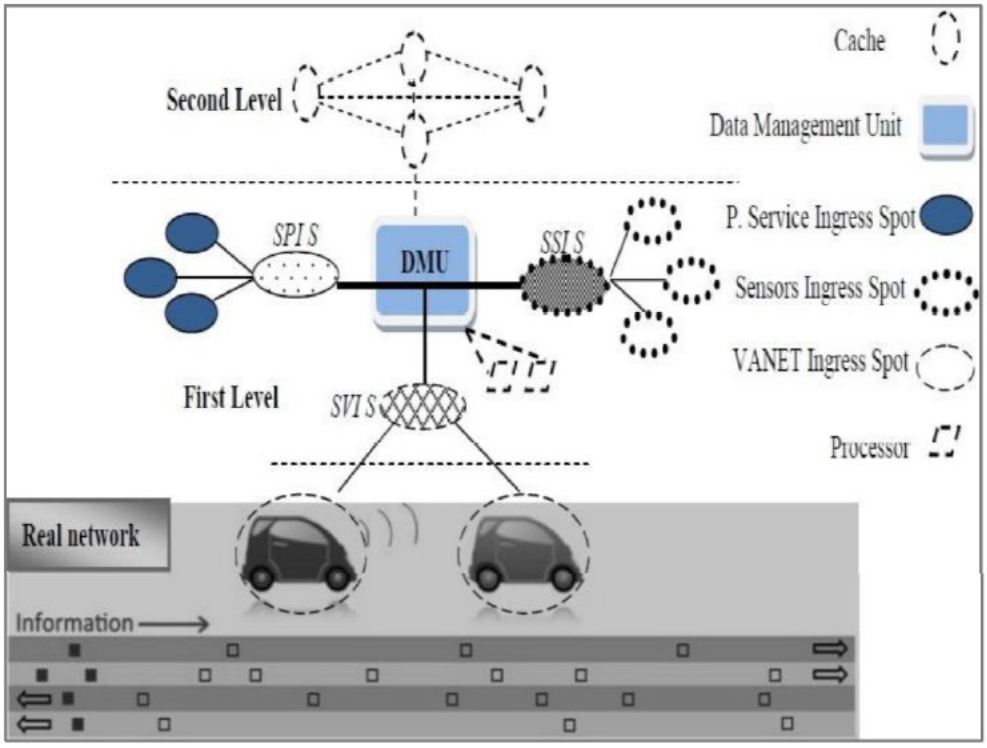


**Figure 3.5 – The overlay gateway architecture in** [78]

The gateway solution in [79] for home automation and management service uses a replication of services in all the overlay gateway nodes. It provides access to home devices from remote places. As an example, a landlord controls a centralized heating system, monitor electricity or gas usage of a building from his office without visiting the building. Figure 3.6 shows the overlay gateway architecture for the home automation and management framework. This framework is based on the service-oriented architecture. Networked devices (e.g., security camera, TV, printer) are the containers for the services and act as a peer on the network.

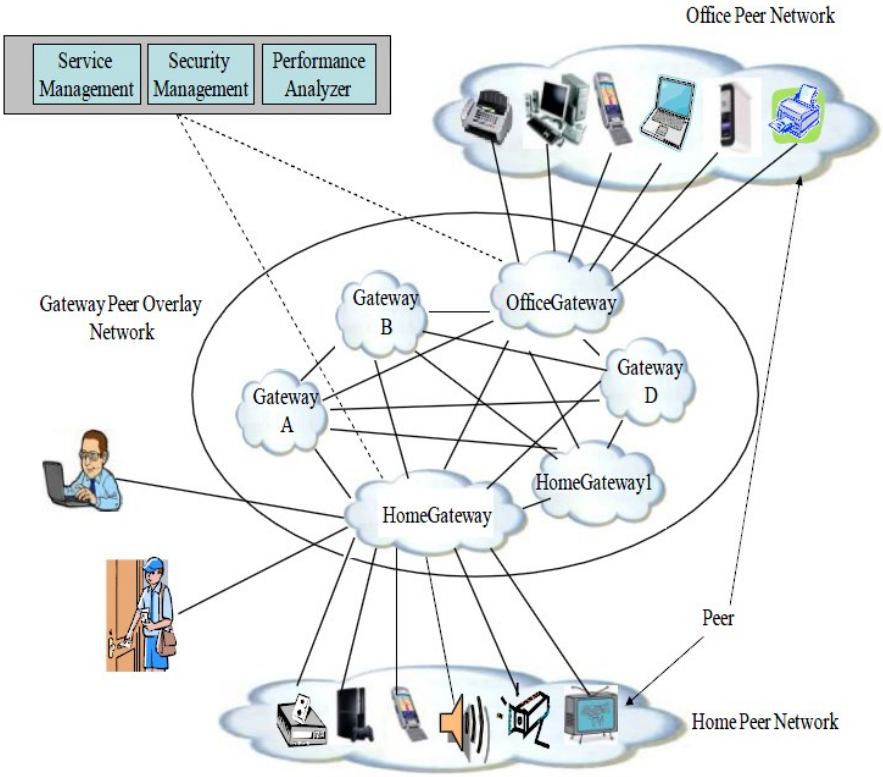


**Figure 3.6 – The home automation and management framework in** [80]

A peer first connects to the network by registering its peer ID, location, offered service(s) and software and hardware capabilities. There are three components in the gateway peer. The Service Management module holds the information about devices that are offering a gateway service. The Security Management handles the security aspects of a service requestor. The

Performance Analyzer keeps track of the resource availability on the gateway. A peer discovers the services through the gateways when they need them. The gateway has all the necessary information to access a service. A service is also replicated in several gateways. A service is available through other gateways in case of failure of a gateway node. XML messages are exchanged between the nodes over the JXTA protocol. This gateway solution meets all the gateway requirements but one. JXTA communications between nodes are not light-weight.

### 3.3.3 Summary of State of the Art Evaluation

All the related works are evaluated based on the requirements that are summarized in Table 3.1 and Table 3.2. This evaluation is assigned one of the following values:

- **YES:** The related work meets the corresponding requirement.
- **NO:** The related work does not meet the corresponding requirement.
- **PAR:** The related work partially meets the corresponding requirement.

**Table 3.3 – State of the art evaluation for General Requirements**

| | **Requirement** | **Ref**[72] | **Ref**[12] | **Ref**[17] | **Ref**[75] | **Ref**[42] |
|---|---|---|---|---|---|---|
| 1 | Supports common features of a SN | YES | YES | YES | YES | YES |
| 2 | Application Domain Independent | YES | YES | YES | NO | NO |
| 3 | Scalability | YES | NO | NO | NO | NO |
| 4 | Reuse existing SN Infrastructure and Follow Standard | PAR | PAR | YES | PAR | PAR |
| 5 | Support Resource Constrained Devices | NO | YES | YES | YES | YES |

Table 3.3 summarizes the evaluation of the related works for automatic M2M sharing in the SN compared to the general requirements.

The architecture proposed in [72] does not support resource-constrained devices hence does not meet out last requirement. SenseFace [12] considered an overlay network on top of existing SNs. None of the architecture supports the scalability requirement in terms of the number of M2M devices.

The architecture in [17] proposes sharing in existing SNs and supports both posting and retrieving. The proposed architecture in [72], [12], [75] and [42] do not discuss the internal architecture of the SN and retrieving the SN posts, thus partially meet the fourth requirement.

Table 3.4 shows the summary of our evaluation of the gateway related works in comparison with the gateway related requirements.

**Table 3.4 – State of the Art Evaluation of Gateway Requirements**

| | **Requirement** | **Ref**[77] | **Ref**[78] | **Ref**[79] |
|---|---|---|---|---|
| **1** | Independent of the lower layer | YES | YES | YES |
| **2** | Lightweight Communication | NO | NO | NO |
| **3** | Self-Organizing | PAR | PAR | YES |
| **4** | No Permanent Centralized Point | YES | YES | YES |
| **5** | Supports Synchronous and Asynchronous Communications | YES | YES | YES |

As we can see, no gateway solution meets the lightweight communication requirement. Both the solutions proposed in [77] and [78] do not support handling the involuntary departure. In [79], involuntary departure is handled by replicating services in multiple gateways.

## 3.4 Chapter Summary

This chapter introduced the motivating scenarios and a set of requirements for enabling M2M in the SN. It also described the related works and evaluated them with the derived requirements. None of the SN solutions meet all of the requirements. Especially, the scalability requirement is not handled by any of them. Additionally, the gateway solutions do not support lightweight communication mechanism. Next chapter proposes a novel architecture for M2M enabled SNs for automated sharing experiences for smart living. It will discuss the proposed architecture before describing the different entities and procedures with an illustrative scenario.

# Chapter 4: The M2M-Enabled SN: The Proposed Architecture

This chapter proposes an architecture to enable M2M communication in SNs. First it introduces the system overall architecture to meet the requirements presented earlier. It then presents the P2P overlay gateway architecture and discusses various components and functionalities. Later, the SN Server architecture is presented alongside the REST resources and the operational procedures of the SN Server. This chapter concludes with the validation of the architecture with respect to previously derived requirements.

## 4.1 An Overall Architecture of M2M enabled SN

The proposed M2M enabled SN architecture consists of three layers, which are the M2M, the network and the application layer. The overall architecture is shown in Figure 4.1. The M2M layer includes M2M devices, an M2M gateway and the M2M network. The M2M devices collect the data and send them to the M2M gateway through the M2M network. The gateway processes the data and sends them to the application layer. The application layer consists of the SN Server and the storage. The SN Server supports the general functionalities of an SN, for instance, posting in profile, retrieving friend's posts, add and removing friends from the network. It also receives the data from the M2M gateway. It then analyzes those data to find events like user body temperature is high (fever) or the user started running. If the SN detects any event, it saves the event in the storage and propagates event details in the SN. In the network layer, the internet and the access network provide the connectivity of the SN Server with the end-users as well as with the M2M gateway.

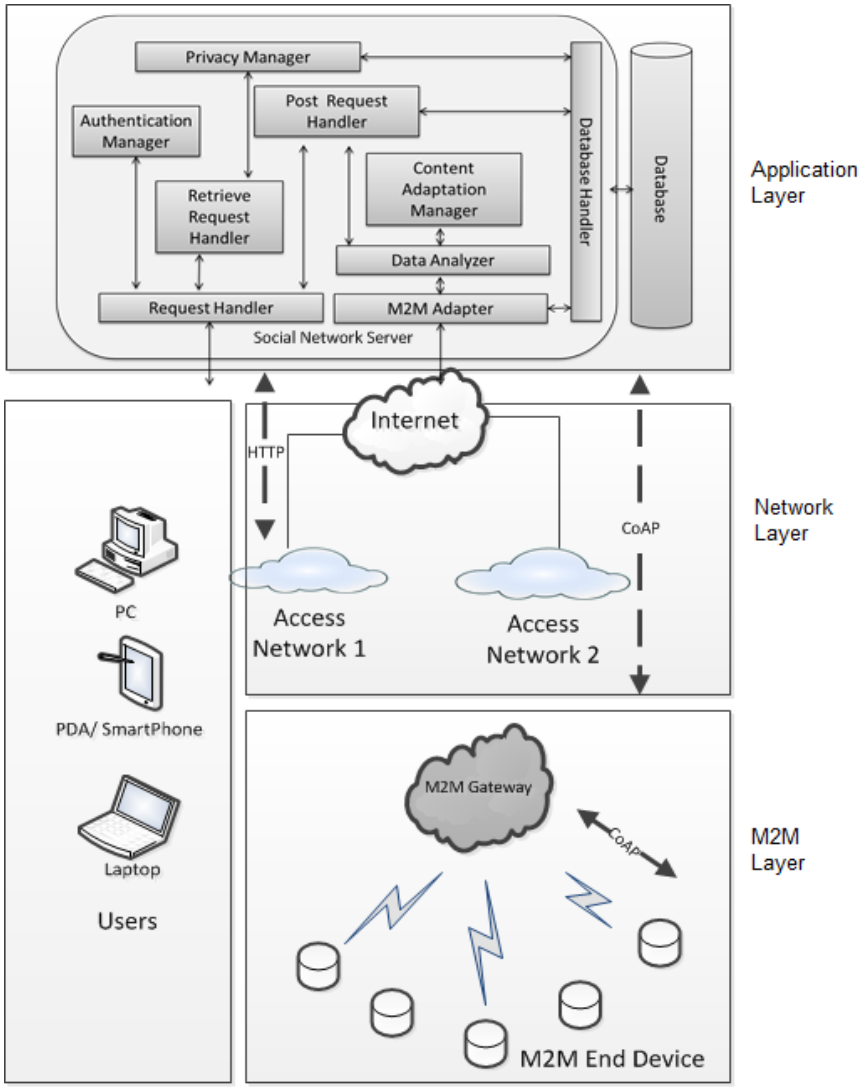


**Figure 4.1- The overall architecture of the proposed M2M-enabled SN**

As mentioned earlier in Chapter 3, the reason for the gateway based architecture is that resource-constrained devices (e.g., sensors) are less capable of data processing and storage. Additionally, resource-constrained devices are not capable of connecting directly to an SN. However, as shown in the overall architecture (Figure 4.1), the gateway will certainly be a bottleneck for the M2M devices (e.g. sensors) to reach the SN Server. It hence needs to be designed carefully as it plays an important role for scalability. We, therefore, choose a P2P overlay based gateway on top of the smartphones to address this scalability issue. More on

scalability in the overlay network can be found in [21]. We discuss the M2M gateway and the SN Server architecture in details in following sections. We first start with the gateway architecture.

## 4.2 The Gateway Architecture

Our gateway architecture is inspired by the architecture in [77]. It comprises three types of peer nodes, which are Sink Entry Point (SEP), Publisher Point (PP), and Data Storage (DS). All these nodes have super-peers to manage that group and the peers are grouped into four groups. The first group interacts with the mobile sinks. The second group communicates with the SN. The third group is the storage group, which is responsible for the local data storage in the gateway. The fourth group consists of the super-peers of all aforementioned three groups. A super-peer controls its peer group. It also talks to other super-peers to complete a task (e.g., forward data to the SN). The peer roles for the gateway are SEP, Super Sink Entry Point (SSEP), PP, Super Publisher Point (SPP), DS, and Super Data Management (SDM). The architecture of the gateway is shown in Figure 4.2.

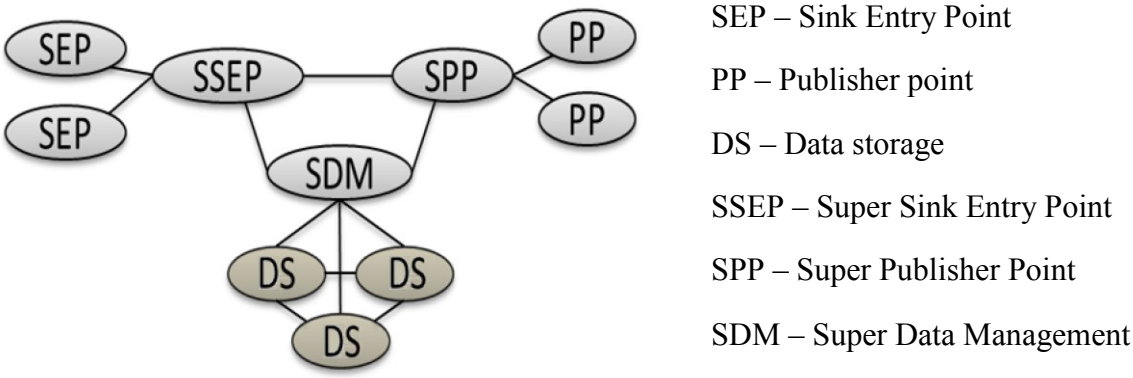


**Figure 4.2 – The overlay gateway architecture**

### 4.2.1 Node Functionalities

The nodes of the overlay gateway are known as peers. Based on the functionalities, there are three types of peers in the gateway. All three types of peers have super-peers. Only super-peers are able to exchange data with other types of super-peers. The functionalities of the peers are as follows:

- **Sink Entry Point (SEP):** An SEP works as an entry point to the gateway for the M2M end-devices through the mobile sinks. It interacts with the sinks and forwards the M2M data towards the SN. The SEP also sends requests to the M2M end-devices for fresh data. We assume that the data received from M2M end-devices in the sink node are in XML format.

- **Publisher Point (PP):** A PP forwards the data from the gateway to the SN Server to publish the information. It interacts with the SN Server using CoAP.

- **Data Storage (DS):** The DS nodes work as a local storage for the gateway. The storage capacity of the gateway depends on the capacity of the devices that are hosting the storage service.

- **Super Sink Entry Point (SSEP):** An SSEP has a similar role as an SEP. It also works as a super-peer for the SEPs. An SSEP interacts with other super-peers. It is the core of the overlay gateway and works as an orchestrator. An SSEP decides whether the information should be stored, filtered, processed on the gateway or be sent to the SN.

- **Super Publisher Point (SPP):** An SPP is the leader of the PPs. It decides which PP should forward data to publish to the SN. An SPP is also able to publish directly to the

SN in case there is no other PP. The SPP talks to other super-peers on the gateway to accomplish different tasks like query a data in storage or forward a request to read a sensory data.

- **Super Data Management (SDM):** An SDM peer works as a supernode for the DS nodes. It replies with stored data to the requests from other super-peers of the gateway.

### 4.2.2 Overlay Protocol

We chose the CoAP [19] as an overlay protocol. We choose CoAP for several reasons. The CoAP is a lightweight and standard-based protocol. It supports both synchronous and asynchronous communications. This protocol is used for the communication between the gateway peers and also for the communication between the M2M devices and the SN server. The CoAP follows the REST architectural style.

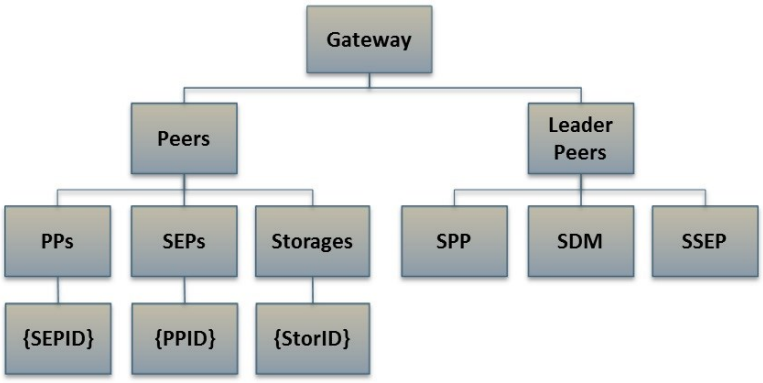


**Figure 4.3 – The gateway resource model**

### 4.2.3 Node Management

We assume that the overlay role of each peer is predefined. Each peer knows its role and thus the group it needs to connect to. All the leader peers are assigned with a multicast address, and all the other peers know this address. A peer sends a CoAP discovery request to the gateway multicast address to discover its super-peer. The discovery request contains the information

about the super-peer, which it is looking for. An appropriate super-peer (e.g., SSEP, SDM, or SPP) reply to that request with the address the peer should connect to. When connected, a peer sends all of its requests to its super-peer. The resources in the gateway should be modeled as CoAP is based on REST architectural style. The REST resource model of the gateway is shown in Figure 4.3.

**Table 4.1 – REST interfaces for the gateway node discovery**

| Resources | Operation | CoAP action |
|---|---|---|
| Super Sink Entry Point | Read: get the information of Super Sink Entry Point | GET :/leaders/SSEP |
| Super Data Management | Read: get the information of Super Data Management | GET :/leaders/SDM |
| Super Publisher Point | Read: get the information of Super Publisher Point | GET :/leaders/SPSE |

Table 4.1 lists the REST requests for discovery of the leader peers. A leader peer responds with its address when it receives discovery requests. If there is no leader in that group yet, the requesting node becomes the leader for the group. The multicast address is also assigned to the peer, and it starts listening to the gateway multicast address. The gateway peer joining procedure is depicted in Figure 4.4. A peer in a group knows about all the other peers in the group. Each time a group changes (a peer joins or leaves the group), the leader sends the information to all the members of that group and handle the voluntary departures.

However, the gateway proposed in [77] does not support all the gateway related requirements. The communications between the nodes use SIP, which has a high overhead than CoAP, especially considering the resource-constrained devices. Also, it does not handle the

involuntary departure of a gateway node. We propose handling the involuntary departure of the nodes.

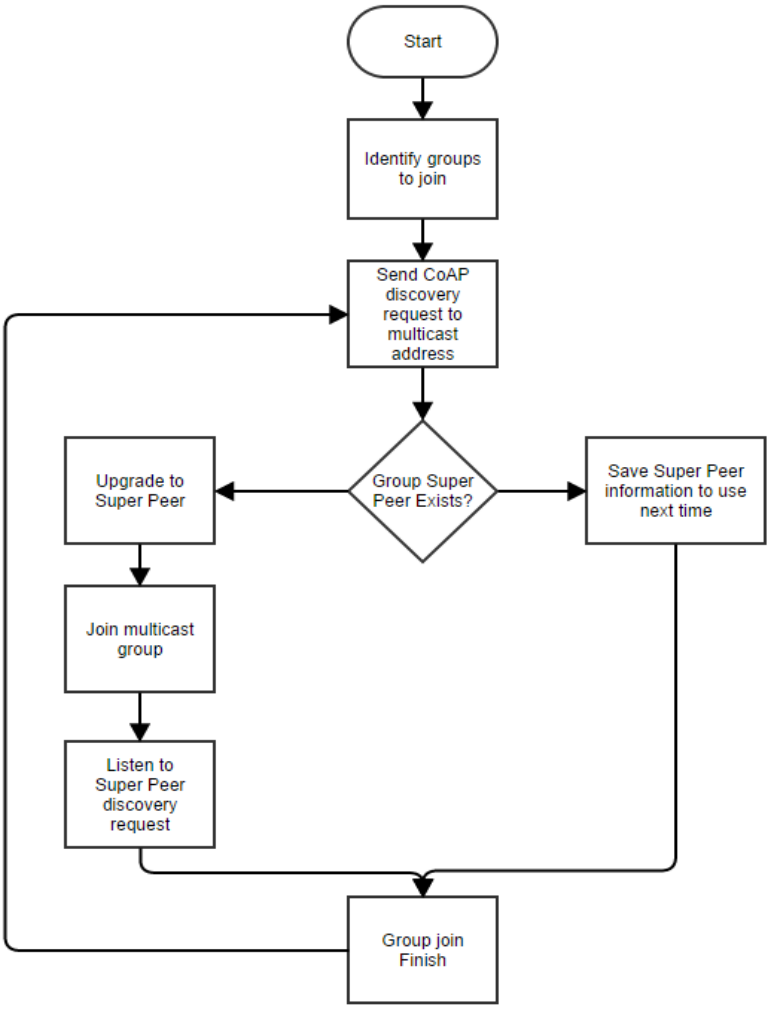


**Figure 4.4 – The gateway node joining process**

### 4.2.4 Handling Failures and Involuntary Departures

The gateway stores a small amount of data (e.g., configuration data, temporary data for processing) in the DS nodes (storage peers). The gateway requests the corresponding M2M device for the fresh data in case of failure of a DS peer. The gateway stores the data again in DS node for a future usage. The gateway requests for configuration data from the SN server if it cannot find the data in the DS peer. In case of failure of a super-peer (a voluntary or involuntary

departure), the gateway uses the popular heartbeat mechanism, a recovery mechanism as proposed in [81].

### 4.2.5 **Posting Procedure on Gateway**

The gateway is functional if there is at least one SEP and one SPP for a posting procedure. An SSEP subscribes to an SEP and in turn SEP subscribes to a sink node. To achieve a subscription option, a CoAP GET request message contains the observe option that is set to true. When a change occurs in a resource, a response message, 2.05 Content, is forwarded to the SSEP. The reply message contains the data as a payload in the CoAP response. The SSEP decides if data should be saved in the SDM or will be sent to the SPP for forwarding to the SN immediately. The SPP chooses a PP and forwards the request upon getting the CoAP POST from SSEP. Figure 4.5 depicts the message flow on the gateway nodes for a posting procedure where an SSEP is subscribing to a sink node. When the SSEP gets the content, it creates a POST request to the SPP. The SPP forwards the request to a PP, which sends the request to the SN Server.

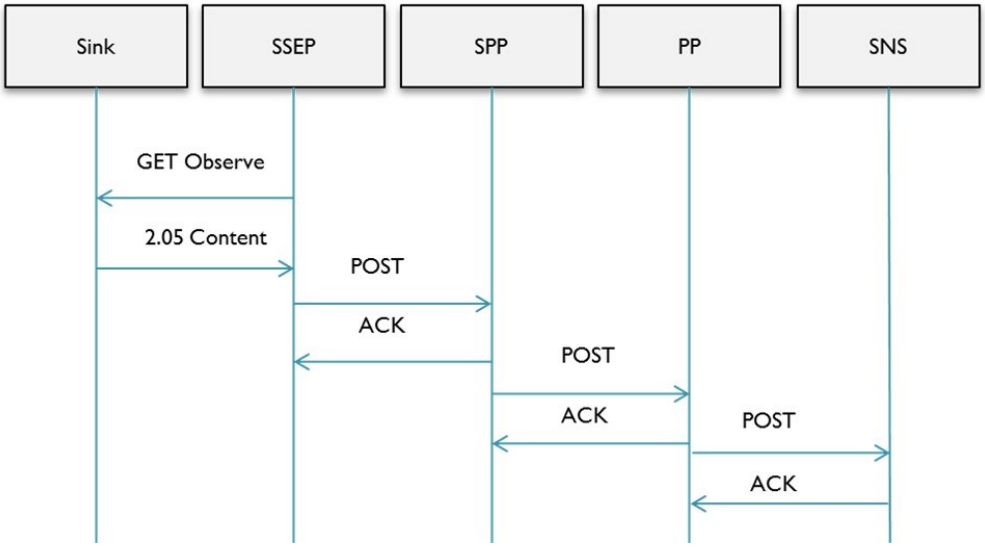


**Figure 4.5 – The posting procedure in the gateway nodes**

## 4.3 The SN Server Architecture

The proposed architecture has two components in the application domain; an SN Server and a Database. The SN server includes several components as shown in Figure 4.6 to enable the M2M and SN functionalities. The Database Handler in the SN Server makes it possible to communicate with any external database. The SN Server is an important part of our architecture that enables the data analysis and sharing in the SN. It also converts the M2M data to human readable information/image before sharing. Here we talk about the SN Server components and their functionalities.

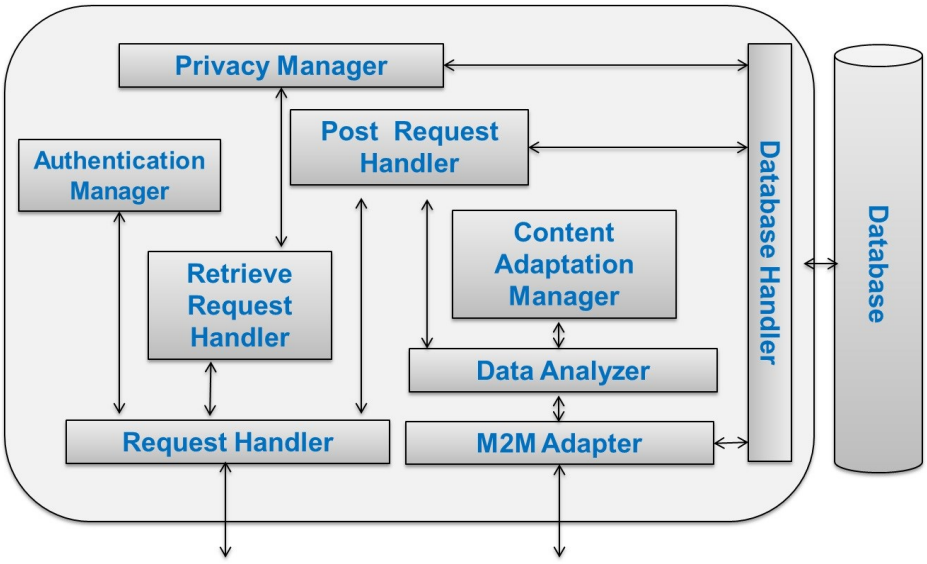


**Figure 4.6 – The SN Server architecture**

- **Request Handler:** The Request Handler is the first point of contact to the users. It receives and replies all kind of requests from the users. It then forwards the user requests to the appropriate handler.

- **Post Request Handler:** The Request Handler forwards all the post related requests to the Post Request Handler. It then creates the post and saves it in the database through

database handler. The Post Request Handler also process requests from the Data Analyzer.

- **Retrieve Request Handler:** The Retrieve Request Handler processes all the retrieve requests. It retrieves all the posts and shared-information that belongs to the user from the database/repository and sends them back to the users.

- **Authentication Manager:** The Authentication Manager handles the authentication and authorization of the users' requests. The Request Handler authenticates all requests by forwarding them to the Authentication Manager. Only requests from an authenticated user are forwarded to the other handlers.

- **Privacy Manager:** The Privacy Manager is responsible for retrieving the posts according to privacy settings of the users. A user can configure his/her posts as private or public. A user also can select posts for specific user or a group of users.

- **M2M Adapter:** The M2M Adapter maps the M2M devices with a profile in the SN Server. It also works as an interface to the SN Server for the M2M devices. The M2M Adapter talks with the M2M devices and forwards the information to the Data Analyzer.

- **Data Analyzer:** The Data Analyzer analyzes M2M data using various algorithms or applications that are available for the event detection. It raises an alarm by creating a post in the SN upon discovery of the event.

- **Content Adaptation Manager:** The Content Adaptation Manager is responsible for translating the M2M data to SN sharable information. The M2M data can be converted to human readable text, images or video.

- **The Database Handler:** The Database Handler talks with the database to save all the profile information and the M2M data to the server repository.

### 4.3.1 REST Resources of the SN Server

We choose OpenSocial as it is the only standard of the SN. The proposed SN Server resources are based on OpenSocial API. OpenSocial uses a RESTful interface for manipulating the SN Server resources [20]. The SN Server resources are shown in Figure 4.7. We have extended OpenSocial to support M2M in the SN Server. The applications that run on top of OpenSocial are interoperable with any SN that supports OpenSocial API. That is also a reason to opt OpenSocial for the SN Server.

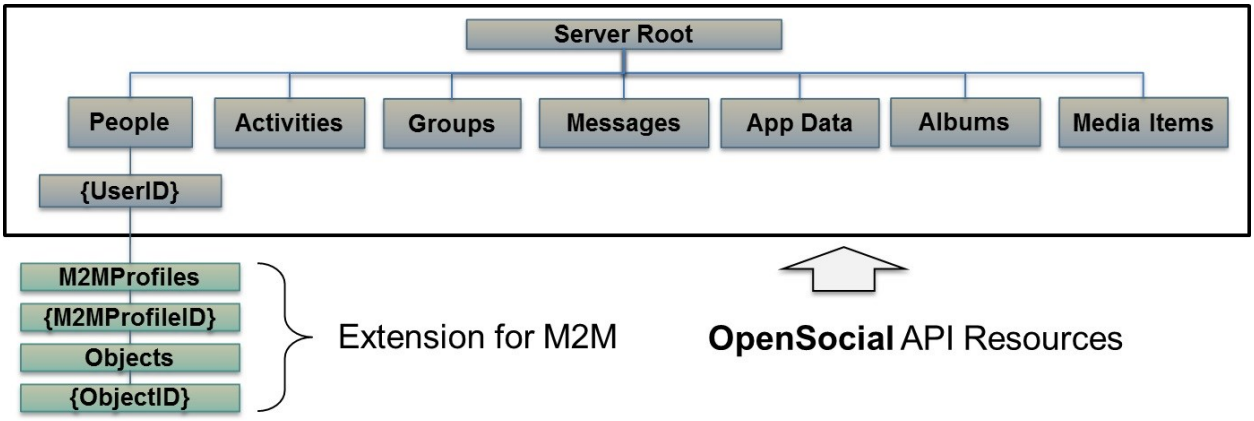


**Figure 4.7 – The SN Server REST resources**

Table 4.2 shows the resource modeling of the M2M extension for the SN Server. The M2MProfiles resource is a child to the OpenSocial People resource. A person may have multiple M2M profiles, each of which may have several M2M objects (e.g. one temperature sensor and one accelerometer under one health M2M profile). For example, if anyone wants the value of a temperature sensor with id “temp002” of M2M profile “healthProfile” of user “bob”, he has to

send a GET request to "/People/bob/M2MProfiles/healthProfile/Objects/temp002" URI. The reply depends on the resource definition of the 'temp002' sensor.

**Table 4.2 - The SN Server REST Resources**

| Resource | Operation(s) | HTTP Action(s) |
|---|---|---|
| List of all M2M profiles of a user | Create: create a new M2M profile for the user | POST:/People/{userID}/M2MProfiles |
| | Read: get all the M2M profiles of the user | GET:/People/{userID}/M2MProfiles |
| A specific M2M profile of a user | Read: get information about a specific M2M profile (e.g., name, number of objects) | GET:/People/{userID}/M2MProfiles/{M2MProfileID} |
| | Update: modify a specific M2M profile | PUT:/People/{userID}/M2MProfiles/{M2MProfileID} |
| | Delete: delete a specific M2M profile | DELETE:/People/{userID}/M2MProfiles/{M2MProfileID} |
| List of all M2M objects of an M2M profile | Read: get all the M2M objects of the M2M profile | GET:/People/{userID}/M2MProfiles/{M2MProfileID}/Objects |
| | Create: create an M2M object under M2M profile | POST:/People/{userID}/M2MProfiles/{M2MProfileID}/Objects |
| A specific M2M object | Read: get information of the M2M object (e.g. type of object, value) | GET:/People/{userID}/M2MProfiles/{M2MProfileID}/Objects/{ObjectID} |

### 4.3.2 Operational Procedures of the SN Server

The proposed SN enables the users to share posts among his/her network. It also enables the M2M to share on behalf of the user. The SN supports registering a new user (creating a new profile), authentication of the user (e.g. login, logout). Users can send a friend request to another

user to include within his/her network. The other user has to accept a friend request to approve the connection. Information posting and retrieving are done using the web user interface. This SN also supports M2M device management and analyzes M2M data. The Content Adaptation module converts M2M data into human readable information in the SN Server.

## 4.4 Validation against Requirements

### 4.4.1 The Overall Architecture

An OpenSocial based SN supports all the basic features of a traditional SN. Therefore, our proposed architecture meets the first requirement. The Data Analyzer module is extensible to support different application domain. Thus, the architecture satisfies the second requirement. Our proposed gateway inherits the scalability from the P2P overlay nature of the architecture and makes the architecture meet the third requirement. OpenSocial has reference implementations that are easily deployable and ready to use. The architecture satisfies the fourth requirement by using an OpenSocial reference implementation. We have extended OpenSocial resources to support the M2M communication. The gateway allows the resource-constrained devices to connect to the SN Server thus the architecture meets the last requirement as well.

### 4.4.2 The Gateway

We have chosen a P2P overlay-based gateway that makes the gateway scalable. We also have proposed the CoAP as the protocol for the gateway. It works on the application layer of the network stack. The CoAP is a lightweight and standard protocol with low overhead. It is designed for the resource-constrained devices [82]. Furthermore, it fosters the scalability of the gateway. CoAP also supports both synchronous and asynchronous communication. Thus, the gateway meets the first, second and fifth requirements. The gateway also supports self-

organization of the nodes and has no permanent centralized point, therefore, meets the third and fourth requirements.

## 4.5 Chapter Summary

In this chapter, we introduced an architecture to enable M2M communication in the SN. We proposed an OpenSocial based SN architecture. To address the scalability requirement, we proposed a P2P overlay based gateway that uses CoAP as the overlay protocol. We discussed all the functional components of the architecture with the REST resource model of the gateway resources and M2M extension to the OpenSocial. We also validated the design of the architecture against the requirements. Next chapter will discuss the software architecture, the prototype implementation and a partial performance evaluation of the prototype.

# Chapter 5: Implementation and Evaluation

This chapter presents a detailed software architecture of the SN and the gateway, a proof concept implementation and its partial evaluation. The detailed software architectures show the internal components of the proposed architecture, gateway, and describes the functionality in details. It also introduces a fall detection algorithm which is used for analyzing tri-axis accelerometer data in the SN. It concludes by presenting the proof of concept implementation and a partial performance evaluation.

## 5.1 Software Architecture of the SN Server

The software architecture of the SN server and its components are shown in Figure 5.1.

- **The M2M Adapter:** Data received from the M2M layer is first received by the M2M Adapter. It has two components, a CoAP-HTTP Proxy and an Object-Profile Mapper. CoAP requests are converted to HTTP requests by the CoAP-HTTP Proxy. The Object Profile Mapper parses and maps the data with a profile. Then data is forwarded to the Data Analyzer.

- **Data Analyzer:** Data Analyzer is responsible for all M2M related data analysis. It comprises the Event Detector and an array of Data Analyzer plugins. The Event Detector looks for potential events with the help of appropriate data analyzer plugin. For instance, accelerometer data analyzer for fall detection is used by the Event Detector module if an accelerometer data is received.

- **Content Adapter:** When the Data Analyzer detects any event (e.g., found fever by analyzing body temperature), it forwards those data to the Content Adapter. The Content Adapter converts them into suitable SN shareable information. A post is then created in the SN by sending a post request to the Post Request Handler.

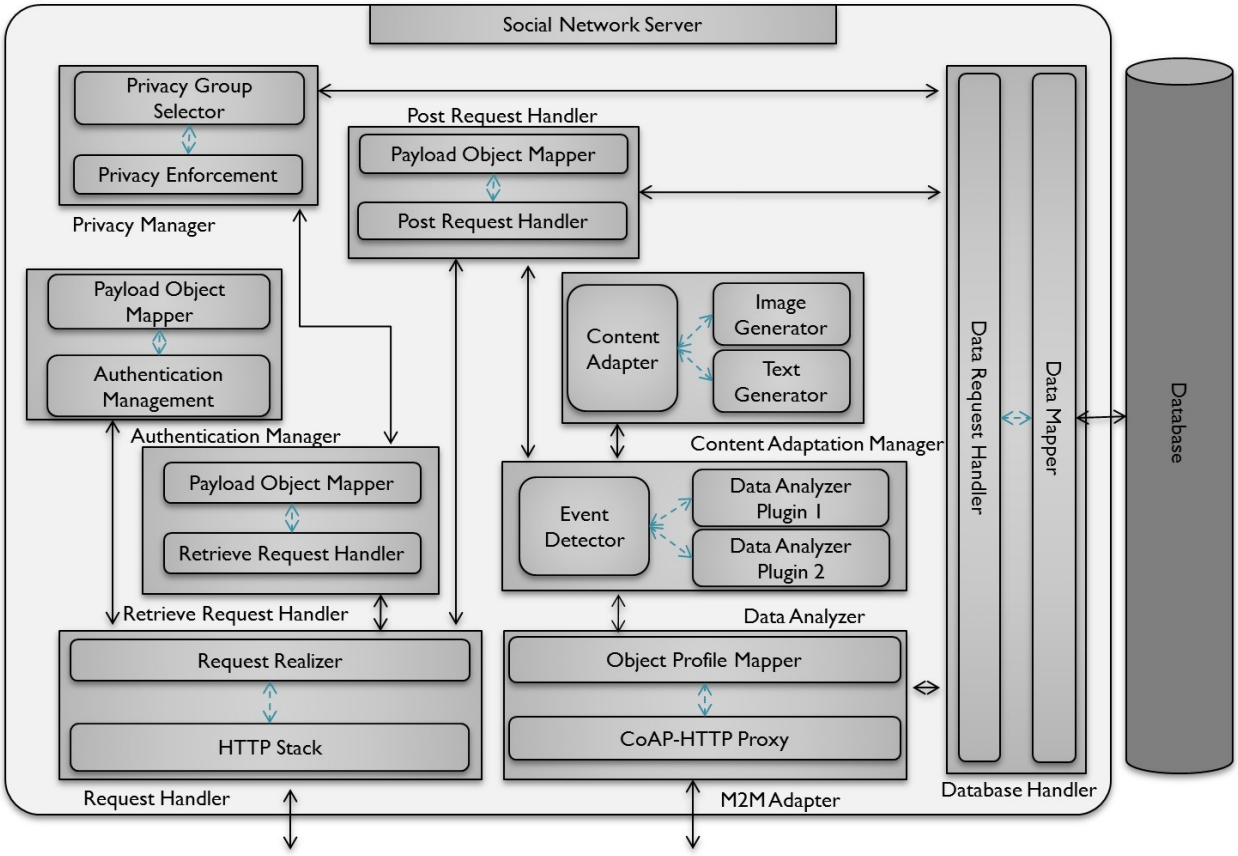


**Figure 5.1 – The software architecture of the SN Server**

- **Request Handler:** Users can post from a web browser. Any browser-based post or retrieve request is first realized by the HTTP Stack of the Request Handler. HTTP Stack converts the HTTP requests to system objects and forwards the request to the Request Realizer. Request Realizer first authenticates the user by forwarding the message to the Authentication Manager. After authentication, it forwards the request to the Post Request Handler or Retrieve Request Handler.

- **Authentication Manager:** The Authentication Management component makes sure that all the requests are authenticated before perform any operation in the SN. The Payload Object Mapper retrieves the credentials from the request and talks with the Database Handler for authentication purpose.

- **Post Request Handler:** The Post Request Handler receives requests from both the Request Handler and Data Analyzer. The Payload Object Mapper changes a request payload into system-object to create a post to the SN.

- **Retrieve Request Handler:** The Retrieve Request Handler generates contents to send to the user. The contents are typically a list of activities within users COIs. The Retrieve Request Handler confirms that the proper content is being retrieved with the help of the Privacy Manager.

- **Privacy Manger:** In Privacy Manager, the Privacy Group Selector searches database for information with whom (e.g. friends) the information should be shared. The Privacy Enforcement knows the users' COIs through the Privacy Group Selector and ensures that the proper content is being generated.

- **Database Handler:** The Database Handler has two components. The Data Request Handler is the interface for the other components to talk to the database. The Data Mapper module is responsible for talking to different databases. Changes in the Data Mapper make the SN possible to exchange data with different databases.

## 5.2 Software Architecture of the Overlay Gateway

The software architecture for the overlay gateway is depicted in Figure 5.2. All the participating peers in the overlay gateway follow the same architecture. The details about the components are given below.

- **Gateway Services:** This module contains the logic for the services that a peer has to offer. A peer can play any role that is preconfigured (e.g., SEP, SSEP, DS).
- **Role Manager:** The Role Manager contains the information about the role of a peer. It makes sure the services are available from the Gateway Service as per assigned role.

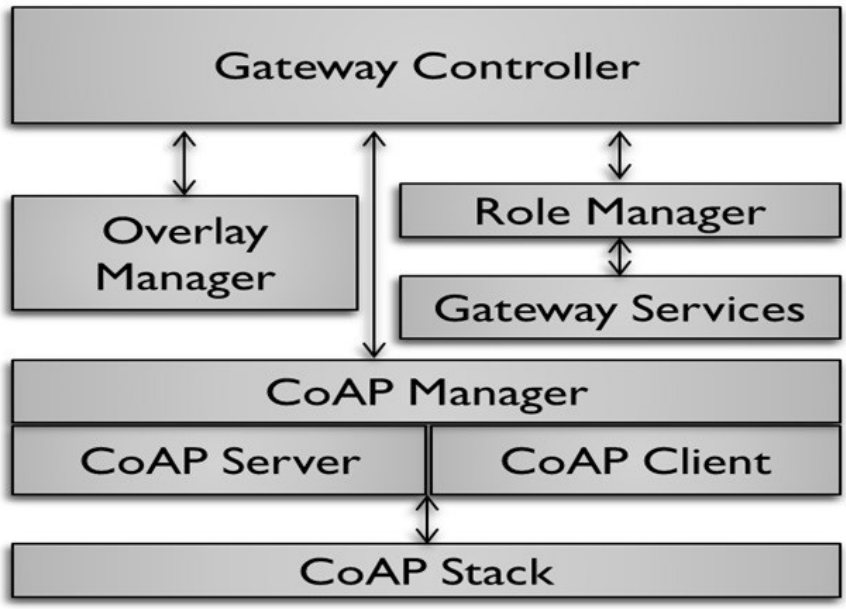


**Figure 5.2 – The gateway peer software architecture**

- **Overlay Manager:** The Overlay Manager module allows the discovery of the other nodes in the overlay. As mentioned earlier, discovery is done by sending CoAP requests to a multicast address. The Overlay Manager saves the information about the overlay and helps the CoAP Manager to create the CoAP requests and responses for appropriate peer and super-peer.

- **Gateway Controller:** The Gateway Controller works as the orchestrator of the gateway peer. It connects all the modules to avail the gateway services. The Gateway Controller contains the information of what role the peer is playing and what group it joins.
- **CoAP Manager:** The CoAP Manager creates all CoAP messages for requests and replies. The CoAP Manager uses both CoAP Server and CoAP Client depending on the type of messages it has to create. It uses an integrated lightweight CoAP stack to achieve CoAP functionalities on the peer.

## 5.3 An Illustrative Scenario

Let us illustrate the operations of our architecture with the scenario of John in Section 3.1. John is wearing an accelerometer sensor that can sense rapid movements. The sensor sends the data periodically to the SN server through the gateway on his smartphone. The gateway only forwards data to the SN if it detects any movement by analyzing the data. The SN Server also analyzes the sensory data in more details. When the analyzer detects a fall event, it requests content adaptation for the SN and posts about the event in the SN. Figure 5.3 shows the sequence diagram for the message exchange between the different modules in the SN for this posting procedure. The M2M gateway sends a CoAP POST request to the M2M Adapter. M2M Adapter converts the COAP request to an HTTP request and maps sensory data with his profile. It then sends a POST request to the Data Analyzer. The Data Analyzer detects events with the help of the appropriate analyzer plugin. In this scenario, a fall detection plugin is used. Data is converted to human readable text by the Content Adaptation Manager. The SN then creates a post in John's profile, which is available to his COIs.

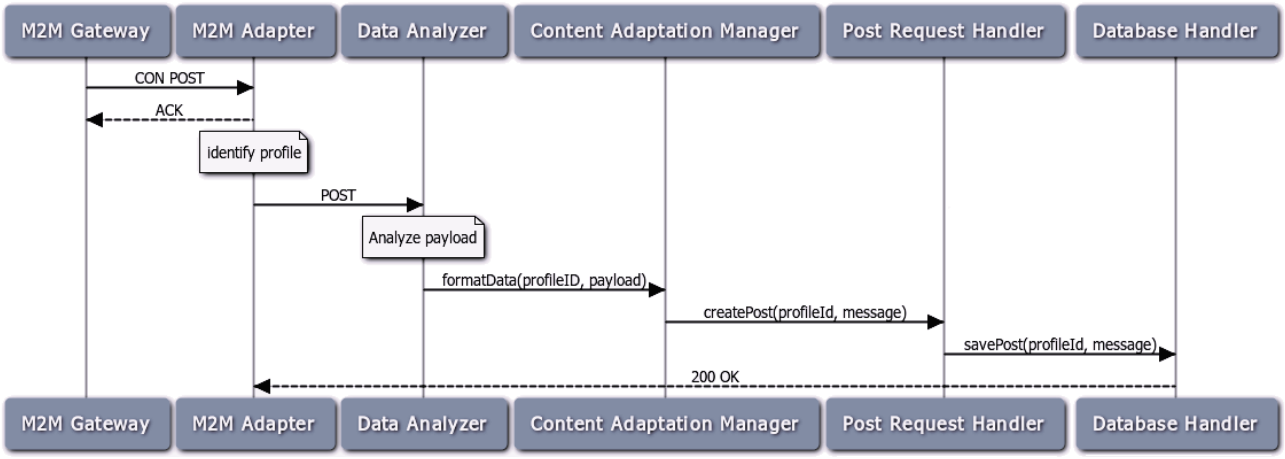


**Figure 5.3 – An illustrative scenario of a posting procedure**

## 5.4 Proof of Concept Implementation

We have realized the scenario mentioned in the previous section. Users can access the SN from a web browser to view posts in the SN. Our implementation includes the Request Handler, Authentication Manager, Post & Retrieve Request Handler, M2M Adapter, Data Analyzer, and the Content Adapter. The Content Adapter generates textual posts for the SN. The gateway implementation includes the SEP, SPP and SDM. The SEP performs filtering of data before forwarding to the SPP. The Data Analyzer detects events using an algorithm named Modulus Acceleration Monitoring as proposed in [83] to detect a fall from tri-axis accelerometer data. The gateway needs one SSEP and one SPP to be functional and perform an automatic post in SN. The SSEP subscribes to the sink node. When a request arrives in the SSEP, the SSEP forwards the data to the SPP to publish it in the SN.

### 5.4.1 Prototype Functionalities

The prototype implemented the following functionalities for the users.

- Users can register and/or edit their profile.
- Authentication is implemented for the SN users through Login/Logout feature.

- Users can add friends. Two actions together works to enable the feature:
    - Send a Friend Request
    - Accept a Friend Request
- Users can post text-based information on their profile using a web browser.
- Automatic event detection is implemented for a body temperature sensor and an accelerometer. The Modulus Acceleration Monitoring and double–bump pattern algorithm is used for detecting a fall.
- SSEP, SDM and SPP are implemented in the gateway. SSEP subscribes to the sink node. SSEP sends data to both the SDM and the SPP. All payloads are in XML format. SDM parses the XML and saves ten latest data of a sensor.

### 5.4.2 The Prototype Architecture

We have used ZING as the base for the SN Server. ZING is an open source SN which is based on Apache Shindig and OpenSocial APIs [84]. We have added the M2M components; the M2M Adapter, the Data Analyzer, and the Content Adaptation Manager with ZING to enable the M2M communication. A MySQL server database is used as the SN repository. The CoAP-HTTP Proxy translates a CoAP request to an HTTP request. The payload of a CoAP message is copied as the HTTP payload. Figure 5.4 shows the prototype architecture of the M2M enabled SN. Most of the components of an SN are built-in with the ZING. We added the M2M capability on top of the existing features. In Figure 5.4, the gray boxes show the modules for the M2M. The Content Adaptation Manager only contains a text generator.

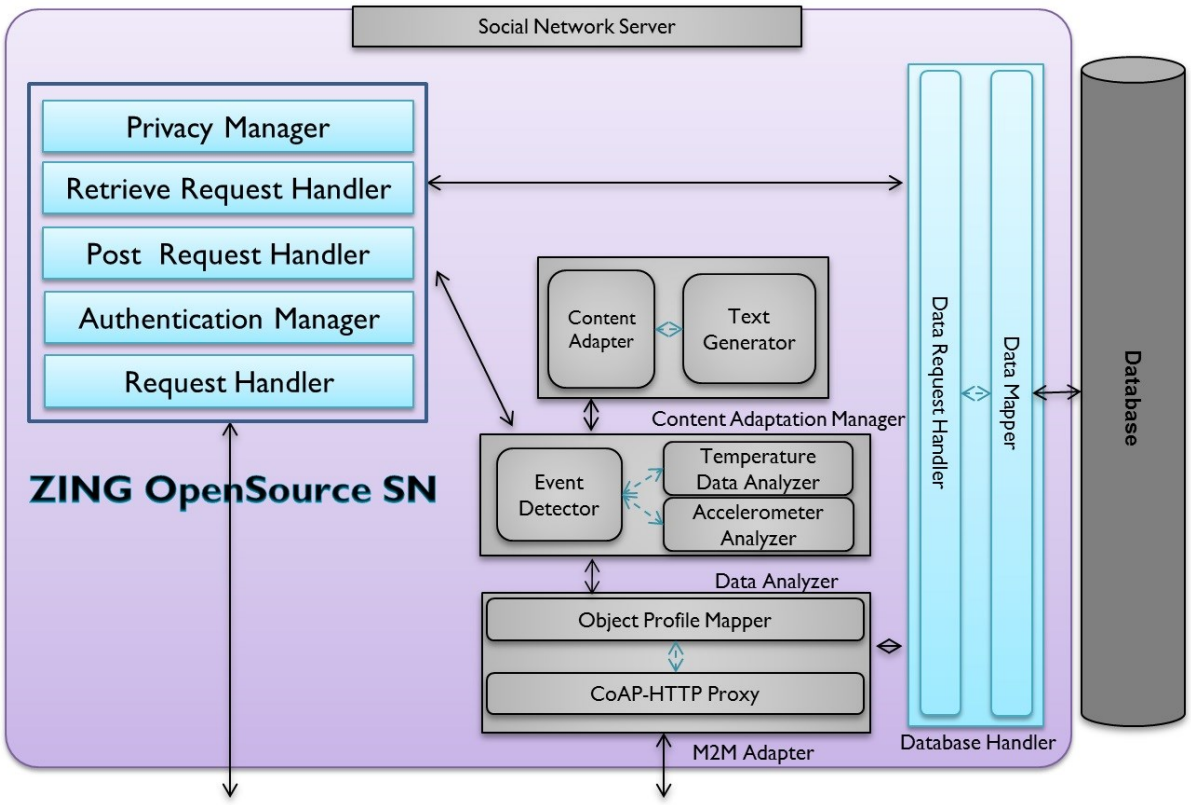


**Figure 5.4 – The prototype architecture of the SN Server**

### 5.4.3 The Fall Detection Algorithm

A fall detection algorithm uses accelerometer data to analyze the movement pattern or an activity [83]. The Modulus Acceleration Monitoring is one of the popular algorithms for fall detection. It calculates the dynamic acceleration and compares it with a threshold and sequence of events. We have used the fall detection algorithm proposed in [83], which monitors modulus acceleration and a double-bump pattern. The algorithm for calculating acceleration modulus and detecting fall is given in Figure 5.5. A candidate fall is detected by following two steps.

1) Start of fall is detected by comparing dynamic acceleration's modulus against a threshold.

2) Acceleration modulus is checked for double bump pattern for candidate's fall or false alarm.

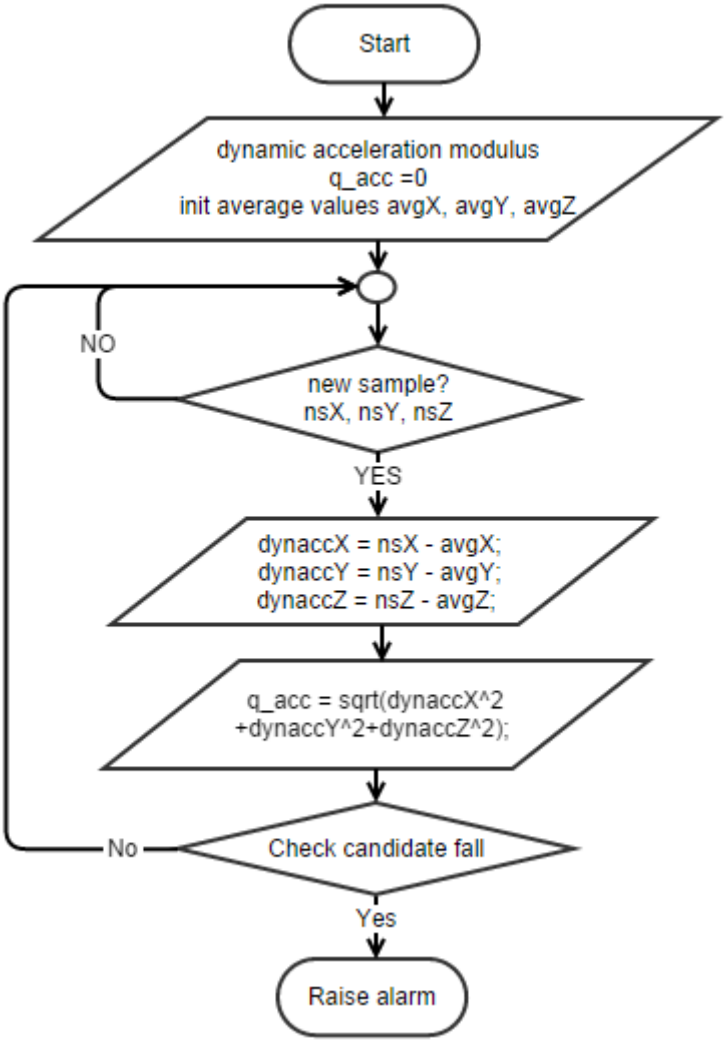


**Figure 5.5 – Detecting fall by calculating acceleration modulus**

### 5.4.4 Experimental Setup

The gateway nodes run on a laptop equipped with a 2.4 GHz processor and 6GB of RAM. A personal computer with a 2.66GHz processor and 4GB RAM serves as the SN Server. The same computer hosts a MySQL server. The computers connect to each other using a Wi-Fi network.

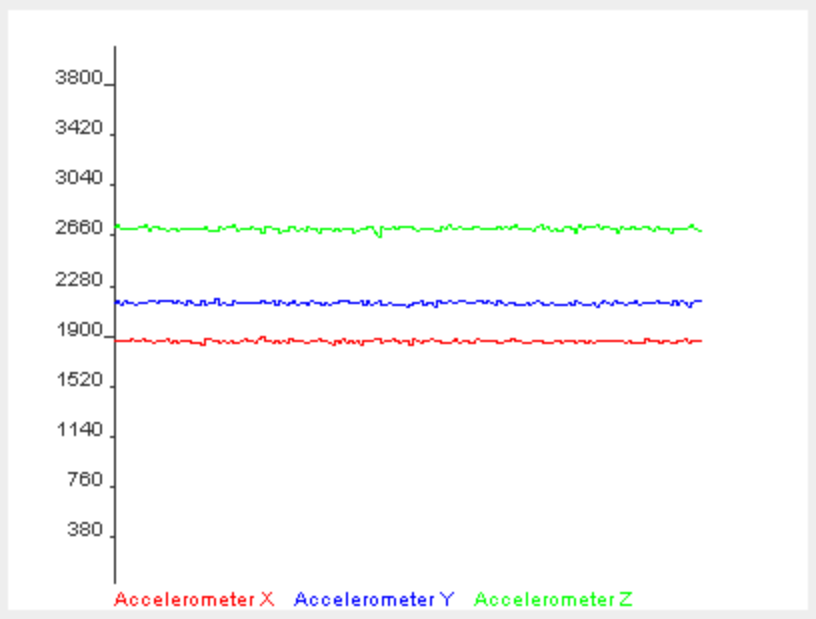


**Figure 5.6 – The graph showing raw data of the tri-axis accelerometer for no movement**

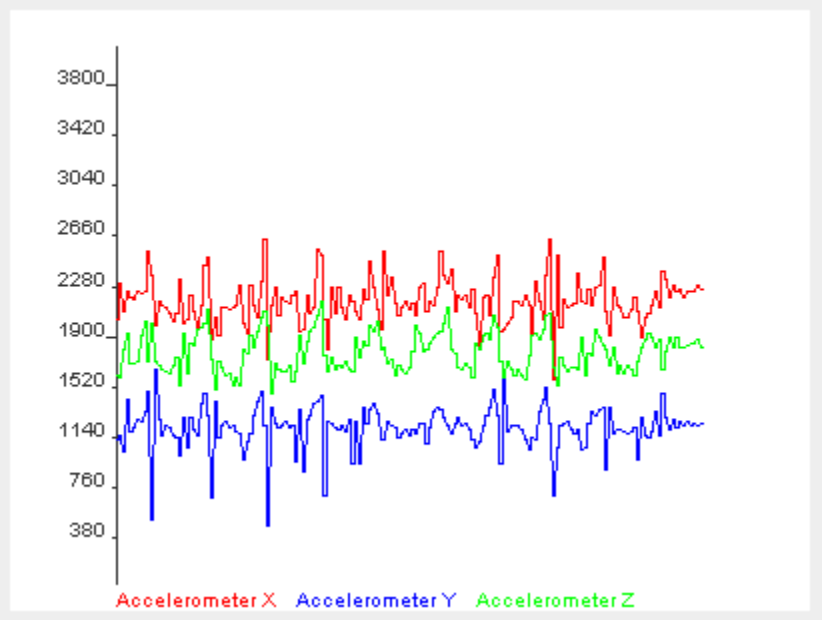


**Figure 5.7 – The raw data of the tri-axis accelerometer while walking**

We use a SHIMMER Platinum Dev Kit [83] accelerometer sensor to collect the sensory data and to realize the illustrative scenario mentioned in Section 5.3. We simulate the falls with random activities like walking, sitting and sudden rapid movements. We have implemented the Modulus Acceleration Monitoring algorithm [83] to detect rapid movements in the gateway. The

gateway forwards that data to the SN only if it finds any rapid movement in the accelerometer data. Figure 5.6, Figure 5.7 and Figure 5.8 is showing the graph of tri-axis accelerometer data for the no movement/activity, walking and a fall respectively.

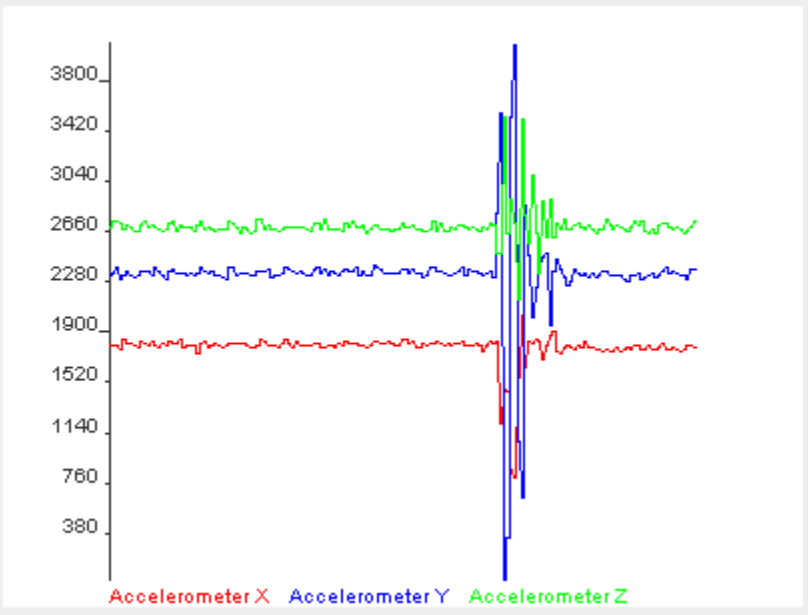


**Figure 5.8 – The graph of the tri-axis accelerometer data for a rapid fall**

We simulate a temperature sensor on a sink node. We have implemented a random function to generate values for the temperature between 35°C and 45°C. Any change of the temperature value causes the sensor to forward a message to the gateway. The gateway forwards the message to the SN if it exceeds a pre-configured threshold (in our case we have used 37.5°C for detecting fever).

We measure the performance metrics for 100 messages because a sensor only generates a message when there is a particular change in its value. For example, a temperature sensor forwards a message when the temperature changes from 37°C to 38°C. The users can login to the SN Server using a web browser from a personal computer or a smartphone to check the posts of his connections.

We have implemented our prototype on top of Zing [25]. Zing is an open-source implementation of the SN. It is based on the Apache Shindig framework and OpenSocial APIs [21]. We added the M2M functionalities to ZING. Zing uses the MySQL server as the SN repository. The Data analyzer module is a black box where various data analyzers can be integrated based on the sensor/machine-device type. For the CoAP, we have used jCoAP, which is a Java-based open source CoAP reference implementation. We have used RawCap [85] for tracing the CoAP packets. Figure 5.9 is showing the homepage of Alice, which contains the auto-generated posts of her friends.

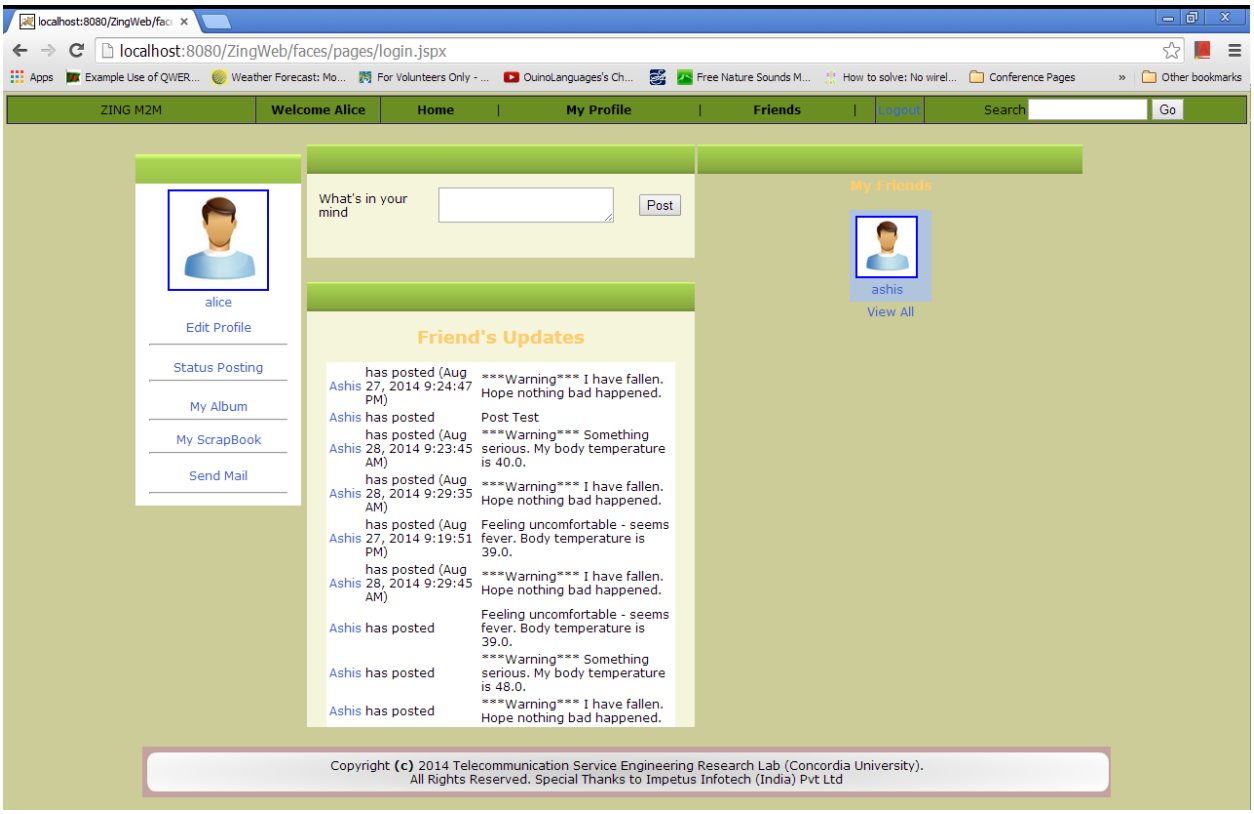


**Figure 5.9 – The homepage of the SN showing the automatic posts**

## 5.5 Performance Evaluation

Two performance metrics are selected for evaluation: delay and network load. The values are measured in previously mentioned deployed environment.

### 5.5.1 Performance Metrics

We have evaluated the performance of the prototype in terms of delay and network load.

- **Delay:** The delay is the time difference between the time when a sink sends a message and the post is created in the SN Server. This delay is measured as the time it takes for a request to be forwarded through gateway nodes, the processing time for finding a potential event and for the adaptation for the posting in the SN Server. The delay is measured in milliseconds.

- **Network Load:** The network load is measured as the number of bytes sent through the network from the sink to the gateway. We also measured the number of bytes sent through the network from the gateway to the SN Server.

### 5.5.2 Performance Results

As mentioned earlier, we have measured delay and network load for 100 messages. The average delay for posting about an event of the temperature sensor is 49.77 milliseconds. The average delay for posting about an event of the accelerometer is 115.34 milliseconds.

The network load in the gateway for 100 messages is 32498 bytes for the temperature sensor data and 589186 bytes for the accelerometer data that is on average 324.98 bytes and 5891.86 bytes respectively. The gateway forwarded 51% of messages containing the temperature sensor data. For the accelerometer, the gateway forwarded only 27% of the received messages. The gateway filters the rest of the data.

Figure 5.10 and Figure 5.11 are showing the delay for posting in the SN for the temperature sensor and accelerometer respectively. These graphs only include the delays of the messages that successfully reached the SN and the SN posts about the event.

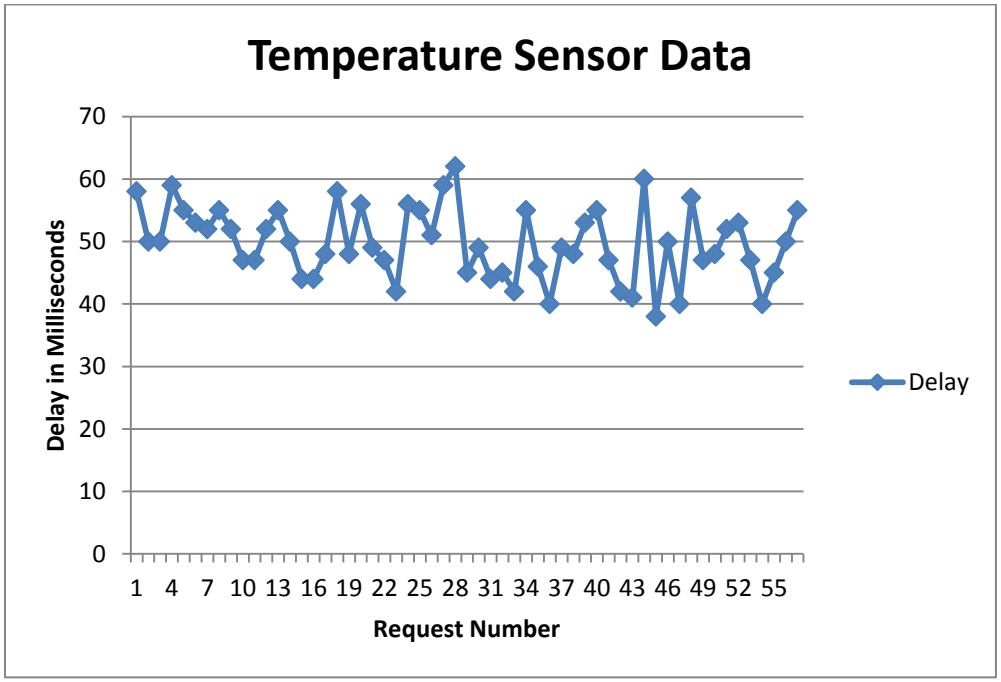


**Figure 5.10 – The delay per request for the temperature sensor data**

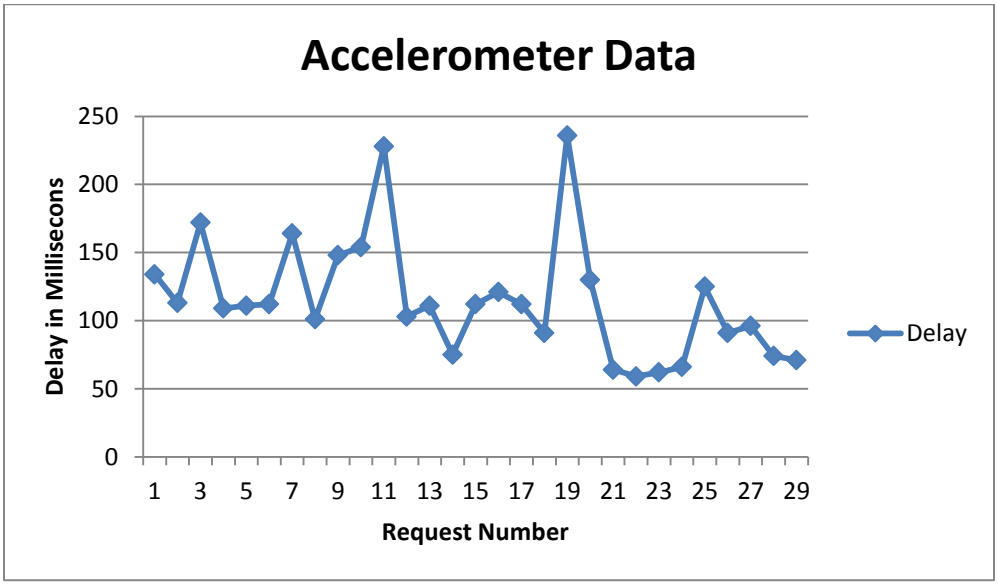


**Figure 5.11 – The delay per request for the accelerometer data**

As we can see, the average delay and the network load for the accelerometer are higher than that of temperature sensor. The reason is that the accelerometer aggregates 100 samples per message. These delays are acceptable considering the scenario. According to [86], a delay less than 500ms for the medical telemetry is considered as reasonable.

## 5.6 Chapter Summary

This chapter introduced the detailed software architecture of the SN Server and the gateway. It also presented a proof of concept prototype implementation of the proposed architecture. It discussed the experimental setup, tools and libraries we used to implement and deploy the prototype. The proof-of-concept prototype shows that the proposed SN and the overlay gateway are feasible to generate the automatic posts in the SN within an acceptable delay. Finally, the overlay gateway promotes scalability and improves the network load. More enhanced event detection algorithms and more sophisticated filtering in the gateway can improve the performance of the proposed architecture. The following chapter will summarize the thesis contribution. It will also present some additional future works.

# Chapter 6: Conclusion and Future Work

We summarize the contributions of the thesis in this chapter. We conclude the chapter with potential future work.

## 6.1 Summary of Contributions

The number of the M2M devices around us is rapidly increasing. Similarly, people are getting involved on the internet and SNs more often. The numbers of users in various SNs are also growing. The M2M communication is not yet widely considered to be used to ease the sharing experience in SNs. This thesis has introduced an architecture to enable the M2M communication to generate the automatic posts for the SNs. The M2M generates the information, and the SN disseminates the information among the users COIs.

In this thesis, we stated the research domain with the problem statement and motivating scenarios. We then derived a set of requirements for the automated information sharing in the SN. We evaluated the existing work and found that no solutions satisfy all of the requirements. We proposed an architecture to enable M2M communication in SNs. The architecture is based on an overlay-based gateway to handle M2M devices and uses CoAP for the M2M communication. As a result, the gateway makes the architecture scalable with respect to the number of M2M devices. The proposed SN architecture is an extension to OpenSocial to support M2M communication. Furthermore, we validated the proposed architecture and the gateway against the derived requirements.

Finally, we implemented a proof of concept by extending the OpenSocial based Zing SN to demonstrate the feasibility of the proposed architecture. Then, we partially evaluated the

performance of the prototype for two performance metrics. We found that the gateway filters a good amount of data and improves the network load. There is a delay on the gateway for processing M2M data. However, performance results show that the delay is reasonable for medical telemetry. These performances can be increased by enhancing the event detection algorithms and more sophisticated filtering on the gateway.

## 6.2 Future Work

As a future improvement to this proposed architecture, we can consider to extend it in different directions. First, in this architecture we have not considered the security of the M2M devices. The gateway peers that we have proposed are accessible without any security restriction. Most M2M data are sensitive and must be handled carefully. There are private data that users might want to protect from outsiders. Personal health statistics or data also should be only available to appropriate users. In this architecture, the CoAP clients can access the CoAP server modules without any authentication. We should consider including the CoAP security in these gateway peers. Second, providing quality of service to the users of the SN as day by day the number of users and number of the M2M devices are adding up. Users might want some service priority over other users. Similarly, some users might want to pay for a reliable service. Third, we only considered the sharing of some textual information. Other types of information can be supported by the SN adaptation module and can be integrated in future. For example, a surveillance system can take a photo of an area after certain time and can have an access history of the location. A personal camera can posts photos with location information and makes an album based on that. An IP camera can stream a video to certain interested users. There can be a video conference for a group of people using their cellphone camera. There should be a plugin enhancement for different kinds of data type that might be coming to the SN Server.

We have assumed that the M2M data is in XML format. We have used a custom defined format for the XML data. There should be some work for adoption of an M2M standard data format for extensive use of the system and generalize it for all M2M scenarios.

# Bibliography


[1] G. Wu, S. Talwar, and K. Johnsson, "M2M: From mobile to embedded internet," *IEEE Commun. Mag.*, vol. 49, no. April, pp. 36–43, 2011.

[2] M. J. Booysen, J. Gilmore, S. Zeadally, and G. Van Rooyen, "Machine-to-machine (m2m) communications in vehicular networks," *KSII Trans. Internet Inf. Syst.*, vol. 6, no. 2, pp. 1–21, 2012.

[3] D. Niyato, L. Xiao, and P. Wang, "Machine-to-machine communications for home energy management system in smart grid," *Commun. Mag. IEEE*, vol. 49, no. 4, pp. 53–59, 2011.

[4] Z. Fan and S. Tan, "M2M communications for E-health: Standards, enabling technologies, and research challenges," in *2012 6th International Symposium on Medical Information and Communication Technology (ISMICT)*, 2012, pp. 1–4.

[5] "Facebook." [Online]. Available: www.facebook.com. [Accessed: 10-Sep-2014].

[6] "Twitter." [Online]. Available: twitter.com. [Accessed: 05-Sep-2014].

[7] "Google+." [Online]. Available: plus.google.com. [Accessed: 05-Sep-2014].

[8] M. B. N. B. E. Danah, "Social network sites: Definition, history, and scholarship," *J. Comput. Commun.*, vol. 13, no. 1, pp. 210–230, Oct. 2007.

[9] "Reddit." [Online]. Available: www.reddit.com. [Accessed: 05-Sep-2014].

[10] "LinkedIn." [Online]. Available: www.linkedin.com. [Accessed: 05-Sep-2014].

[11] Pew Research Center's Internet Project, "Social Media Update 2013," 2014. [Online]. Available: http://www.pewinternet.org/2013/12/30/social-media-update-2013/. [Accessed: 20-Aug-2014].

[12] M. A. Rahman, A. El Saddik, and W. Gueaieb, "Senseface: A sensor network overlay for social networks," in *IEEE Instrumentation and Measurement Technology Conference, 2009. I2MTC '09*, 2009, no. May, pp. 5–7.

[13] A. J. Jara, M. A. Zamora, and A. F. G. Skarmeta, "An architecture based on internet of things to support mobility and security in medical environments," in *7th IEEE Consumer Communications and Networking Conference (CCNC)*, 2010, pp. 1–5.

[14] M. Burke, C. Marlow, and T. Lento, “Feed me: motivating newcomer contribution in social network sites,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2009, pp. 945–954.

[15] “livelens.” [Online]. Available: www.livelens.com. [Accessed: 01-Dec-2014].

[16] “Google Glass.” [Online]. Available: https://www.google.ca/glass/start/. [Accessed: 17-Nov-2014].

[17] M. C. Domingo, “A Context-Aware Service Architecture for the Integration of Body Sensor Networks and Social Networks through the IP Multimedia Subsystem,” *IEEE Commun. Mag.*, vol. 49, no. 1, pp. 102–108, 2011.

[18] A. Osseiran, F. Boccardi, V. Braun, K. Kusume, P. Marsch, M. Maternia, O. Queseth, M. Schellmann, H. Schotten, H. Taoka, H. Tullberg, M. A. Uusitalo, B. Timus, and M. Fallgren, “Scenarios for 5G mobile and wireless communications: The vision of the METIS project,” *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 26–35, 2014.

[19] Z. Shelby, K. Hartke, and C. Bormann, “Constrained Application Protocol (CoAP),” *IETF RFC 7252*. [Online]. Available: https://tools.ietf.org/html/rfc7252.

[20] Google and MySpace, “OpenSocial API.” [Online]. Available: http://opensocial.org/. [Accessed: 05-May-2014].

[21] E. K. L. E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, “A survey and comparison of peer-to-peer overlay network schemes,” *IEEE Commun. Surv. Tutorials*, vol. 7, no. 2, 2005.

[22] R. T. Fielding, “Architectural Styles and the Design of Network-based Software Architectures,” 2000.

[23] L. Richardson and S. Ruby, *RESTful web services*. 2008, p. 440.

[24] I. Jung, H. Kim, D.-K. Hong, and H. Ju, “Protocol Reverse Engineering to Facebook Messages,” in *4th International Conference on Intelligent Systems, Modelling and Simulation*, 2013, pp. 539–542.

[25] “SixDegrees.com.” [Online]. Available: http://en.wikipedia.org/wiki/SixDegrees.com. [Accessed: 12-Aug-2014].

[26] “AIM.” [Online]. Available: www.aim.com. [Accessed: 12-Aug-2014].

[27] “ICQ.” [Online]. Available: www.icq.com. [Accessed: 12-Aug-2014].

[28] “Asian Avenue.” [Online]. Available: www.asianave.com. [Accessed: 12-Aug-2014].

[29] “Black Planet.” [Online]. Available: BlackPlanet.com. [Accessed: 12-Aug-2014].

[30] “MiGente.” [Online]. Available: www.migente.com. [Accessed: 12-Aug-2014].

[31] “LiveJournal.” [Online]. Available: www.livejournal.com. [Accessed: 12-Aug-2014].

[32] “Ryze.com.” [Online]. Available: ryze.com. [Accessed: 12-Aug-2014].

[33] “Tribe.net.” [Online]. Available: tribe.net. [Accessed: 12-Aug-2014].

[34] “Friendster.” [Online]. Available: www.friendster.com. [Accessed: 12-Aug-2014].

[35] “Match.com.” [Online]. Available: www.match.com. [Accessed: 12-Aug-2014].

[36] “Flickr.” [Online]. Available: www.flickr.com. [Accessed: 12-Aug-2014].

[37] “Last.fm.” [Online]. Available: www.last.fm. [Accessed: 12-Aug-2014].

[38] “YouTube.” [Online]. Available: www.youtube.com. [Accessed: 12-Aug-2014].

[39] “MySpace.” [Online]. Available: www.myspace.com. [Accessed: 12-Aug-2014].

[40] C. Kadushin, *Understanding social networks: Theories, concepts, and findings*, vol. 42, no. 2. Oxford University Press, 2012, pp. 249–251.

[41] A. Papadimitriou, D. Katsaros, and Y. Manolopoulos, “Social Network Analysis and Its Applications in Wireless Sensor and Vehicular Networks,” *Next Gener. Soc. Technol. Leg. Issues, Lect. Notes Inst. Comput. Sci. Soc. Informatics Telecommun. Eng.*, vol. 26, pp. 411–420, 2010.

[42] I. Lequerica, M. G. Longaron, and P. M. Ruiz, “Drive and share: efficient provisioning of social networks in vehicular scenarios,” *IEEE Commun. Mag.*, vol. 48, no. 11, pp. 90–97, 2010.

[43] “Cure Together.” [Online]. Available: curetogether.com. [Accessed: 12-Aug-2014].

[44] “PatientsLikeMe.” [Online]. Available: www.patientslikeme.com. [Accessed: 12-Aug-2014].

[45] S. Mohan and N. Agarwal, “A convergent framework for QoS-driven social media content delivery over mobile networks,” in *Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE), 2011 2nd International Conference on*, 2011, pp. 1–7.

[46] T. Reynaert and W. De Groef, “PESAP: a Privacy Enhanced Social Application Platform,” in *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference*

*on and 2012 International Confernece on Social Computing (SocialCom)*, 2012, pp. 827–833.

[47] "hi5." [Online]. Available: www.hi5.com. [Accessed: 12-Aug-2014].

[48] J. Goldman, *Facebook Cookbook*, First Edit. Published by O'Reilly Media, 2009, p. 405.

[49] M. Häsel, "Opensocial: an enabler for social applications on the web," *Commun. ACM*, vol. 54, no. 1, p. 139, Jan. 2011.

[50] "Introducing JSON." [Online]. Available: www.json.org. [Accessed: 18-Oct-2014].

[51] "Extensible Markup Language (XML) 1.0 (Fifth Edition)." [Online]. Available: http://www.w3.org/TR/REC-xml/. [Accessed: 18-Oct-2014].

[52] "Atom," *RFC*. [Online]. Available: https://tools.ietf.org/html/rfc4287. [Accessed: 18-Oct-2014].

[53] B. Leiba, "Oauth web authorization protocol," *IEEE Internet Comput.*, vol. 16, no. 1, pp. 74–77, 2012.

[54] The Apache Software Foundation, "Apache Shindig." [Online]. Available: http://shindig.apache.org/. [Accessed: 21-Jul-2014].

[55] R. H. Glitho, "Application architectures for machine to machine communications: Research agenda vs. state-of-the art," *7th Int. Conf. Broadband Commun. Biomed. Appl.*, pp. 1–5, Nov. 2011.

[56] "European Telecommunications Standards Institute." [Online]. Available: http://www.etsi.org/. [Accessed: 10-Sep-2014].

[57] "Internet Engineering Task Force." [Online]. Available: https://www.ietf.org/. [Accessed: 23-Sep-2014].

[58] "3rd Generation Partnership Project." [Online]. Available: http://www.3gpp.org/. [Accessed: 10-Sep-2014].

[59] ETSI, "Machine-to-Machine communications (M2M); M2M service requirements," 2010. [Online]. Available: http://www.etsi.org/deliver/etsi_ts/102600_102699/102689/01.01.01_60/ts_102689v010101p.pdf. [Accessed: 05-Aug-2014].

[60] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup protocol for Internet applications," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 17–32, 2003.

[61] B. Cohen, “BitTorrent.” [Online]. Available: http://bittorrent.org/. [Accessed: 10-Sep-2014].

[62] F. Belqasmi, R. Glitho, and C. Fu, “RESTful web services for service provisioning in next-generation networks: a survey,” *Commun. Mag. IEEE*, vol. 49, no. 12, pp. 66–73, 2011.

[63] “WADL.” [Online]. Available: http://www.w3.org/Submission/2009/SUBM-wadl-20090831/. [Accessed: 15-Jul-2014].

[64] “CoRE.” [Online]. Available: https://datatracker.ietf.org/wg/core/. [Accessed: 12-Aug-2014].

[65] W. Colitti, K. Steenhaut, N. De Caro, B. Buta, and V. Dobrota, “Evaluation of constrained application protocol for wireless sensor networks,” in *18th IEEE Workshop on18th IEEE Workshop on Local & Metropolitan Area Networks (LANMAN)*, 2011, pp. 1–6.

[66] RFC, “User Datagram Protocol.” [Online]. Available: https://www.ietf.org/rfc/rfc768.txt. [Accessed: 18-Oct-2014].

[67] K. Kuladinithi, O. Bergmann, T. Pötsch, M. Becker, and C. Görg, “Implementation of coap and its application in transport logistics,” in *In Proceedings of the Workshop on Extending the Internet to Low power and Lossy Networks*, 2011, vol. April.

[68] R. P. J. Preethi Rajasekaran and R. P. V. Chander, “A smarter toll gate based on Web of Things,” in *IEEE International Conference on Electronics, Computing and Communication Technologies*, 2013, pp. 1–6.

[69] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design patterns: elements of reusable object-oriented software*, vol. 206. 1994, p. 395.

[70] K. Hartke, “Observing Resources in CoAP,” *Internet-Draft*. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-core-observe/. [Accessed: 23-Sep-2014].

[71] J. Breslin, S. Decker, M. Hauswirth, and G. Hynes, “Integrating social networks and sensor networks,” *W3C Work. Futur. Soc. Netw.*, no. January, pp. 15–16, 2009.

[72] M. Kranz, L. Roalter, and F. Michahelles, “Things that twitter: social networks and the internet of things,” *What can Internet Things do Citiz. Work. Eighth Int. Conf. Pervasive Comput. (Pervasive 2010)*, 2010.

[73] M. A. Rahman, A. El Saddik, and W. Gueaieb, “Data visualization: From body sensor network to social networks,” in *IEEE International Workshop on Robotic and Sensors Environments, 2009. ROSE 2009.*, 2009, pp. 157 – 162.

[74] K. Knightson, N. Morita, and T. Towle, “NGN architecture: Generic principles, functional architecture, and implementation,” *IEEE Commun. Mag.*, vol. 43, no. 10, pp. 49–56, 2005.

[75] A. Triantafyllidis, “A pervasive health system integrating patient monitoring, status logging, and social sharing,” *IEEE J. Biomed. Heal. Informatics*, vol. 17, no. 1, pp. 30–37, 2013.

[76] The World Wide Web Consortium (W3C), “SOAP.” [Online]. Available: http://www.w3.org/TR/soap/. [Accessed: 10-Jun-2014].

[77] M. Pulgarin, R. Glitho, and A. Quintero, “An Overlay Gateway for the Integration of IP Multimedia Subsystem and Mobile Sink Based-Wireless Sensor Networks,” in *IEEE 72nd Vehicular Technology Conference Fall (VTC 2010-Fall)*, 2010, pp. 1 – 5.

[78] M. Aly and A. Quintero, “A presence-based architecture for a gateway to integrate vehicular ad-hoc networks (VANETs), IP multimedia subsystems (IMS) and wireless sensor networks (WSNs),” in *9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2013, pp. 1648–1653.

[79] A. Muhammad and P. Fergus, “Peer-to-Peer Overlay Gateway Services for Home Automation and Management,” in *IEEE 24th International Conference on Advanced Information Networking and Applications Workshops*, 2010, pp. 880–886.

[80] a. Muhammad, M. Merabti, and B. Askwith, “E-System: Package Delivery Framework,” *2009 Second Int. Conf. Dev. eSystems Eng.*, pp. 196–201, Dec. 2009.

[81] C. Fu, R. Glitho, and F. Khendek, “A Novel Session Recovery Mechanism for Cluster-based Signaling Architecture for Conferencing in MANETs,” in *27th International Conference on Distributed Computing Systems Workshops (ICDCSW’07)*, 2007, p. 19.

[82] W. Colitti, K. Steenhaut, and N. De Caro, “Integrating wireless sensor networks with the web,” in *Extending the Internet to Low powerand Lossy Networks (IP+SN 2011)*, 2011, pp. 2–6.

[83] F. Stroiescu, K. Daly, and B. Kuris, “Event detection in an assisted living environment,” in *Annual International Conference of the IEEE, Engineering in Medicine and Biology Society, EMBC*, 2011, vol. 2011, pp. 7581 – 7584.

[84] Impetus Labs, “Zing.” [Online]. Available: https://code.google.com/p/zing/. [Accessed: 05-May-2014].

[85] NETRESEC AB, “RawCap.” [Online]. Available: http://www.netresec.com/?page=RawCap. [Accessed: 08-Aug-2014].

[86] J. Park, J. Lee, and K. Kang, “Designing and predicting QoS of a wireless system for medical telemetry,” *IEEE 37th Conf. Local Comput. Networks Work. (LCN Work.*, pp. 737–744, Oct. 2012.