IntechOpen

Mobile Computing
Technology and Applications

Edited by Mutamed Khatib and Nael Salman

MOBILE COMPUTING - TECHNOLOGY AND APPLICATIONS

Edited by **Mutamed Khatib**
and **Nael Salman**

Mobile Computing - Technology and Applications
http://dx.doi.org/10.5772/intechopen.70979
Edited by Mutamed Khatib and Nael Salman

Contributors

Justice Opara-Martins, Matthew Liotine, Amr Elsayed, El-Sayed M. El-Horbaty, Tamer Abdelkader, Feng Liu, Ziyou Gao, Bin Jia, Davy Janssens, Geert Wets, Feda Alshahwan, Eui-Nam Huh, Tien-Dung Nguyen, Yunkon Kim, Xuan-Qui Pham, Tri Nguyen

Notice

Statements and opinions expressed in the chapters are these of the individual contributors and not necessarily those of the editors or publisher. No responsibility is accepted for the accuracy of information contained in the published chapters. The publisher assumes no responsibility for any damage or injury to persons or property arising out of the use of any materials, instructions, methods or ideas contained in the book.

First published in London, United Kingdom, 2018 by IntechOpen
eBook (PDF) Published by IntechOpen, 2019
IntechOpen is the global imprint of INTECHOPEN LIMITED, registered in England and Wales, registration number: 11086078, The Shard, 25th floor, 32 London Bridge Street
London, SE19SG – United Kingdom
Printed in Croatia

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

Additional hard and PDF copies can be obtained from orders@intechopen.com

Mobile Computing - Technology and Applications
Edited by Mutamed Khatib and Nael Salman

p. cm.

Print ISBN 978-1-78923-222-6

Online ISBN 978-1-78923-223-3

eBook (PDF) ISBN 978-1-83881-504-2

Meet the editors

Mutamed Khatib received his BSc degree in Telecommunication Engineering from Yarmouk University, Irbid, Jordan, in 1996, and MSc degree in Electrical and Electronic Engineering from Jordan University for Science and Technology, Irbid, Jordan, in 2003. He received his PhD degree in Wireless and Mobile Systems from Universiti Sains Malaysia (USM), Malaysia, in 2009. His research interests are in the field of mobile networks and coding such as broadband access network and services, adaptive signal processing for wireless communications, and cellular networks. From 1996 to 2005, he worked as transmission, outside broadcasting and studio engineer in Palestinian Broadcasting Corporation (PBC). In 2005, he worked as an instructor in the Engineering Faculty at Palestine Technical University (Kadoorie), Tulkarm, Palestine. He was the head of telecommunication department at the same university for 2 years, was the dean of faculty of engineering and technology there, and is now the vice president for academic affairs, in which he is teaching advanced courses in telecommunications and coding as an associate professor. Dr. Khatib has a number of publications to his credit in various international journals and conference proceedings. He is the editor of a book titled *Advanced Trends in Wireless Communications and Contemporary Issues in Wireless Communications* published by IntechOpen Open Access Publisher from Croatia in 2011 and 2014, respectively, and he is the author of two scientific books and two book chapters. Dr. Khatib is an editor and reviewer for many international journals in his field and Palestinian Engineers Association.

Dr. Nael Salman received his PhD degree in Computer Engineering from Middle East Technical University (METU), Ankara, Turkey, in 2006. He is an assistant professor of computer engineering at the Department of Computer Systems Engineering at Palestine Technical University-Kadoorie (PTUK). Besides his job as a lecturer, Dr. Salman served in several administrative positions: the head of Electrical and Computer Systems Engineering Department, the Computer Center director, dean of Community service and Development affairs, and dean of Admission and registration at PTUK. Dr. Salman started his job as a lecturer at PTUK in 1996. From 2000 to 2007, he worked as an instructor in the Department of Computer Engineering at Cankaya University, Ankara, Turkey. In January 2007, he rejoined PTUK where he is still working. Dr. Salman has a number of publications in various international journals and refereed conference proceedings. He is also a program committee member of Software Engineering Conference organized annually in Turkey. Dr. Salman also participated as an IT consultant in several Tempus, QIF-funded projects. His research interests include the fields of software engineering, software measurement, web application development, network security, and database systems.

Contents

Preface

The rapid demand of wireless mobile communication technology has produced capabilities on researches to introduce software systems that satisfy the needs of users. The affordable price of mobile devices such as smart phones and tablets and the huge number of mobile applications are increasing the number of users for mobile technology day by day and forced researchers to work harder to introduce secured, fast, and easy-to-use applications.

The resulting computing platform, which is often referred to as mobile computing, no longer requires users to stay in a fixed location in the network and enables them to unrestricted mobility. Portability creates an entire new era of applications and, possibly, new great markets for both personal computing and consumer electronic systems. In the new technologies, information is easily accessible from virtually any place and time, and it is stored in a highly decentralized, distributed information infrastructure.

Under these conditions, mobile computing cannot be considered as "scaled-down" version of the established field of distributed computing. The nature of wireless communication media and the mobility of computers combine to create fundamentally new problems in networking, operating systems, and information systems. Furthermore, many of the applications envisioned for mobile computing place novel demands on software systems.

In this book, latest trends in mobile computing will be discussed. In the first section, cloud computing topics will be discussed widely in three chapters to give information to the reader about topics such as challenges, services, edge computing, and distributed clouds needed to integrate this promising issue into the next generation.

We would like to thank the authors who participated in this book and put the latest research in our hand in order to move a step forward toward new and modern mobile computing technologies. Also, we would like to express our sincere thanks to IntechOpen publisher for the opportunity they gave us to edit this valuable book.

Dr. Mutamed Khatib, Editor and Dr. Nael Salman, Co-Editor
Palestine Technical University Kadoorie
Tulkarm, Palestine

Section 1

Cloud Computing

Chapter 1

Taxonomy of Cloud Lock-in Challenges

Justice Opara-Martins

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.74459

Abstract

This chapter reviews key concepts and terminologies needed for understanding the complexity of the vendor lock-in problem being investigated in this book. Firstly, we present aspects of cloud computing that contribute to vendor lock-in and briefly introduce existing results from cloud-related areas of computer science that contributes to understanding and tackling vendor lock-in. Secondly, we explore the literature on proprietary lock-in risks in cloud computing environments to identify its causes (i.e., restrictions), consequences, mitigations strategies, and related challenges faced by enterprise consumers migrating to cloud-based services. Then, we propose taxonomy of cloud lock-in perspectives based on reports of real experiences on migration to understand the overall cloud SaaS migration challenges. Finally, we narrow down to our perspective on cloud lock-in to three main perspectives which takes the use of sound techniques from IS research discipline and cloud-related literature into consideration, to improve the portability, security and interoperability of cloud (and on-premise) applications in hybrid environments. Collectively, the discussions presented herein, accordingly enables both academia and IT practitioners in the cloud computing community to get an overarching view of the process of combating application and data lock-in challenges, and security risks in the cloud.

Keywords: cloud computing, taxonomy, vendor lock-in, SaaS migration, ICT services

1. Introduction

Cloud computing has been revolutionizing the IT industry by adding flexibility to the way Information and Communication Technology (ICT) services is consumed, enabling organizations to pay only for the resources and services they use [1]. In an effort to reduce IT capital and operational expenditures (OpEx), organizations of all sizes are using clouds to provide the resources required to run their applications. Clouds vary significantly in their specific technologies and implementation, but often provide infrastructure, platform, and software resources

as services [2, 3]. Vendor lock-in problem is a highly referenced topic followed by security in the technical and business IT world [4–8]. Research shows that several companies are already migrating to cloud-based services, but challenges of vendor lock-in is prohibiting a widespread adoption rate across enterprises of different sizes and industry sectors [9–11]. Simply put, businesses are wary of being tied to particular cloud computing vendors due to lack of competing, compatible product [12]. Due to the ever growing interest in cloud computing services, there is an explicit and constant effort to evaluate the current trends in vendor lock-in and security for such technology. Progress of current research effort in demystifying the complexity of vendor lock-in problem in cloud computing technology is contingent on having a rigorous organization of its knowledge domain and a comprehensive understanding of all the relevant elements and components of this technology and their relationship. Currently, there is a lack of sufficient standardization of cloud computing services [4–6, 13], and each cloud service vendor uses different technologies, SLAs, protocols, APIs, security protocols, and formats. This further makes interoperability, portability, and security as elusive tasks to accomplish when working with multiple (hybrid cloud) services. Moreover, with the amount of cloud computing services and vendors increasing quickly, the need for a detailed taxonomy framework rises [14]. Such taxonomy should provide a common terminology and baseline information for easy communication and understandability of vendor lock-in risks when comparing the offers to find the right cloud service/vendor of choice. Therefore, the vast amount of cloud computing services and the lack of universal definitions and standards agreement to avoid lock-in lead to the question whether cloud computing services and related challenges can be classified in a taxonomy based on their characteristics to easily compare them. While the work in [15] proposes taxonomy for a quick classification of cloud services making it easier to compare them, however, it does not specifically addresses the challenges of vendor lock-in affecting widespread cloud adoption and migration.

To this end, the main goal of this chapter is to identify, classify, and organize the main risks of vendor lock-in associated to cloud computing migration, helping in the task of underlying the challenges that remain unanswered from the enterprise perspective. This taxonomy demonstrates a dissection of vendor lock-in problem into three main perspectives, and illustrates their classification and associated elements. It makes it very easy to comprehend the cloud computing vendor lock-in field as a whole, correlate, classify, and compare the various existing industry solution proposals, mitigation strategies, and best practices. In turn, this will provide a solid foundation for future research analysis, and a review to assist academia, and a scientific research community to expedite its contributions and insights regarding lock-in avoidance for enterprise cloud migration use cases. Over the years, taxonomy techniques have been used to create models that allow for the classification of concepts within a domain. In this chapter, we apply taxonomy techniques in the cloud computing domain to discuss the many aspects involved with cloud computing lock-in that are important from an enterprise perspective. Aiming at a better understanding of the categories of applications, data, and security controls that could trigger a lock-in situation during cloud migration, this chapter contributes by proposing a detailed taxonomy based on characteristics that are fundamental for enterprise production applications typically associated with the cloud and big data paradigm.

The remainder of this chapter is organized as follows. Section 2 discusses and reviews related work that is relevant to our work. Section 3 presents the current service models of cloud computing and their specific lock-in challenges. Section 4 briefly highlights an ontology of cloud lock-in perspectives which gives a better understanding and definitions of categories that is to be used in the design of our work. Section 5 presents the classification of perspectives used in our proposed taxonomy of cloud lock-in challenges. Section 6 concludes this chapter and recommends future research directions.

2. Related work

Several taxonomies for cloud computing can be found in [16, 17], but most are created from the perspective of vendors that are part of the market landscape and not from the perspective of enterprise IT, the consumers of cloud services, and software. Providing taxonomic information is essential not only for cloud service providers, but also for enterprise firms, and compliance authorities to detect, manage, and control invasive proprietary components. Taxonomy identifies and enumerates the components of cloud computing that are providing basic knowledge underpinning management and implementation of the cloud computing. There is, however, no standard taxonomy, as everyone tries to define cloud computing and its services in their own way. The taxonomy described by the Cloud Computing Use Case Discussion Group [18] is categorized into three views: service developer, service provider, and service consumers. This taxonomy does not cover the potential challenges of vendor lock-in and related security risks. Crandell [19] defines a taxonomy based on cloud service offerings divided into three layers, namely application services (e.g., Salesforce CRM and other SaaS vendors), platform services (e.g., GAE, Moso, and Heroku), and infrastructure services (e.g., Amazon Web Services, Flexiscale). This taxonomy is valuable for any company with an application that runs in a data center or with a hosted provider that does not want to reinvent the wheel or pay a premium. Laird's [20] cloud vendor taxonomy gives the classifications and vendors with their related group. This taxonomy divides the cloud vendors into infrastructure (i.e., public cloud and private cloud), platform (e.g., business user platforms and DevOps platform), services (billing, security, fabric management, and system integrators), and applications. This taxonomy gives a visual map of the SaaS, PaaS, and cloud computing industries. At the end of the spectrum, Forrester's cloud taxonomy [21] is categorizing cloud services by IT infrastructure vs. business value and by the level of privacy offered. This taxonomy focuses on the dimensions of privacy and business value. It does not address vendor lock-in issues but instead focuses on the modes of cloud computing (public scale-out clouds, public server cloud, virtual private scale-out clouds, virtual private server clouds, private clouds, virtual private SaaS, public SaaS, PaaS, on-premises, ASP concepts, etc.) To provide an even clearer and more explicit perspective on the complexity of cloud computing vendor lock problem, we propose taxonomy of cloud lock-in challenges building on several incremental enhancements of existing taxonomies. In this chapter, we will adjust, refine, and extend those taxonomies, making them even more suitable and flexible for understanding the complexity of cloud vendor lock-in problem.

Few studies recently, articles, have attempted to establish a similar structure for cloud computing and its components [22–25]. Although they attain some valuable understanding of several cloud services and components, they tend to be more general classifications without specifics about vendor lock-in. They neither went to the level of detail in the analysis as we did (in Section 4 and 5), nor they included all the cloud layer attributes we captured in our taxonomy. Their main objective is to classify the commercial cloud offerings in order to analyze the cloud computing market opportunities. As such, they do not address the specific lock-in potentials or limitations of the several cloud layers, nor the research opportunities associated with each cloud layer. We believe our proposed cloud ontology is more comprehensive, and encompass more detailed analysis of the cloud computing knowledge domain.

Providing lock-in taxonomy of unified and holistic SaaS architecture has not been addressed yet in a way comparable to the approach proposed in this chapter. Pursuing this further, SaaS architectures have been compared and analyzed in several studies [26–28]. Mahjoub et al. [29] conducted a survey on current cloud providers and technologies in order to help users choosing the better cloud offer that compiles with their needs or building their own cloud infrastructure with the most suitable open source technologies. The characteristics, architectures, and applications of several popular cloud computing platforms are analyzed and discussed in the paper by Peng et al. [30]. Wind [31] derived criteria from the literature analysis and compares existing open-source Cloud Computing management platforms. Important high-level criteria are taken into account (i.e., security, interoperability, user interface, etc.). Notwithstanding, cloud computing taxonomies have already been defined in several works [32–38]. The National Institute of Standards and Technology (NIST) [34], Rimal et al. [37], and Intel [38] presented a general cloud computing taxonomy. In contrast to our taxonomy, they did not address vital SaaS capabilities, such as data lock-in, contract lock-in, and application lock-in. Furthermore, Forrester [39] introduced a market-oriented taxonomy of cloud computing and did not incorporate technical capabilities of SaaS platforms and applications as discussed in our work. OpenCrowd [37] presented a general cloud computing taxonomy. The taxonomy only addresses IaaS layer and identifies four components: storage, compute, services management, and cloud broker. In contrast, our taxonomy addresses 11 components (relevant to IaaS, PaaS, SaaS, XaaS) and categorizes them into ordered layers.

3. Service models and vendor lock-in risks

Currently, there is little on offer in the way of tools, procedures or standard data formats or service(s) interfaces that could guarantee data and service portability in the cloud computing environment. This makes it extremely difficult for a customer to switch cloud providers, or to move data and services from an in-house IT environment to the cloud. In effect, this potential dependency for service provision on a single cloud provider, may lead to organizational risks should the cloud provider, for instance, go out-of-business or bankrupt. Organizations considering adopting cloud computing models are concerned about the potential for lock in and the operational challenges that a storage migration (as an example) would require. Thus, it becomes important to understand that the extent and nature of lock-in varies per cloud type:

- SaaS lock-in: SaaS providers typically develop a custom application tailored to the needs of their target market. The consumer data of a SaaS product is typically stored in a custom database schema designed by the SaaS provider. However, if the provider does not offer readymade data export functionality, the customer will need to develop a program to extract their data and write it to a file ready for import to another provider. Where the customer has developed programs to interact with the provider's API directly (e.g., for integration with other applications), these will also need to be re-written to consider the new provider's API. SaaS suffers from data lock-in, contract lock-in, and application lock-in risks.
- PaaS lock-in: occurs at both the API layer and at the component level. At the API layer, PaaS lock-in occurs as different providers offer different APIs. PaaS lock-in happens at the component (i.e., runtime) layer as standard runtime environments are often heavily customized to operate safely in a specific cloud environment. PaaS suffers from framework lock-in and data lock-in (as in SaaS), but in this case, the onus is completely on the customer to create compatible export routines and more importantly for the customers' developers to understand and consider these differences that are pointed out.
- IaaS lock-in: varies depending on the specific infrastructure services consumed. Virtual machines (VMs) that can be moved to the cloud from (heterogeneous) data centers, and between vendors' IaaS clouds, are an asset for organizations. However, doing so requires cloud IaaS providers to support a standardized VM file format. Currently, there is a little in offer in terms of standardized file format for virtual machine images and VM management. While virtualization can remove concerns about physical hardware, distinct differences exist between common hypervisors such as ZEN, VMware, and others. For example, data lock-in is the obvious concern with IaaS storage services. IaaS storage provider offerings vary from simplistic key-/value-based data stores to policy enhanced file-based stores. Moreover, feature sets can vary significantly, hence so do storage semantics. However, application level dependence on specific policy features (e.g., access controls) may limit customer's choice of IaaS provider.

However, since the focus of this chapter is on mitigating potential risks of vendor lock-in at SaaS layer of the cloud computing stack, therefore, the following sub-section(s) presented below: (1) narrows the discussion parameters for SaaS application migration scenarios, and (2) serves the purpose of highlighting some but certainly not all the cases where interoperability, portability, and security are important issues when migrating in the cloud computing environment.

3.1. SaaS lock-in challenges

Despite the numerous advantages of cloud computing to organizations, many challenges such as data lock-in, application lock-in, and contract lock-in remain inadequately addressed. In this section, we aim to address these issues of concern as it pertains to SaaS usage and their implications to enterprise cloud adopters. We tackle the vendor lock-in challenges that act as barriers to either adopting cloud-based SaaS services in enterprises, or migrating/switching between SaaS vendors. Thus, our line of reasoning here provides a concise yet relevant

discussion and in-depth analysis of these issues with some fundamental guidelines that should be observed by organizations, entering a cloud computing service SaaS contract. While it is important to understand that the extent and the nature of vendor lock-in vary as per the cloud type, be aware, however, that our focus within this chapter is aimed at SaaS lock-in, specifically. Both PaaS lock-in and IaaS lock-in is outside the scope of this thesis.

As cloud computing adoption rate soars across enterprises (small or large), the risks of vendor lock-in is prevalent. Limited studies exist, except for [40–44], to analyze and highlight the complexity of vendor lock-in problem in the cloud environment. Therefore, when selecting SaaS offerings from cloud vendors, organizations need to consider and balance service criticality against the significance of avoiding potential risks of vendor lock-in. Though it is claimed that vendor lock-in is not exclusively a computing problem, since it also occurs in the classic IT setting—in this case, the customer has more control over the data and services. However, Conway and Curry in [45, 46] argues that due to the immaturity of current cloud computing environment, data, applications, and services are primarily vulnerable to the risk of lock-in. In general, with cloud computing architectures, the risk of vendor lock-in rises with the number of hardware and software components the vendor provides. Thus, the highest lock-in risks occur with SaaS services because the vendor controls all key components of the customer's information system. SaaS lock-in affects both data and applications. Besides, cloud SaaS offerings are often based on proprietary non-standard data formats and application logic, which can make migrating data and services to another cloud SaaS vendor difficult. This potential dependency for service provision on a cloud SaaS vendor may lead to specific data and application lock-in challenges as described below.

- Data lock-in challenge: in using cloud SaaS offerings, enterprise data are typically stored in a custom database schema designed by the SaaS vendor. SaaS cloud vendors generally do not provide conceptual or logical data models for their service. Most SaaS vendors offer API calls to read and export data records. However, if the provider does not offer readymade data "export" functionality, the enterprise will need to develop a program to extract their data and write it to a file ready for import to another vendor. It should be noted that database schemas, data formats, and application programming interfaces (APIs) are valuable in providing the function of interoperability of communication and processing within the SaaS cloud [41, 44]. However, the closed proprietarily coding of these key components across SaaS vendor offerings results in the need for resource (i.e., human effort, time, and cost) to be focused into developing a solution to break free from having the enterprise data locked into SaaS offerings (e.g., data models, platforms, and programming languages). While custom code may be needed for data transformation, it is also wise to check that standard data formats used by the enterprise can be supported by other cloud SaaS vendors or there is a transformation mechanism available. This further drives the requirement for consumers using the SaaS services to understand the business and associated data that needs to be managed to support the business process being automated or replaced, before making important migration decisions.
- Application lock-in challenge: replacing an on-premise ICT system with its cloud SaaS counterpart benefits from the advantages of converting capital expenditure to operational

cost [47, 48]. However, cloud SaaS applications are developed to run on a particular operating system. SaaS vendors typically develop these custom applications tailored to the needs of their target market. Porting them to operate on another cloud SaaS provider's environment is a significant effort, because the application processing logic is supplied by the vendor and data may be proprietary [43]. Likewise, a company can spend a considerable amount of time and effort moving its SaaS applications (and data stored in one system) to a cloud SaaS environment due to application lock-in risks. For instance, enterprise SaaS customers with a large user-base can incur very high switching costs when migrating to another SaaS vendor as the end-user experience is impacted (e.g., re-training staffs). However, it may be easy in the case of SaaS to terminate a service from one cloud vendor and start service with another. If the terminated vendor is contractually required to provide data, migrating may be of questionable use without significant cooperation and resources provided by the vendor. For example, if the data is maintained in a proprietary database architecture (e.g., NoSQL data models), a conversion effort will be required, and, unless the appropriate cooperation is obtained, the project may prove costlier and take longer than forecast. Furthermore, where the customer has developed programs to interact with the vendor's API directly (e.g., for integration with other applications), this will also need to be re-written to consider the new vendor's APIs. Accordingly, as pointed out by Polikaitis [49], standardizing on cloud SaaS environment is a serious decision with long-term financial implications for an enterprise.

The vendor lock-in challenges discussed in this section are high-category risks that organizations must tackle when considering cloud SaaS solutions. They present two potential drawbacks for cloud service consumers; first, the provider has the customer organization at a disadvantage, as it can push disagreeable terms on the customer because it has no viable exit strategy. Secondly, if the provider goes out business in the worst case, the customer may have trouble sourcing an alternative. This can take considerable time, cost, and effort to find a SaaS replacement and move the entire organization's data. However, regarding these challenges, an exit strategy will either mitigate or exacerbate the impact of such risks. There is a need for these organizations to understand what the exit strategy looks like, even if it is unlikely that they will exit a service soon—besides, no company would want to buy into a service where they feel they had no alternative provider [49]. An exit strategy in this context refers to a way of moving to another SaaS vendor if the enterprise wishes to do so. Hence, a missing exit strategy is said to exacerbate data and application lock-in risks in SaaS offerings.

3.2. SaaS lock-in dimensions and approaches for adoption

In any relationship between a cloud SaaS service vendor and cloud SaaS consumer, vulnerabilities exist that can result in vendor lock-in situations [50]. For example, a lack of standard technologies and unification of interfaces within the cloud stack creates barriers for migration. In today's cloud computing marketplace, data, applications, and services are vulnerable to the risk of lock-in. It is the cloud service customer's data that is the primary asset at risk from lock-in situations here. Hence, if a cloud SaaS customer's data cannot be migrated, accessed, or retrieved due to related challenges with portability and interoperability issues at the individual levels of

the cloud computing stack, business continuity is at risk. These issues, consequently, translate into two core dimensions of SaaS lock-in as precisely described below.

- Horizontal lock-in: cloud service consumers face horizontal lock-in situations when vendors restrict them to freely replace a SaaS solution with a similar or competitive product offering. This situation can arise when a customer wishes to move to another SaaS solution but is hindered by obstacles or migration limitations put in place by their vendor. This consequently affects data portability, re-creation of cloud-based services to on-premise (i.e., roll-back), integration and interoperability, etc. Some of the likelihood of issues with SaaS cloud vendors or technology products which give rise to horizontal lock-in situations are: discontinuing software products without clear roadmaps for replacement, developing economically unsupportable solutions, releasing products without appropriate quality checks, vendor application highly customized to suit enterprise, etc.
- Vertical lock-in: in this situation, cloud SaaS customers are restricted to the use of specific software and hardware within the overall cloud service stack because of a chosen SaaS solution. This implies also that the use of an operating system, database hardware vendor, and even any required implementation (or integration) partner during migration may be dictated by vendor. At the SaaS layer, vertical lock-in can be difficult to avoid since the choice and location of hardware at the cloud provider's data center is out of the cloud service customer's control. Thus, the idea will be to ensure whether the data centers are locked or not into a particular operating system environment through their choice of virtualization. Common issues and challenges fraught with vertical SaaS lock-in includes but not limited to enterprise infrastructure built around vendor proprietary standards, SaaS applications built using vendor proprietary APIs, data in SaaS cloud products resides in proprietary database with no ability to export, and the vendor owns data rights necessary to operate SaaS solution, etc.

Therefore, while the business value of cloud computing is compelling, it is clear from raised above that many organizations still face the challenge of lock-in when adopting cloud SaaS service capabilities. With regards to cloud adoption approaches in enterprises, for simplicity, in this section, we categorize cloud computing SaaS services into two broad titles, namely: (1) horizontal SaaS offerings and (2) vertical (or sector-specific) SaaS offerings. Horizontal SaaS offerings are typically applicable to organizations across a range of business sectors, i.e., they are not specific to a business but can be found in almost any kind of organization. Some common horizontal SaaS applications are in the areas of email, customer relationship management (CRM), productivity, collaboration, analytics, etc. With the proven success and maturing of horizontal SaaS offerings, sector-specific SaaS offerings are emerging to include application in the areas of logistics and supply chain management (SCM), for example. Vertical SaaS offerings refer to specialized applications that will be used to support a focused business function or core processes that are found within that industry, e.g., patient record management for hospitals, hotel management software, etc.

Being that cloud SaaS solutions are strategically engineered to have control points, it makes difficult for customers to migrate away from their technology to competing solutions [51]. Thus, it is important that customers review the SaaS lock-in discussed above, to determine

cloud vendors and technologies that have the highest replacement or switching costs, and are most likely to create operational, financial, or legal issues. Organizations should also analyze SaaS offerings (i.e., vertical or horizontal) in terms of Total Cost of Ownership (TCO)/Return of Investment (ROI) against associated risks such as vendor lock-in, interoperability, portability, and security, including defining a clear strategy for both private and public implementations before adopting specific SaaS offerings.

4. Ontology of cloud lock-in perspectives

To identify open challenges and facilitate future advancements, it is essential to synthesize and classify the research on cloud computing vendor lock-in conducted to date. In this section, we discuss causes and problems of vendor lock-in, and present a taxonomy of cloud lock-in challenges covering the hardware, operating system, virtualization, and data center levels. A model encompassing the various elements (or triggers) of vendor lock-in risks in cloud computing is presented. The analysis of this multi-dimensional model shows that each element can create different effects of lock-in on specific business processes operating in a cloud environment. With the intent to create a cloud lock-in model, both for studying proprietary lock-in challenges in the context of service migration and for supporting decision-making for enterprise cloud adoption. Authors' aim in this section is to consider the various risks and challenges of vendor lock-in presented in **Figure 1**, and organize them in hierarchical categories of perspectives, thus creating a cloud computing vendor lock-in taxonomy. But before doing so, it is important to make clear that for any given information processing system (whether in cloud or non-cloud environments), there are a few user categories—or more accurately, several "*roles*"—that have an interest in the system. Each role is interested in the same system, but their relative views of the system are different, they see different issues, they have different requirements, and they use different vocabularies (or languages) when describing the system. In this direction, rather than attempting to deal with the full complexity of cloud lock-in problem, author mainly attempts to recognize these different interests by defining different

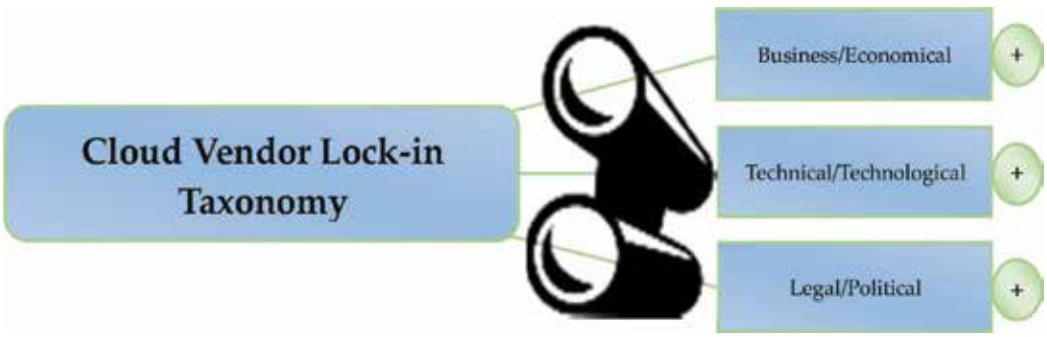

Figure 1. Perspectives for categorizing vendor lock-in risks in cloud computing. Top level overview of the viewpoints of cloud lock-in taxonomy, highlighting the three main perspectives to view the broad problem of vendor lock-in—related to business, technical and legal categories.

viewpoints of the lock-in problem in question. Each of these perspectives or viewpoints is chosen to reflect one set of inter-related consumer cloud lock-in concerns.

Across the three inter-related perspectives of vendor lock-in, organizations can use the proposed taxonomy to review their existing processes for cloud adoption and migration, data governance, and purchase policies to see if these support a strategy to achieve a high-level of flexibility and control to reduce the chance of being unavoidably locked into a single cloud provider offering. The aim of this taxonomy is to give both cloud service consumers (i.e., enterprises, end-users, developers, etc.) and cloud service providers guidance in the provision and selection of cloud services, indicating how to mitigate the risk of being tied to a cloud service provider—due to the difficulty and costs of switching to use equivalent cloud service from other providers. The taxonomy of cloud vendor lock-in perspectives partitions the challenges to be addressed into three viewpoints: business, technical, and legal. Each of the viewpoints can be used as problem analysis technique as well as solution space of the relevant issues of the lock-in problem domain. The main structure of the taxonomy along with its top levels of classification is depicted in **Figure 1**. The illustration is not meant to be exhaustive but to give a precise yet accurate view of the broad problem of cloud lock-in from different perspectives.

The three main perspectives of cloud vendor lock-in problem(s) are: business (or economics) perspective, technical (or technological) perspective, and legal (or political) perspective. Together they provide a complete picture of cloud computing vendor lock-in challenge. The concerns addressed in each of the perspectives are precisely presented. For instance, the business dimension is subdivided into standards, interoperability, portability, and security. The technical perspective includes constraints related to integration, compatibility, and APIs that are implementation-specific requirements or restrictions which may hinder interconnectability and/or trigger lock-in situation in the cloud. The complete organization of this scenario is presented in Section 5.3. While the first two categories correspond to enterprise architecture requirements (for enabling interoperability and portability) of products and IT services based on standard interfaces to interact seamlessly without the need for a large amount of integration efforts, the legal or political perspective is split into four sub-categories (i.e., SLA compliance, contract termination, cloud migration strategies, and metadata and data ownership) per the service life cycle and measures in which various aspects of cloud services offered and managed for a cloud service consumer can result in a lock-in situation. It is also noted that the lock-in risks in this dimension or perspective cover the complete information lifecycle (i.e., generation, use, transfer, transformation, storage, archiving, and destruction) inside the cloud providers' perimeter and in its immediate boundaries (or interfaces) to the consumers. The expansion of this categorization is depicted in **Figure 4**.

5. Taxonomy of vendor lock-in risks in cloud computing

A clear perspective of the main risk factors that contribute to a lock-in situation in the cloud environment and how such risk(s) should be organized to ease decision-making is the main step for having a comprehensive analysis of the status of cloud computing vendor lock-in

challenges. To organize the complex and broad data related to cloud lock-in problems and to facilitate further studies in this area, based on our previous work [44], the main problems (i.e., risks or challenges) of cloud lock-in are identified and grouped into a model composing of 11 categories namely: standards, portability, interoperability, security, integration, compatibility, APIs, data, contracts, SLA compliance, and cloud migration (in/out) strategies. These elements are placed in a hierarchical order of significance to the broad lock-in problem, in general. Moreover, the elements are significant considerations to the use of cloud services, and are also indicators to how component may trigger and/or intensify the risk of lock-in involved. The hierarchical categorization approach employed in this work assists in demonstrating how each element of vendor lock-in relates to several other components in the architecture of cloud computing. At a high level, the model establishes a common language (i.e., ontology) for easy understanding and communication of the capabilities and requirements which should be standardized in a cloud environment to facilitate open collaboration and interoperability amongst cloud providers, thereby avoiding the risk of a single provider lock-in. At a low level, the model is further composed into taxonomy to support consumers cloud service selection and adoption strategy in terms of validating cloud provider's solutions to achieve architectural integrity of business solutions of an enterprise's cloud ecosystem.

5.1. Classification methodology

Prior to presenting the proposed taxonomy, it is worthy to mention that the identification of elements and components of the categorization used is based on the critical review of key literatures in [40–45], the preceding section and subsequent sections. This critical review followed the systematic approach proposed by [52]. Some of these studies include standards and proposal documents from academia and industry as well as independent quantitative and qualitative studies conducted by author. The systematic review also covered general computing, IT and information systems (IS) journals, conference proceedings, books, industrial white papers, and technical reports. The fundamental purpose was to identify broadly any possible factors and issues that might lead to or intensify potential risks of vendor lock-in. Through this extensive and critical literature review, author established and proposed a set of potential cloud lock-in risk factors using taxonomy. The taxonomy is explained using case-examples from existing services of major cloud providers with an emphasis on the distinction made between services in software application programs (SaaS), platform (PaaS), and infrastructure (IaaS), which are commonly used within traditional enterprise computing or as the fundamental basis for cloud service classification. Based on the review of existing literature studies and the results extrapolated from our systematic study, the following constraints and challenges have been identified with switching between cloud SaaS vendors: switching cost, data portability, API propagation and integration issues, interoperability and standards, security risks, contract and SLA management, and legal challenges (data location constraints, data ownership rights, cloud in/exit issues, legal jurisdiction and compliance, etc.). They have been further grouped into three main challenge (i.e., technical, business environment, and legal) areas of SaaS migration, and briefly analyzed below. The first four are technical constraints to the growth (i.e., in terms of migration to, and adoption) of cloud computing SaaS services; the next four are internal business environment obstacles to switching between cloud vendors

once the SaaS solution has been and/or replaced; and the last four challenges are policy and legal issues intrinsic to cloud SaaS migration process. These challenges represent shared concerns that need to be addressed prior to SaaS adoption, or switching between cloud SaaS services and vendors.

As an example, integration and data portability, for instance, are two core lock-in risk factors mentioned and discussed in several of the referenced studies. This is because as new cloud SaaS services are deployed within an existing enterprise environment, the need to integrate them with various on-premise systems and other cloud services becomes important. Thus, the integration task and the need to ensure data portability have increased the complexity of decision-making in respect of enterprise cloud SaaS migration [41, 42, 53–55]. Therefore, as organizations struggle with the complexities of integrating cloud services with other critical systems residing on-premise, the ability to share data (i.e., portability) across these hybrid environments remains critical, and continues as more enterprise workloads and projects are committed to cloud computing SaaS services. As would be seen in the subsequent section, different elements of lock-in encountered in each category is described below to aid readers' understandability of the overall complexity of cloud lock-in situation in more details. Each of these elements in the categorization (or classification) model below, results in subdivisions highlighting the main risk factors of vendor lock-in that have been identified.

5.2. The business perspective

It focuses on the needs of the consumers of a cloud product or service offering. It describes the business challenges of vendor lock-in in terms of answering what is required of a cloud provider to meet customers' expectations to avoid over dependency on a product and the vendor. From a business perspective, avoiding vendor lock-in is requested by reasons varying from optimal service selection regarding utilization, costs or profits, to technology (hardware or software) changes. The adoption of cloud computing is still hindered by the lack of proper technology (or technology maturity), knowledge (of use), transparency, and trust issues. One of the problems spanning across these reasons is the low level of portability and interoperability of cloud applications and data storage services. The vendor lock-in challenge with respect to both low-level resource management and application level services is related also to the lack of world-wide adoption of standards or interfaces to leverage the dynamic landscape of cloud related offers. Portability and interoperability standards provide customers the ability to switch cloud providers without a lock-in to a provider. Moreover, data and applications in the cloud reside on systems' consumers who do not own and likely have only limited control over—which can result in loss of data and application security issues. Lack of interoperable and portable standards for different security policy or control, key management or data protection between providers may open undiscovered security gaps when moving to a new provider or platform. Hence, it becomes important to consider several items, for portable and interoperable security standards, to protect sensitive data being moved to or in the cloud. In this direction, author acknowledges that not all information used within a cloud system may qualify as confidential or fall under regulations requiring protection. Therefore, the security categories proposed in this case are based on information security lifecycle for protecting data

in terms of confidentiality, availability, and integrity (which can be applied not only to cloud environments, but also to any solution which requires basic interoperable and portable security integration). The complete organization is depicted in **Figure 2**.

5.3. The technical perspective

The technical dimension is subdivided into integration, compatibility, and APIs. In this case, the classification proposed are based on technical constraints placed on consumer's ability to achieve seamless integration and compatibility with user, administrative, and programming interfaces for using and controlling a cloud service. Since the interfaces and APIs of cloud services are not standardized, different providers use different APIs for what are otherwise comparable cloud services. These APIs expose the semantics (i.e., description of cloud services by its provider) and technologies (i.e., middleware and applications used to support a cloud service) used by a provider by providing the service management functionality. This implicit lack of standards (as pointed earlier) adoption by cloud providers is in fact a breeding ground for various types of heterogeneity (e.g., hardware and platform), because each cloud provider uses different technologies, protocols, and formats. This heterogeneity is a crucial problem as it gives rise to vendor lock-in situations in cloud computing. Thus, the need for a well-defined standard interface plays an important role toward achieving compatibility and manageability inside and between clouds. Then, cloud service consumers can take advantage of seamlessly integrating different provider offerings, combining benefits of each cloud to build solutions

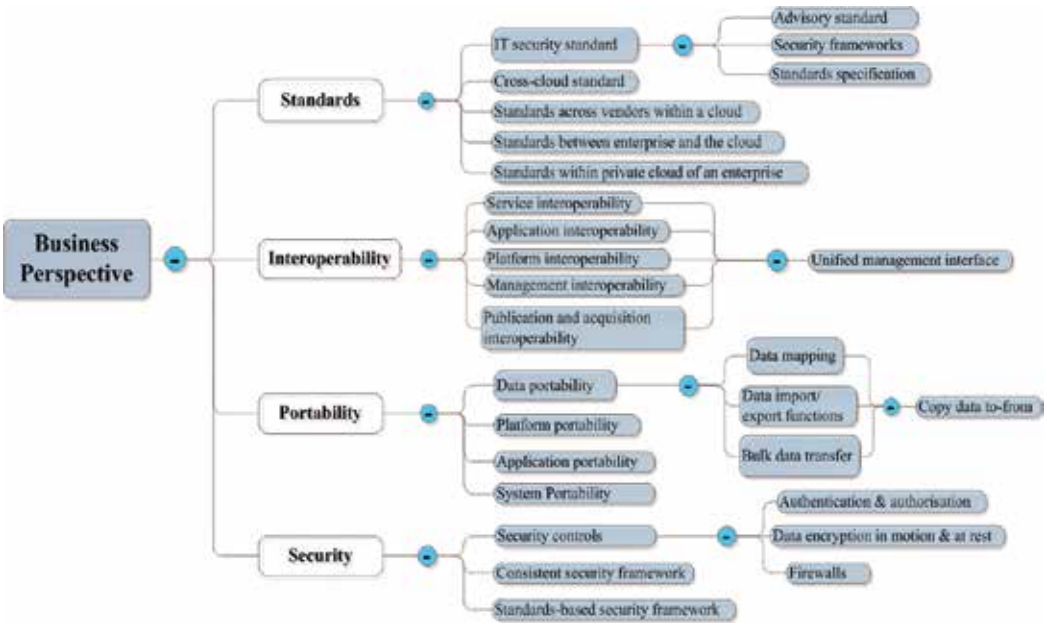

Figure 2. Vendor lock-in taxonomy-business perspective. NB: components from the business perspective of vendor lock-in are subdivided into four categories (i.e., standards, interoperability, portability and security). These elements are significant considerations to the use of cloud services, and are also indicators to how each component may trigger and/or intensify the risk of lock-in involved.

that are coherent to their respective business goals. The complete categorization of technical perspective of lock-in is presented in **Figure 3**.

5.4. The legal perspective

The need to avoid the risk(s) of cloud vendor lock-in from a legal perspective is to limit possible constraints on data, application, and services per the locations or national laws, as well as grant customers the free will to avoid dependence on only one external provider. The categorization in this dimension includes aspects related to contract and license issues, exit process or termination of use of a cloud service, judicial requirements and law (such as multiple data locations and privilege management). The legal perspective is split into four sub-categories (i.e., SLA compliance; contract termination and exit process; cloud migration strategies; and metadata and data ownership) per the service life cycle and measures in which various aspects of cloud services offered and managed for a cloud service consumer can result in a lock-in situation. It is also noted that the lock-in risks in this scenario cover the complete information lifecycle (i.e., generation, use, transfer, transformation, storage, archiving, and destruction) inside the cloud providers'

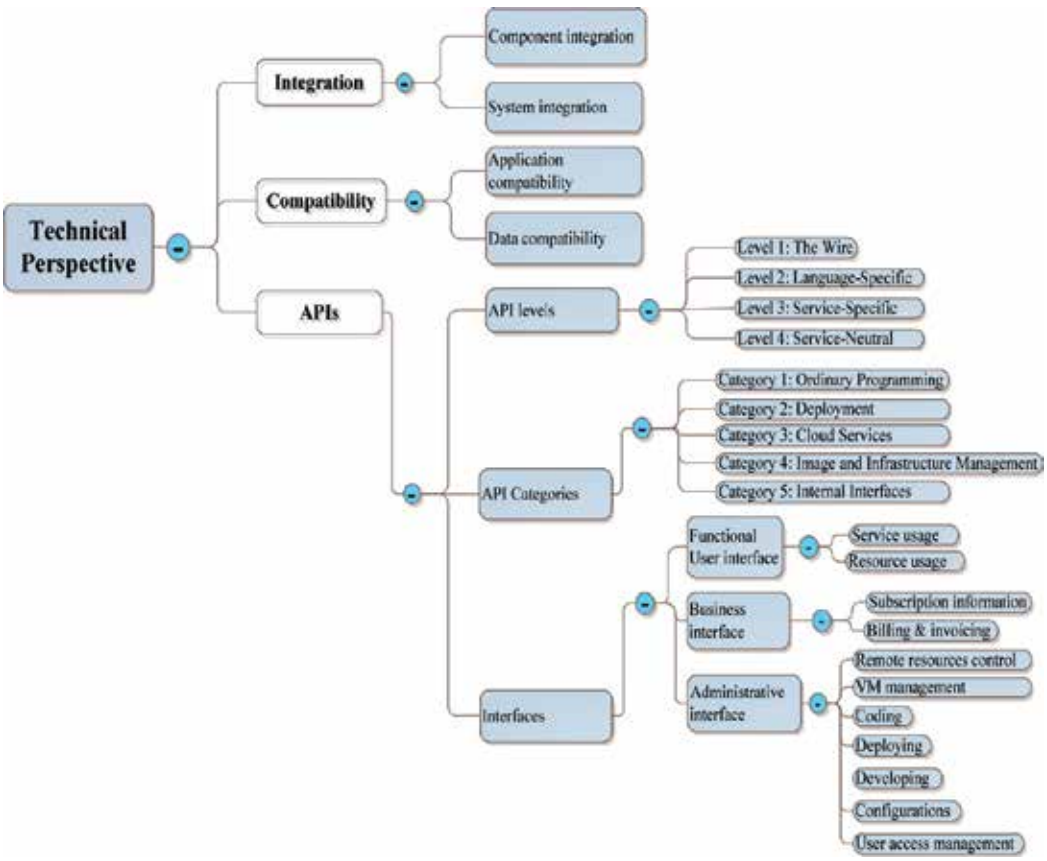

Figure 3. Vendor lock-in taxonomy—technical perspective.

perimeter and in its immediate boundaries (or interfaces) to the consumers. Audit and monitoring are also important aspects worth considering in the legal dimension, due to the requirements that a cloud provider should ensure to fulfill service agreements. For instance, the exit process or termination of the use of a cloud by a customer requires careful planning from an information security perspective. From a data security and storage perspective, it is important that once the customer has completed the termination process, none of the customer's data should remain with the provider. Thus, the exit process must allow customer to retrieve their data in a suitably secure form, backups must be retained for agreed periods before being eliminated and associated event logs and reporting data must be retained until the exit process is complete. Meanwhile, customers are advised to negotiate directly with their cloud service provider to ensure appropriate exit process provisions and assurances that are included and adequately documented in their cloud SLA and contracts. The expansion of this categorization is depicted in **Figure 4**.

5.5. Our contribution

In this section, we examined current proprietary lock-in risk factors in cloud computing services and proposed a model for classifying these to aid better enterprise cloud procurement

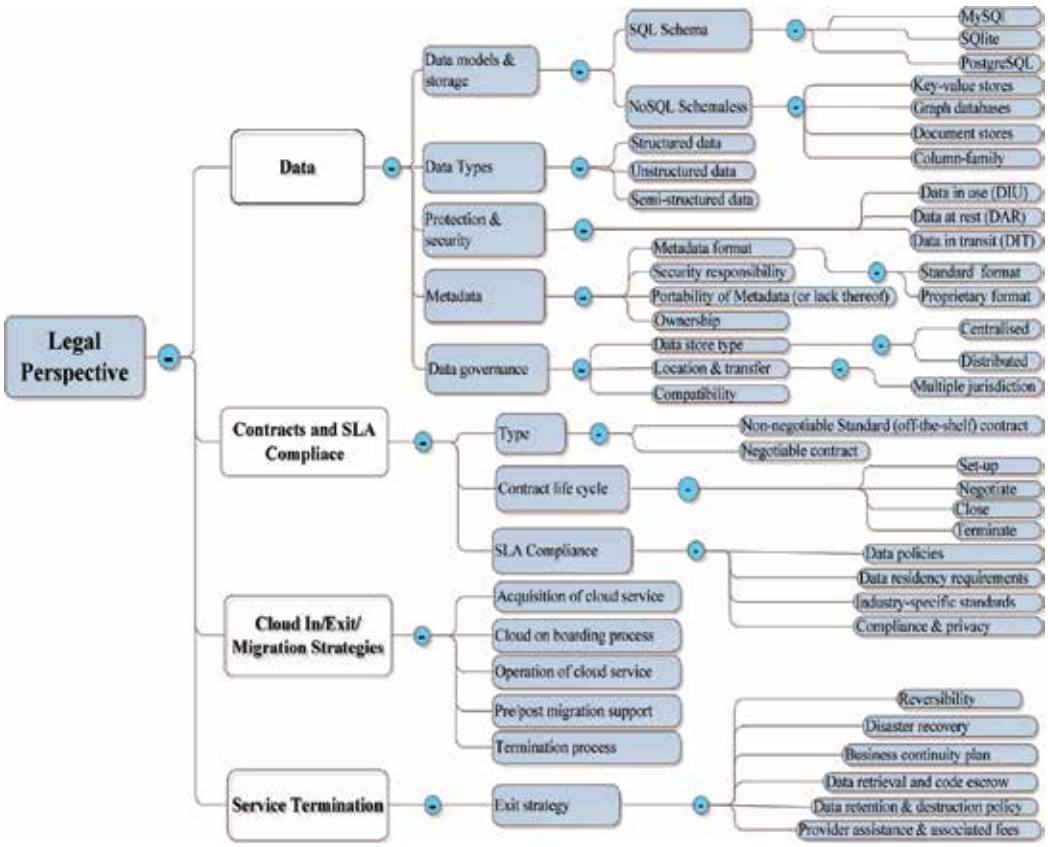

Figure 4. Vendor lock-in taxonomy—legal perspective.

and migration decision process. Aiming to organize this information into a useful tool for comparing, relating, and classifying already identified lock-in challenges and risks as well as future ones, we present a taxonomy proposal for cloud computing vendor lock-in. We focus on issues that are specific to cloud computing, without losing sight of important issues that also exist in other distributed IT systems. Here proposed taxonomy is capable of classifying both current and future cloud computing services for potential risks of single provider lock-in. The simple layered-tree structure used in the taxonomy development allows quick and easy analysis, by giving the user a set of core elements at each level. This clear structure makes analyzing and understanding the complexity of vendor lock-in challenges in cloud computing services more efficient than using table-based comparisons. While, table-based comparisons of cloud computing services exist [59], however, they are mainly for commercial use and the degree of detail varies greatly. Further, the taxonomy not only helps to categorize a cloud lock-in risk scenario, but it also helps potential customers and developers to point out what characteristics the service they seek or wish to develop should have (to avoid lock-in).

In summary, the contribution of this chapter is twofold. First, we identified the main risk factors of vendor lock-in that must be considered when transforming IT services to cloud-based solutions, or migrating to a cloud computing environment. These lock-in challenges were derived based on an extensive literature review and previous works on enterprise migration to public/private and/or hybrid cloud environment. We believe that the identified lock-in challenges are pertinent issues that require urgent solutions in the cloud environment, in order to pave the way for providing more standard and secured cloud services. Second, in order to classify and categorize the core elements of cloud computing lock-in, using a systematic classification scheme, taxonomy of cloud lock-in challenges is proposed. Moreover, this taxonomy extends our previous works presented in, respectively, providing an enhanced review of the cloud computing lock-in challenges previously presented, as well as a deeper analysis of the related work by discussing the main SaaS lock-in risk dimensions. Furthermore, we discuss the PaaS and IaaS lock-in aspects related to cloud computing, a fundamental yet still underserved field of research.

6. Conclusion

This chapter provides a comprehensive analysis of cloud computing vendor lock-in problem, and proposed taxonomy of cloud lock-in perspectives. The three main perspectives of cloud vendor lock-in problem(s) are: business (or economics) perspective, technical (or technological) perspective, and legal (or political) perspective. Together they provide a complete picture of cloud computing vendor lock-in challenge. The concerns addressed in each of the perspective have been precisely and concisely discussed in this chapter. The hierarchical categorization approach used in taxonomy assists in demonstrating how each element of the vendor lock-in problem relates to several other components in the architecture of a cloud computing system. At a high level, the model establishes a common language (i.e., ontology) for easy understanding and communication of the capabilities and requirements which should be standardized in a cloud environment to facilitate open collaboration and interoperability amongst cloud providers—thereby avoiding the risk of a single provider lock-in for cloud consumers. At a low

level, the model is further composed into taxonomy to support consumers cloud service selection and adoption strategy in terms of validating cloud provider's solutions to achieve architectural integrity of business solutions of an enterprises' cloud ecosystem. In contrast to existing works in the field, our study extends the current scope of cloud computing migration beyond one specific challenge area, instead it addresses the complex vendor lock-in problem from three main perspectives or categories—thereby contributing substantially to the growing body of knowledge on cloud computing.

In our future work, we present a high-level (combined) view of the proposed taxonomy. It will show the transformation between the different vendor lock-in perspectives. This high-level taxonomy of vendor lock-in risks identifies the key cloud computing interoperability, portability, API interface categories, as well as other relevant and intricate components of cloud systems that should be portable and interoperable. For example, standardization of the interfaces between these components is the first step to achieving interoperability and portability—as it prevents being locked into any cloud or provider. In the expanded taxonomy diagram, a layer represents a set of functional and non-functional requirements that provide similar capabilities or serve a similar purpose to support a vendor neutral (and technology-independent) sourcing strategy for cloud applications and services. We also survey various key works in the area and map them to our taxonomy to guide future design and development efforts.

Author details

Justice Opara-Martins

Address all correspondence to: joparamartins@bournemouth.ac.uk

Computing and Informatics Research Centre, Bournemouth University, Bournemouth, United Kingdom

References

[1] Sun X, Gao B, Zhang Y, An W, Cao H, Guo C, Sun W. Towards delivering analytical solutions in cloud: Business models and technical challenges. In: Proceedings of the IEEE 8th International Conference on e-Business Engineering (ICEBE 2011). Washington, USA: IEEE Computer Society; 2011. pp. 347-351

[2] Assunção MD, Calheiros RN, Bianchi S, Netto MA, Buyya R. Big data computing and clouds: Trends and future directions. Journal of Parallel and Distributed Computing. 2015;**79**:3-15

[3] Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computing Systems. 2009;**25**(6):599-616

[4] Opara-Martins J, Sahandi R, Tian F. Critical analysis of vendor lock-in and its impact on cloud computing migration: A business perspective. Journal of Cloud Computing. 2016;**5**(1):4

[5] Di Martino B, Cretella G, Esposito A, Sperandeo RG. Semantic representation of cloud services: A case study for Microsoft windows azure, In: Intelligent Networking and Collaborative Systems (INCoS), 2014 International Conference on. 2014. pp. 647-652. DOI: 10.1109/INCoS.2014.76

[6] Satzger B, Hummer W, Inzinger W. Winds of change: From vendor lock-in to the meta cloud. IEEE Internet Computing. 2013;**1**:69-73

[7] Binz T, Breiter G, Leyman F, Spatzier T. Portable cloud services using tosca. IEEE Internet Computing. 2012;**3**:80-85

[8] Petcu D, Macariu G, Panic S, Craciun C. Portable cloud applicationsfrom theory to practice. Future Generation Computer Systems. 2013;**29**(6):1417-1430. DOI: 10.1016/j.future.2012.01.009

[9] Armbrust M, Fox A, Griffith R, Joseph AD, Katz RH, Konwinski A, Lee G, Patterson DA, Rabkin A, Stoica I, Zaharia M. Above the Clouds: A Berkeley View of Cloud Computing, Technical Report UCB/EECS-2009-28. Electrical Engineering and Computer Sciences: University of California at Berkeley, Berkeley, USA; February 2009

[10] Katz DS, Jha S, Parashar M, Rana O, Weissman JB. Survey and Analysis of Production Distributed Computing Infrastructures. CoRR abs/1208.2649

[11] Toosi AN, Calheiros RN, Buyya R. Interconnected cloud computing environments: Challenges, taxonomy, and survey. ACM Computing Surveys (CSUR). 2014;**47**(1):7

[12] Hilley D. Cloud Computing: A Taxonomy of Platform and Infrastructure-Level Offerings. Georgia Institute of Technology; 2009

[13] Di Martino B, Cretella G, Esposito A. Classification and positioning of cloud definitions and use case scenarios for portability and interoperability. In: Future Internet of Things and Cloud (FiCloud), 3rd International Conference on. 2015. pp. 538-544. DOI: 10.1109/FiCloud.2015.119

[14] Abadi DJ. Data management in the cloud: Limitations and opportunities. IEEE data Engineering Bulletin. 2009;**32**(1):3-12

[15] Höfer CN, Karagiannis G. Cloud computing services: Taxonomy and comparison. Journal of Internet Services and Applications. 2011;**2**(2):81-94

[16] Rimal BP, Choi E, Lumb I. A taxonomy, survey, and issues of cloud computing ecosystems. In: Cloud Computing. London: Springer; 2010. pp. 21-46

[17] Clemons EK, Chen Y. Making the decision to contract for cloud services: Managing the risk of an extreme form of IT outsourcing. In System Sciences (HICSS), 2011 44th Hawaii International Conference on. IEEE; 2011. pp. 1-10

[18] Cloud Computing Use Case Discussion Group. Cloud computing use case. White Paper version 1.0.5. 2009

[19] Crandell M. Defogging cloud computing: A taxonomy. 2008. Available from http://gigaom.com/2008/06/16/defogging-cloud-computing-a-taxonomy/

[20] Laird P. Different strokes for different folks: A taxonomy of cloud offerings. Enterprise cloud submit, INTEROP. 2009

[21] Ried S. Yet another cloud—How many clouds do we need? Retrieved from Forrester Research. 2009. http://www.forrester.com/

[22] Wang L, Ranjan R, Chen J, Benatallah B, editors. Cloud Computing: Methodology, Systems, and Applications. CRC Press; 2017

[23] Zafar F, Khan A, Malik SUR, Ahmed M, Anjum A, Khan MI, Javed N, Alam M, Jamil F. A survey of cloud computing data integrity schemes: Design challenges, taxonomy and future trends. Computers & Security. 2017;**65**:29-49

[24] Wan Z, Wang P. A survey and taxonomy of cloud migration. In: Service Sciences (ICSS), 2014 International Conference on. IEEE; 2014, May. pp. 175-180

[25] Islam T, Manivannan D, Zeadally S. A classification and characterization of security threats in cloud computing. International Journal of Next-Generation Computing. 2016;**7**(1)

[26] Di Martino B., Cretella G, Esposito A. Towards a unified owl ontology of cloud vendors' appliances and services at paas and saas level. In: Complex, Intelligent and Software Intensive Systems (CISIS), 2014 Eighth International Conference on. IEEE; 2014, July. pp. 570-575

[27] Fatema K, Emeakaroha VC, Healy PD, Morrison JP, Lynn T. A survey of cloud monitoring tools: Taxonomy, capabilities and objectives. Journal of Parallel and Distributed Computing. 2014;**74**(10):2918-2933

[28] Almorsy M, Grundy J, Müller I. An analysis of the cloud computing security problem. 2016. arXiv preprint arXiv:1609.01107

[29] Mahjoub M, Mdhaffar A, Halima RB, Jmaiel M. A comparative study of the current cloud computing technologies and offers. In: Proceedings of the 2011 First International Symposium on Network Cloud Computing and Applications. IEEE Computer Society; 2011. pp. 131-134

[30] Peng J, Zhang X, Lei Z, Zhang B, Zhang W, Li Q. Comparison of several cloud computing platforms. In: Proceedings of the 2009 Second International Symposium on Information Science and Engineering. IEEE Computer Society; 2009. pp. 23-27

[31] Wind S. Open source cloud computing management platforms: Introduction, comparison, and recommendations for implementation. In: 2011 IEEE Conference on Open Systems, ICOS. 2011. pp. 175-179

[32] Cordeiro T, Damalio D, Pereira N, Endo P, Palhares A, Goncalves G, Sadok D, Kelner J, Melander B, Souza V, Mangs JE. Open source cloud computing platforms. In: 2010 9th International Conference on Grid and Cooperative Computing, GCC. 2010. pp. 366-371

[33] de Oliveira D, Baiao FA, Mattoso M. Towards a taxonomy for cloud computing from an e-science perspective. In: Cloud Computing. London: Springer; 2010. pp. 47-62

[34] Liu F, Tong J, Mao J, Bohn R, Messina J, Badger L, et al. NIST cloud computing reference architecture. September 2011. http://www.nist.gov/manuscriptpublication-search.cfm?pub_id=909505

[35] Rimal BP, Eunmi C, Lumb I. A taxonomy and survey of cloud computing systems. In: Fifth International Joint Conference on INC, IMS and IDC, 2009, NCM'09. 2009. pp. 44-51

[36] Beloglazov A, Buyya R, Lee Y, Zomaya A. A Taxonomy and Survey of Energy Efficient Data Centers and Cloud Computing Systems. 2010

[37] OpenCrowd—Cloud computing vendors taxonomy. September 2012. http://cloudtaxonomy.opencrowd.com/

[38] Intel Corporation. Cloud computing taxonomy and ecosystem analysis. September 2012. http://www.intel.com/content/dam/doc/case-study/intel-it-cloudcomputing-taxonomy-ecosystem-analysis-study.pdf

[39] Forrester. The evolution of cloud computing markets. September 2012. www.forrester.com/go?docid=57232

[40] Opara-Martins J, Sahandi R, Tian F. Critical review of vendor lock-in and its impact on adoption of cloud computing. In: Information Society (i-Society), 2014 International Conference on; November. IEEE; 2014. pp. 92-97

[41] Opara-Martins J, Sahandi R, Tian F. Implications of integration and interoperability for enterprise cloud-based applications. In: International Conference on Cloud Computing. Springer International Publishing; 2015. pp. 213-223

[42] Opara-Martins J, Sahandi R, Tian F. A business analysis of cloud computing: Data security and contract lock-in issues. In: P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2015 10th International Conference on; November. IEEE; 2015. pp. 665-670

[43] Opara-Martins J, Sahandi R, Tian F. Critical analysis of vendor lock-in and its impact on cloud computing migration: A business perspective. Journal of Cloud Computing. 2016; **5**(1):1-18

[44] Opara-Martins J, Sahandi R, Tian F. A holistic decision framework to avoid vendor lock-in for cloud SaaS migration. Computer and Information Science. 2017;**10**(3):29

[45] Conway G, Curry E. Managing cloud computing—A life cycle approach. In: CLOSER. 2012. pp. 198-207

[46] Conway G, Curry E. The IVI cloud computing life cycle. Cloud Computing and Services Science. 2013:183-199

[47] Sahandi R, Alkhalil A, Opara-Martins J. SMEs' perception of cloud computing: Potential and security. In: Working Conference on Virtual Enterprises; October. Springer Berlin Heidelberg; 2012. pp. 186-195

[48] Sahandi R, Alkhalil A, Opara-Martins J. Cloud computing from SME's perspective: A survey-based investigation. Journal of Information Technology Management. 2013;**24**(1):1-12

[49] Polikaitis A. Vendor and Sourcing Management: Maintaining Control of Vendor Relationships by Avoiding Vendor Lock-in. IDC Opinion Report. 2015. http://core0.staticworld.net/assets/2016/04/19/idc-vsm-avoiding-vendor-lock-in.pdf [Accessed 12 November 2016]

[50] Burns M. Cloud-based ERP: The risk of vendor lock-in. In: Emerging Issues and Technologies for ERP Systems, Class of Enterprise Systems Integration, 2011-12. UK: School of Computing and Mathematics, University of Derby; 2012. pp. 77-80

[51] Sakr S, Liu A, Batista D, Alomari M. A survey of large scale data management approaches in cloud environments. IEEE Communications Surveys Tutorials. 2011;**13**(3):311-336

[52] Peng HT, Hsu WW, Chen CH, Lai F, Ho JM. September. FinancialCloud: Open cloud framework of derivative pricing. In: Social Computing (SocialCom), 2013 International Conference on. IEEE; 2013. pp. 782-789

[53] Dillion T, Wu C, Chang E. Cloud computing: Issues and challenges, Advanced Information Networking and Application (AINA). In: 24th IEEE International Conference; 2010. 2010. pp. 752-757

[54] Garg SK, Versteeg S, Buyya R. A framework for ranking of cloud computing services. Future Generation Computer Systems. 2013;**29**(4):1012-1023

[55] Cusumano M. Cloud computing and SaaS as new computing platforms. Communications of the ACM. 2010;**53**:27-29

[56] Ardagna D, Di Nitto E, Casale G, Petcu D, Mohagheghi P, Mosser S, Matthews P, Gericke A, Ballagny C, D'Andria F, Nechifor CS. Modaclouds: A model-driven approach for the design and execution of applications on multiple clouds. In: Proceedings of the 4th International Workshop on Modelling in Software Engineering. IEEE Press; 2012. pp. 50-56

Chapter 2

Integrating Cloud Computing with Next-Generation Telematics for Energy Sustainability in Vehicular Networks

Matthew Liotine

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.74460

Abstract

This research focuses on new approaches to enhance the economic viability of newer hybrid/electric vehicle technology utilizing a telematic and cloud computing framework. First, an economic foundation is proposed that rewards drivers for energy efficient driver behavior in units of energy based on a predefined standard. Next, a service model is presented that allows drivers to transfer information regarding their energy efficiency through a telematic and cloud computing network. Based on existing cloud computing technology and telematic standards, a network architecture is proposed to transfer this information to service integrators and content providers that can use this information to create vehicle energy resource management capabilities for vehicle users and fleet owners. Such an architecture would enable drivers or fleet owners to redeem energy units for monetary or promotional incentives, thereby realizing more economic value for the vehicle investment.

Keywords: cloud computing, transportation telematics, pervasive computing, connected vehicle networks, mobile networks, vehicle ad-hoc networks, information and communications technology, sustainable mobility

1. Introduction

Recent decades have seen rapid growth in transportation to the point where its sustainability, both economically and environmentally, has been challenged. Vehicle ownership rates and miles driven have nearly tripled since 1950 [1], causing dramatic increases in congestion, pollution, and energy consumption. Furthermore, cars and light trucks alone now consume about 60% of

US transportation energy [2]. The federal government's growing concern over these issues has been realized through a pledge to reduce greenhouse gas emissions by 17% below 2005 levels by 2020 through more mandated stringent fuel economy standards for autos [3]. However, such standards are insufficient to make much of a difference to consumers in the near term since the expense of newer automotive technology can significantly offset fuel economy savings.

In fact, research suggests that the expense of newer hybrid/electric vehicle technology could actually reduce the benefit of a 50% fuel economy improvement by as much as 90% over the expected life of a vehicle [4]. As freeway traffic volume is relatively insensitive to fuel prices, the benefits of fuel economy are marginal at best even as fuel prices continue to climb in the short run. A 10% increase in fuel price would reduce fuel consumption by only less than 1% in the near term [5]. While the long-run benefits of both advanced technology and fuel price behavior may be more promising, such effects may not be realized until the year 2025 [6]. In order for fuel efficient technology to be successful in the near term, it is vital that additional value be created by augmenting it with capabilities such that drivers can realize full economic potential. Research suggests that additional fuel economy savings can be realized at little or no cost through alteration of driving behavior, such as adjusting speeds, aggressiveness, and distances, among others [7]. Further exacerbating this issue is the uncertainty surrounding the impacts on other modes of consumer transportation, namely mass transit. An increase in fuel price is said to have only a modest impact on transit ridership. It is estimated that even a 20% increase in fuel price results in only about a 2% increase in system ridership [8].

In energy, advances in automotive hybrid technology have made available near real-time vehicle fuel consumption information. This information can be leveraged to implement policies aimed at inducing energy-efficient driver behavior by monitoring fuel efficiency across a fleet or community of vehicles and dynamically providing incentives to drivers exhibiting more fuel-efficient behavior in order to preserve limited energy resources. New technology is being considered to help motivate improved driving behavior such as social networking and start-stop technology. Car-to-car communications through connected vehicle technology (CVT) has received widespread attention since it can provide a potential platform for realizing many advanced user services [9] to the extent that the US Department of Transportation has committed significant funding toward research in this area [6, 10]. While potential applications include using car-to-car negotiation to avert collisions [11], control pollution, and to share information about fuel consumption, fuel economy, routes, landmarks, etc., its potential as a platform for energy savings is yet to be fully explored. A CVT network [9] can serve as a possible platform for market-based transportation sustainability by serving as a connected dynamic decentralized system that monitors energy resource usage via the network and allowing drivers to manage their energy quotas with other vehicles in an energy information cloud managed through service providers. The added value of a platform with such capabilities is vital to energy transportation sustainability.

This chapter investigates the design of such a system by combining together economics, telematics, and cloud computing approaches for vehicle energy resource management. In particular, we leverage coordination mechanisms toward realizing an efficient energy resource allocation system incorporating parameters such as fuel efficiency and fuel price, among others. Each vehicle/user could receive energy resource quotas, receive credits for not exceeding

them, and could trade any unused energy resources (potentially with businesses as well as other vehicles/users) using a cap and trade (CAT) mechanism [9]. The network would monitor real-time resource usage levels, manage resource accounts, and facilitate economic transactions between vehicles/users/business and service providers that serve as broker centers. In this chapter, we analyze whether such a system is viable from an architecture perspective. We first investigate the economic foundation of a system for on-the-fly vehicle energy management. We then overview the capabilities of telematics toward realizing an efficient energy allocation system incorporating parameters such as fuel efficiency and driving behavior. We then evaluate the implementation of such a system via a cloud-based connected vehicle network platform.

2. Economic foundation

The fuel utilization of a vehicle is defined as its fuel consumption per unit distance. It is determined by a vehicle's technical characteristics, such as weight, aerodynamic design, and type of fuel. A typical driving cycle is comprised of periods of vehicle acceleration and deceleration, braking, stopping, and cruising. Consequently, fuel efficiency is also affected by the consumption during these periods. Driver behavior during each of these periods can profoundly affect fuel efficiency. Overall, it has been shown that the vehicle cruise velocity and the average velocity are powerful predictors of fuel efficiency [12]. The foundation for establishing a usable energy resource framework is the notion of a monetary unit that can quantify the value of preferred driver behavior. While fuel-efficient driver behavior often results in monetary benefits, it is often difficult to value such benefits due to the market fluctuations in fuel prices. At the outset, we can value such benefits in units of fuel, assuming that the price of fuel remains constant. Fundamentally, fuel efficiency can be achieved when drivers drive in a fuel-efficient manner. The benefits to society for driving in a fuel-efficient manner are well known. To this end, drivers can be rewarded in units of fuel (or other incentives) for driving fuel efficiently, stimulating better behavior.

3. Telematics

Telematics is the combination of the transmission of information over a telecommunication network and processing of this information [13]. It is used in sensor applications to capture, store, and exchange data from sensor devices. In the area of automotive telematics, the vehicle in effect becomes a computing platform and exchanges information with other vehicles, drivers, or stationary systems using wireless communication. Such communication is achieved using various wireless communication technologies, such as cellular communication networks, satellite, and dedicated short-range communication, among others. Telematics is also often referred as Information and Communications Technology (ICT). Telematics has advanced to the point where focus is beyond just being able to connect vehicles to each other or to infrastructure [14]. Advanced transport telematics have developed to the point where efforts have focused on standardized platforms that support information services for demand management by providing information such as travel times, congestion levels, and accidents drawn from Internet websites and information services [15].

To communicate such information to entities external to the vehicle, in-vehicle devices need to use over-the-air services using multiple access methods in technologies such as Universal Mobile Telecommunications System (UMTS), wi-fi, third generation (3G) or fourth generation (4G) wireless. To address this issue, a next-generation telematics pattern (NGTP) is under development by a global consortium comprising major automobile industry players. Unlike its predecessor, the short-lived global system for telematics (GST) project which focused on a service-oriented architecture (SOA) approach to running multiple services on an in-vehicle or mobile computing system, NGTP offers a new telematics framework and a technology-neutral open telematics protocol to bring greater flexibility and scalability to delivering over-the-air services to in-vehicle devices and handsets [16]. NGTP 2.0 is being positioned as a pattern versus a protocol standard, where a pattern is a common paradigm used to address a common reoccurring problem [17], similar to the context used in software engineering. This approach can accelerate development and integration among various entities and enable an open, flexible, and technology-neutral approach using existing technologies to the extent possible. NGTP 2.0 promotes a standard message format used to unidirectionally transmit data from a telematic unit (TU) residing in a vehicle to a dispatch (DSPT) function that extracts the message from the bearer so that it can be handled transparently from backend systems.

The NGTP 2.0 message contains fields that house header, DSPT, and service data. The service data field is essentially the data payload which could include road and vehicle information. Approaches have even been proposed to incorporate road information obtained from a telematic system to manage energy within hybrid-electric vehicles [18]. More recently, standards have been developed to transfer standardized information about vehicle fleets [19] over a telematics interface. The Association of Equipment Management Professionals (AEMP) Telematics Data Standard is an extensible markup language (XML) Web service that provides fleet information as a resource, typically on the Internet at a known uniform resource location (URL). Clients access the information sending requests to the server at the given Internet location. The server responds with an XML equipment information document whose vocabulary is defined by the standard. This architecture thus enables a path to transfer vehicle data from a telematic service provider to a vehicle owner. The data fields include header information detailing attributes about the vehicle (e.g., make, model, serial number), location (e.g., latitude and longitude), operating hours, fuel consumption, and distance traveled. Such data provide the foundation for utilizing the economic parameters described earlier. To leverage this information for Web services, XML has been reconciled with the predominant use of abstract syntax notation one (ASN.1) for efficient data transfer over cellular networks, which can integrate XML-encoding rules (XER) [20]. NGTP 2.0 can support both XML and ASN.1 encodings.

4. Cloud computing

A cloud computing platform could provide a feasible backend to the telematics frontend. Cloud computing is an Internet-based model which has been developed based on parallel, grid, and distributed computing concepts. Recently, cloud computing has become a commercialized service offering by many information technology (IT) companies so that client companies can

offload the operation of their IT software, platforms, and infrastructure to reduce capital and operating costs. It enables organizations to access and configure these resources as a shared pool. Cloud computing infrastructure is based largely on a distributed paradigm, using distributed processing, memory, and data storage. It relies on advanced technology to data and resources. These cloud computing service providers can federate many network-based resources such as servers and storage systems into virtual resource pools on a large scale. In this environment, applications can run in remote distributed systems versus on a single computer or server. Cloud computing also supports the pervasive computing paradigm, in which a user can rely on many interoperating computers working behind the scenes. Using virtualization, hardware and software can be integrated in a unified context and dynamically configured as resources [21]. Examples of cloud computing service offerings include Amazon EC2, which provides a virtual environment for users to run applications [22]; Google app engine which, which allows users to run applications using the python language and provides Web-based interface to manage application execution [23]; Microsoft Azure, which provides a platform to deploy and manage Web-based applications and Microsoft-managed datacenters [24]; and Aneka which provides application programming interfaces (APIs) and tools for rapid development and management of cloud applications [25].

Cloud computer offerings have been classified into software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS). SaaS consumers use the provider's applications running on a cloud infrastructure, accessing them using various client devices through a thin client interface such as a Web browser. The underlying cloud infrastructure includes network, servers, operating systems, and storage that are invisible to the consumer. SaaS providers are likely to be PaaS consumers, paying for memory, storage environments, and tools when they create SaaS applications. PaaS providers are consumers of IaaS providers, who offer cloud resources as virtualized resources pools, again masking the underlying computing and communications infrastructure that comprise the cloud. These roles have developed to the point where they have been standardized [26]. There have been numerous successful applications of cloud computing including college campus resource management [27], bioinformatics [28], medical information integration [29], land resource information management [30], and so on. The distributed nature of cloud computing offers the ubiquity required to support processing information obtained from large numbers of geographically dispersed vehicles.

5. Service model

The service model for the above can be conceptualized from the user perspective in **Figure 1**. In this figure, agents residing in the vehicle will collect and transmit energy-related data, as well as other vehicle information, over a presiding wireless bearer air interface. Many different wireless approaches can be used to access the cloud platform from a TU residing in a vehicle. Currently, 3G wireless data services such as evolution data optimized (EVDO), wideband code division multiple access (W-CDMA), and high-speed packet access (HSPA) have typical upload data rates in the range of 50–400 Kbps, while fourth-generation (4G) data

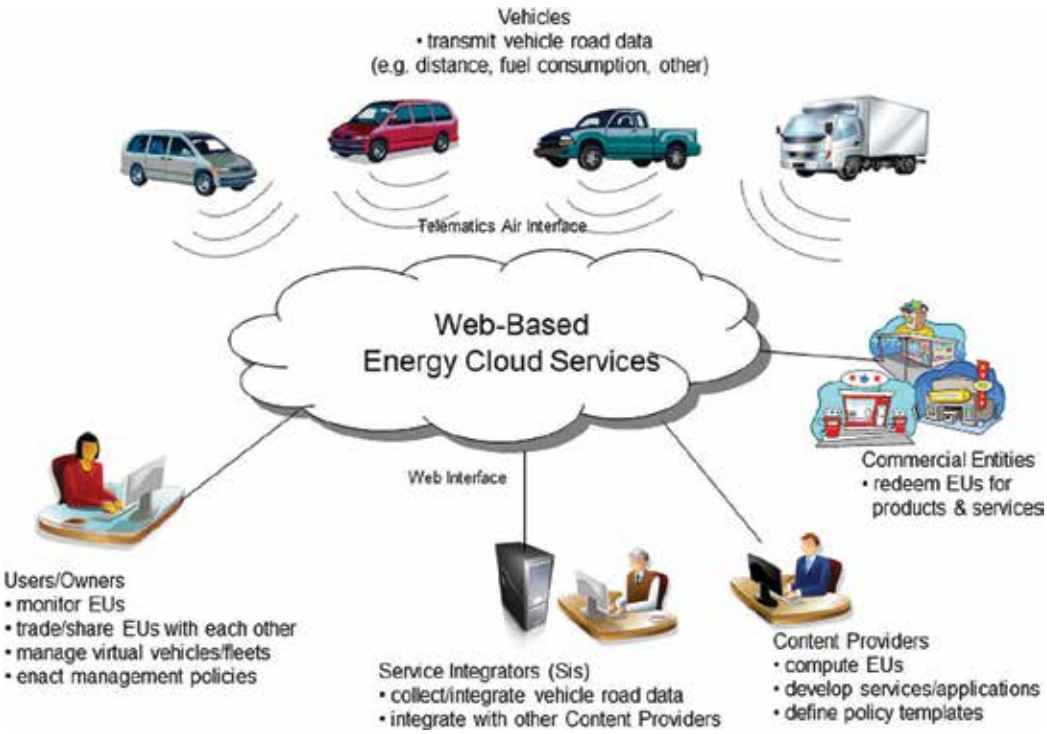

Figure 1. Service model.

services can achieve much higher data rates [31]. The least amount of data that is uploaded from a TU at a given time, the more expedient and economical the service will be. For energy consumption management, it is necessary to employ a data management cycle driven by the types of information that is being reported to the cloud [32]. For vehicles, a popular cycle is the driving cycle, usually associated with a single trip [12]. This cycle has also been adapted by AEMP. As discussed earlier in the UPS application, uploading such information at the conclusion of a drive cycle (or trip) may be more convenient.

From the user perspective, vehicle information is managed by a service integrator (SI) function, as defined by NGTP 2.0. The SI will collect vehicle data and publish them to content providers (CPs) who will develop the services and applications that users or owners can access to manage vehicle or fleet energy consumption. These applications can be developed as per the user's needs. The SI function avoids having CPs to develop custom code to integrate the data with their internal applications. Under NGTP 2.0, SIs and CPs are defined as functions (or roles), so there is no reason why they could be provided by the same organizational entity. For example, today, there are telematic service providers which users can subscribe to in order to obtain services such as pay-for-use insurance, location-based information, and roadside assistance. In these instances, the telematics service provider acts as the SI and CP.

In our model, users can access their services either through a Web portal, desktop application, or mobile app. The user can be presented with a virtual vehicle (VV), which is an abstraction

http://dx.doi.org/10.5772/intechopen.74460

of their physical vehicle. Users can also group multiple virtual vehicles into a virtual fleet (VF). Users can define the access control and the frequency of data collection for each vehicle and/or vehicle fleet. The aforementioned vehicle energy parameters require only mild computational effort and can be associated with both user and vehicle profile management so that the user can manage driver behavior at the individual vehicle or fleet level. Much of this behavior will be driven by the accumulation of EUs and how well the user can capitalize on either trading them with other users or redeeming them for products and services. This creates the opportunity to integrate information delivery with e-mail, blogs, wikis, and social networking services. Trading can be done according to the aforementioned types of policies, whereby CPs can create the virtual policy templates within the applications to enact such policies.

6. System model

The service model would be supported by a system model that calls for coherent integration of a variety of technologies related to cloud computing and telematics. This integration is visualized in **Figure 2** by using a layered approach. The telematic layer embodies many of the capabilities discussed earlier. The TU would utilize a software agent in the form of a fixed or downloadable application atop an open/standard embedded vehicle platform such as open services gateway initiative vehicle expert group (OSGI/VEG), AutoStar, automotive multimedia interface collaboration (AMI-C), cooperative vehicle-infrastructure systems (CVIS), OSEK virtual document exchange (OSEK/VDX), or android. Such platforms offer a middleware layer that standardizes APIs such that service providers or vehicle manufacturers can deliver service solutions more easily [33]. Such systems will utilize real-time operating systems such as QNX or Linux. The vehicle data layer comprises two functions as defined in NGTP 2.0. The DSPT function communicates with the TU, encodes or decodes the dispatcher portions of the NGTP messages, selects wireless bearers depending on the situation at hand, and manages connectivity to the TU. The service handler (SH) function encodes or decodes the service data portions of the NGTP messages, applies additional user or vehicle information, and may conceal certain vehicle characteristics. The DSPT-SH interface focuses on the transportation of information versus analyzing message content. The vehicle virtualization layer acts as a manager and collector of vehicle information and virtualizes the physical vehicles, so that users need not worry about the physical vehicle locations. This layer becomes the enabler for vehicle information applications and management. This layer comprises the SI and CP functions, both defined earlier. Unlike the vehicle data layer, most of the functions of the vehicle virtualization layer are service oriented versus technology oriented. With such information, the physical vehicle can then be abstracted into the virtual vehicle whose characteristics can be offered for use by various applications. Each virtual vehicle will publish data and metadata that describe the technical characteristics of the vehicle (e.g., owner, vehicle identification number, rated fuel efficiency) as well as trip characteristics (e.g., fuel consumption, distance, and time). Information collected from vehicles can be published to different CPs, depending on the context of the application. Different Web services or applications can access the data. An application can subscribe to multiple vehicles or vehicle fleets.

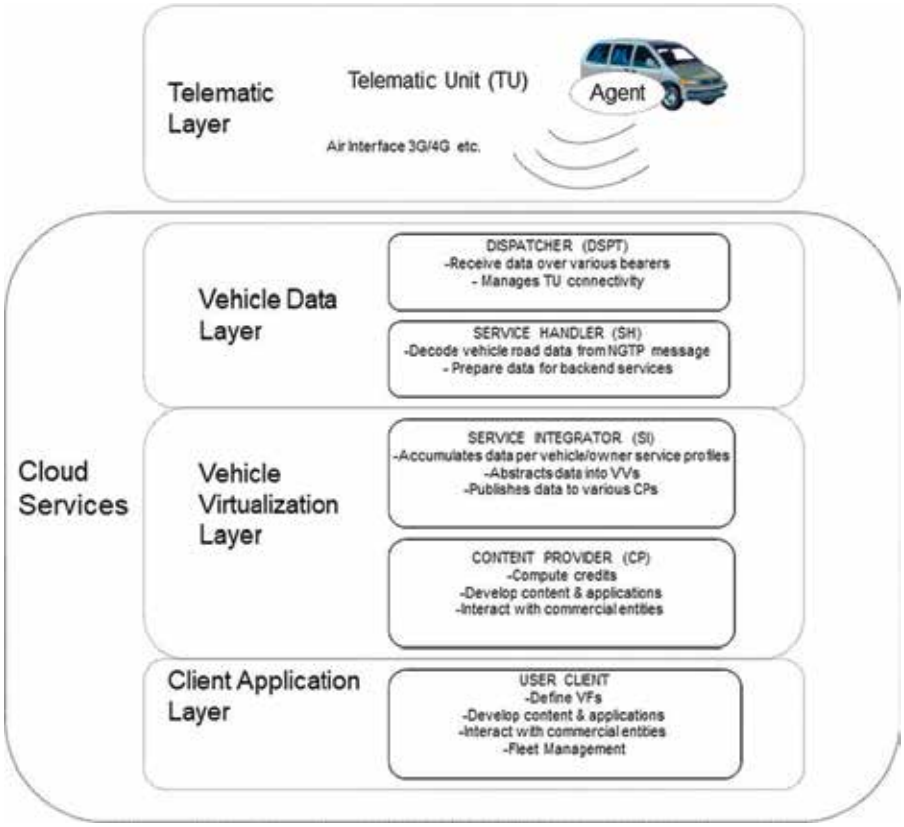

Figure 2. System model.

The essence of cloud computing is the ability to interact with many applications and services remotely from various locations over the Internet and other interfaces. A cloud model can provide a unified interface to access data from many vehicles, allowing it to be accessed and stored in a distributed manner, enabling service applications to be created around that data from multiple remote SIs and CPs. The PaaS layer allows users or companies to create and provide a rich set of applications based on need. For example, many applications can be developed for all of the vehicles owned by a particular user. Furthermore, the ability to integrate these applications with newer ones will likely be necessary. This avoids restricting one vehicle per user. This capability is similar to what has been done in cloud applications for sensor management [32].

For energy management, different applications can be developed and provisioned for different needs to respond to multiple audiences, especially those willing to promote sustainable mobility. Such applications fit well under the SaaS model, using the cloud as the application hosting platform. The end user can have one or more applications that use the vehicle data. These applications can be developed using a rich set of APIs provided by the SH. The end user requests the use of virtual vehicles or vehicle fleets and decides how to use that information either by developing their own applications or by utilizing predefined applications offered by a commercial entity, such as the vehicle manufacturer, fuel provider, car group, or similar.

Users can utilize policy templates to evaluate strategies to optimize fuel efficiency. These templates can be based on the economic parameters cited earlier to formulate decision strategies. In addition, geospatial information delivery methods based on geographic information system (GIS) technology can be leveraged to geographically view regions of fuel efficiency [34].

7. Network model

The service model that has been presented characterizes the end user's view of the energy services. Hidden from the user is the underlying network infrastructure that would be required to enable delivery of these services. **Figure 3** envisions a model architecture that could support delivery of these services. TUs will transmit the vehicle data at the conclusion of a driving cycle over the prevailing air interface of the cellular service provider's bearer network hosted by a base station (BS) unit. (While in this example we use cellular as a bearer network, other wireless access methods such as satellite could also be explored.) The BS is supported by the service provider's access service node (ASN), which serves as a gateway to their backend network. Vehicle data are carried as internet protocol (IP) payload in NGTP 2.0 ASN.1/XML format. The cellular provider's IP media subsystem (IMS) interconnects the backend

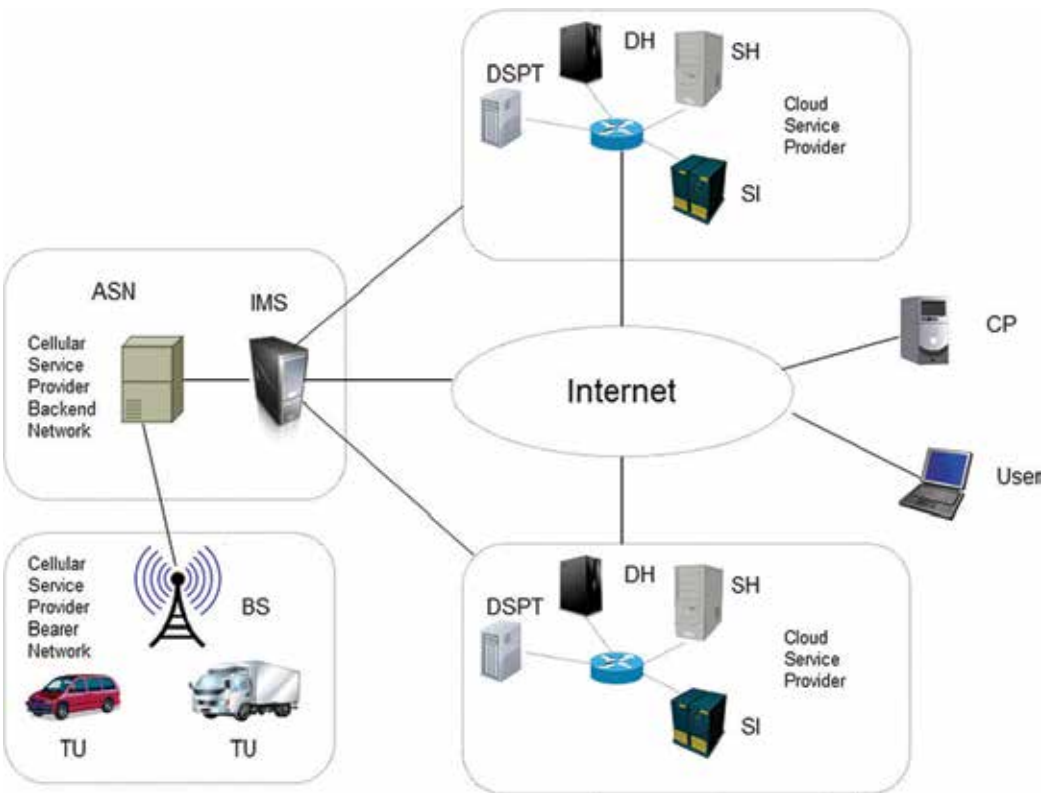

Figure 3. Network model.

network to external networks via IP. All data would be transferred in IP packet form. Several connectivity arrangements can be envisioned between the cellular service provider's backend network and the DSPT: direct connectivity to a DSPT node; connectivity to a cloud service provider's routing gateway; or connectivity to the cloud service provider's network via the Internet. The second option might be more desirable since there could be many distributed DSPT nodes on the cloud. As in the system model, DH, SH, and SI functions can be provided over a cloud service, thereby enabling a broader distribution of these functions. It should be noted that these functions need not reside on the same cloud nor be performed by the same service provider. Similarly, CPs and users are not confined to using the same cloud as well. As mentioned earlier, multiple federated clouds may have to be resorted to in situations where processing resources on a single cloud is insufficient.

Messages between the functions are unidirectional and technology independent. In a typical transaction, a TU will issue an NGTP 2.0 message to a DSPT in the conclusion of a driving cycle. The DSPT assigns a unique identifier (EventID) to identify and track the event. The DSPT will only process the NGTP message header and DSPT data fields. These fields include the nature of the service and network supporting the service. The message is then issued to the appropriate SH via hypertext transfer protocol secure (HTTPS). As stated earlier, message transmission in the IP datagram format enables layer 3 transport over a variety of layer 2 network options. The SH parses the service data payload field of the message. This data is essentially an AEMP equipment information (EI) document whose vocabulary is defined in [19]. Conceptually, the information about a vehicle is provided as a resource on the Internet, at a known uniform resource location (URL) of the DSPT. Furthermore, the SH can assign a URL to an individual vehicle. The SH processes the service data payload and enriches it for the SI. The SH provides this information in a common format by developing an API that enables SIs to parse the data and imports it into their databases for use by a CP that will develop and operate owner/end user applications.

8. Conclusions

As the next generation of more energy-efficient vehicles infiltrates the market, it is clear that their higher costs could offset energy savings over the long run. In order for fuel-efficient technology to be successful in the near term, it is vital that additional value be created by augmenting it with capabilities such that drivers can realize full economic potential. Such values can be enhanced through capabilities that motivate drivers to behave more fuel efficiently. To this end, we have proposed an architectural framework for real-time energy management use by vehicles. The framework is underpinned by a simple economic model which rewards drivers in units of energy when they drive more fuel efficiently than a given standard. By logging their activities using a combined telematic and cloud computing network, energy usage information could be transferred to service integrators and content providers that can use this information to build vehicle energy resource management capabilities. Such an infrastructure is quite feasible since many of the underlying capabilities needed for such a service are already available and have been standardized to many extents.

Author details

Matthew Liotine

Address all correspondence to: mliotine@uic.edu

University of Illinois at Chicago, Chicago, Illinois, USA

References

[1] U.S. Department of Energy. U.S. Department of Energy Data Book. [Online]; September, 2011. Available from: http://cta.ornl.gov/data/chapter8.shtml

[2] U.S. Department of Energy. Transportation Energy Data Book: Edition 30-2011. Oak Ridge National Laboratory; June, 2011

[3] The White House [Online]. 2011. Available from: http://www.whitehouse.gov/the-press-office/2011/07/29/president-obama-announces-historic-545-mpg-fuel-efficiency-standard

[4] Greene DL, DeCicco J. Engineering-Economic Analyses of Automotive Fuel Economy Potential in the United States. Oak Ridge National Laboratory, Center for Transportation Analysis; January, 2000

[5] The White House [Online]; January 29, 2010 [cited 2011 September]. Available from: http://www.whitehouse.gov/the-press-office/president-obama-sets-greenhouse-gas-emissions-reduction-target-federal-operations

[6] Energy.gov. Department of Energy Awards More Than $175 Million for Advanced Vehicle Research and Development. Energy.gov. August 10, 2011

[7] Moawad A, Singh G, et al. Impact of Real World Drive Cycles on PHEV Fuel Efficiency and Cost for Powertrain and Battery Characteristics. In: EVS24 International Battery, Hybrid and Fuel Cell Electric Vehicle Symposium; 2009; Stavanger, Norway. pp. 1-2

[8] U.S. Congressional Budget Office. Effects of Gasoline Prices on Driving Behavior and Vehicle Markets. In: Congress of the United States Congressional Budget Office; January, 2008. pp. 2-4

[9] Ouksel AM, Lundquist D. Cap and Trade Economic Policies for Resource Management in Vehicular Networks and Other Applications. In: 18th Intelligent Transportation Systems (ITS) World Congress; 2011; Orlando, Florida

[10] University of Michigan. UMTRI Awarded $14.9 Million USDOT Connected Vehicle Contract. University of Michigan Transportation Research Institute. August 19, 2011

[11] U.S. Department of Transportation. Applications for the Environment: Real-Time Information Synthesis (AERIS). U.S. Department of Transportation Research and Innovative Technology Administration; 2011

[12] An F, Ross M. Model of fuel economy with applications to driving cycles and traffic management. Transportation Research Record. 1993;**1416**:105-114

[13] Goel A. Fleet telematics real-time management and planning of commercial vehicle operations. Operations Research/Computer Science Interfaces Series. 2008:40

[14] Schmidt A, Paradiso J, Noble B. Automotive pervasive computing. Pervasive Computing. July-September 2011;**1536-1268**(11):12-13

[15] Negrenti E, Valenti G. Validation of Advanced Transport Telematics Applications in 5 European Capitals. In: 2001 IEEE Intelligent Transportation Systems Conference Proceedings; 2001; Oakland, CA

[16] eSafety Forum Working Group on SOA. Final Report of Service Oriented Architecture Working Group. Report and Recommendations. Brussels; 2010

[17] The NGTP Group. NGTP 2.0 Compendium 1.0; 2010

[18] Fijalkowski BT. Automotive mechatronics: Operational and practical issues. Intelligent Systems, Control and Automation: Science and Engineering. 2011;**47**

[19] AEMP. AEMP telematics data standard V 1.2. In: Association of Equipment Management Professionals. 2011

[20] Duibuisson O. ASN.1: A Powerful Notation for XML and Fast Web Services. International Telecommunications Union. April, 2004

[21] Ge J, Zhang B, Fang Y. Research on the resource monitoring model under cloud computing. In: WISM 2010. Berlin Heidelberg: Springer-Verlag; 2010. pp. 111-118

[22] Amazon.com. Amazon EC2 [Online]. Available from: http://aws.amazon.com/ec2/

[23] Google. Google App Engine [Online]; 2012

[24] Microsoft [Online]; 2010. Available from: http://www.windowsazure.com/en-us

[25] Manjrasoft [Online]; 2012. Available from: http://www.manjrasoft.com

[26] NIST. NIST Cloud Computing Reference Architecture. NIST Special Publication. National Institute of Standards; 2011. Report No.: 500-292

[27] Liu X, Song H, He D, Yang H. Research of Campus Resource Management Based on Cloud Computing. In: The 5th International Conference on Computer Science and Education; 2010; Hefei, China. pp. 1407-1409

[28] Bajo J, Zato C, de la Prieta F, de Luis A, Tapia D. Cloud Computing in Bioinformatics. Distributed Computing and Artifical Intelligence. AISC. 2010;**79**:147-155

[29] Huang Q, Ye L, Yu M, Wu F, Liang R. Medical Information Integration Based Cloud Computing. In: 2011 International Conference on Network Computing and Information Security; 2011: IEEE. pp. 79-83

[30] Fang L, Yao S, Liu T, Liu R. A Cloud Computing Application in Land Resources Information Management. In: 2010 Ninth International Conference on Grid and Cloud Computing; 2010: IEEE. pp. 338-393

[31] Sprint. Mobile 4G: The Revolution is Here Now, Version 1.0. Sprint; 2012

[32] Satoh S, Itakura M. Cloud-Based Infrastructure for Managing and Analyzing Environmental Resources. In: 2011 Annual SRII Global Conference; 2011: IEEE Computer Society

[33] Chang TW. Android/OSGi-based vehicular network management system. In: International Conference on Advanced Communication Technology (ICACT). 2010. pp. 1644-1649

[34] Di Martino S, Giorio C, Galiero R. A rich cloud application to improve sustainable mobility. In: W2GIS 2011. Berlin Heidelberg: Springer-Verlag; 2011. pp. 109-123

Chapter 3

Mobile Services Meet Distributed Cloud: Benefits, Applications, and Challenges

Tien-Dung Nguyen, Yunkon Kim, Xuan-Qui Pham,
Tri D.T. Nguyen and Eui-Nam Huh

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.75818

Abstract

As the explosive growth of smart devices and enormous new applications, the variety of corresponding cloud services has been growing quickly. The conventional centralized cloud was faced with an overhead on backhaul links and high latency. Accordingly, a decentralized cloud paradigm including edge computing, mobile edge computing, cloudlet, and so on, was introduced to distribute cloud services to the edge network which located in proximity to mobile devices few years ago. However, this paradigm was not paid attention at that time since cloud technology and mobile network communication were immature to motivate mobile services. Recently, with the overwhelming growth of mobile communication technology and cloud technology, distributed cloud is emerging as a paradigm well equipped with technologies to support a broad range of mobile services. The 5G mobile communication technology provides high-speed data and low latency. Cloud services can be automatically deployed in the edge networks quickly and easily. Distributed cloud can prove itself to bring many benefits for mobile service such as reducing network latency, as well as computational and network overhead at the central cloud. Besides, we present some applications to emphasize the necessity of distributed cloud for mobile service and discuss further technical challenges in distributed cloud.

Keywords: distributed cloud, mobile service, edge cloud, mobile edge cloud

1. Introduction

As cloud computing becomes an essential technology on Information and Communication Technology (ICT) and easy to use for the public, mobile industry has been gradually expanding.

Mobile devices are increasing rapidly with diverse kinds, such as smart glass, smart watch, smart bracelet, or even smart plaster. Although mobile devices are much more powerful, they are still hard to run computing intensive applications due to limitation of computing capacity and battery consumption. For example, machine learning service needs high-performance computing capability, and disaster analysis needs real-time capability. Mobile cloud computing (MCC) [1] has been proposed to offload heavy computing from mobile devices to centralized core cloud as well as to provide huge data storage capabilities to the cloud service customers (CSCs).

According to the rapid growth of mobile devices, the mobile services are also developed overwhelmingly. Some cloud services require low latency, and others require high traffic volume. These requirements apparently degrade performance of the current centralized cloud. If a large number of users simultaneously request cloud services at the same geographical location, the user demands would become a heavy burden on the backhaul links and make long latency for other cloud services [2]. Especially, in wide area network (WAN) environment, cloud service latency is quite difficult to meet quality of service (QoS) due to many uncontrollable causes [3, 4]. Moreover, one of the most important performance metric is measured by user experience that requires low latency to satisfy customers [5–7]. In the recent years, the latency-related issues have been studied for the purpose of bringing best performance for users [2, 8, 9]. Looking at the above limitations of centralized cloud, a next generation cloud is necessary to leverage mobile cloud market.

From the above necessity, the decentralized cloud has been widely researched and developed in the recent years. Many proposed concepts or techniques have been introduced to innovate a next generation cloud. We now alternately take into account these recent approaches to have an overview of the decentralized cloud.

First, several decentralized cloud concepts, such as fog/edge computing [10, 20], mobile edge computing (MEC) [11], cloudlet [12], micro data center [13], and cloud radio access network (C-RAN) [7], are described in **Table 1**.

The above concepts bring different views of decentralized cloud. However, each concept only represents an aspect of distributed cloud. While FC/EC focuses on Wi-Fi communication and supports mainly for Internet of thing (IoT), MEC, on the other hand, focuses on radio access network. Cloudlet does not mention about communication and concentrates on cloud computing resource. Micro data center focuses on data-related features and C-RAN tries to separate an edge network into two parts: cloud computing and radio access network. Each concept cannot draw a general view of distributed cloud. Therefore, in this chapter, we will present a fully distributed cloud that can support all above concepts.

Second, the fifth generation (5G)-based cellular network is a significant technology for distributed cloud, especially for multimedia cloud service and in vehicular ad hoc network (VANET). 5G technology is promising to provide high reliability and low latency. For example, the current communication link in VANET uses IEEE 802.11p (bandwidth limitation to 10 Mb/s per channel), which impacts augmented reality (AR) applications on video transmission (required data rate 40 Mb/s) [15]. The 5G cellular network with wide

Issued by	Concept	Definition
CISCO 2011	Fog/edge computing (FC/EC)	• ends the cloud to the edge of networks, in particular wireless networks for the Internet of Things (IoT) • pushes the computing applications, data, and services away from centralized nodes to the network edge, enabling analytics and knowledge generation to occur close to the data sources.
ETSI 2014	Mobile edge computing (MEC)	• pushes the cloud computing capabilities close to base station, • applied for Radio Access Networks in 4G, 5G. • is renamed to multi-access edge computing in 2016 [14]
Carnegie Mellon Univ. 2013	Cloudlet	• is known as limited-resource cloud node located close to users.
Microsoft 2015	Micro-data center (mDC)	• represents the middle tier of a three-tier hierarchy: mobile device cloudlet cloud • is an extension of hyper-scale cloud data centers (e.g. Microsoft Azure)
China Mobile Research Institute 2010	Cloud Radio Access Network (C-RAN)	• is combination of legacy radio access network and cloud computing to improve performance of base station

Table 1. Survey of decentralized cloud concepts.

spectrum (28 Ghz), multiple-in/out and ultra-dense network technologies can support 7.5 Gb/s data rate and achieve stable connection at 1.2 Gb/s in a mobile environment at the speed of 100 km/h [16].

Third, Software Defined Network (SDN) [17] represents an emerging network paradigm that has potential ability to join Wi-Fi/cellular networks and cloud. Originally, SDN was designed for and deployed in wired network environment with high-speed switch. SDN has data plane to transmit data and control plane to indicate where data should go. By integrating SDN, routers not only execute the basic network functions such as send/receive IP-based data packets but provide cloud services by itself or interact with other routers for computing resources as well. Therefore, the next generation network nodes will be considered to combine network functions and cloud functions.

Lastly, hierarchical network topology plays an important role to efficiently support service delivery and mobility. Proxy Mobile IPv6 (PMIPv6) [18] represents a host-based mobility management protocol that provides continuous services for mobile users. PMIPv6 includes: mobile Taccess gateway (MAG), local mobility anchor (LMA), and authentication-authorization-accounting (AAA). The MAG and LMA cooperate with each other to manage the movements of a mobile node. The AAA server is used for security and payment purpose.

According to the above related works, we will present a new type of distributed cloud based on the above decentralized cloud concepts and techniques. We believe that the distributed cloud has capability to provide high quality and real-time cloud services (augmented reality/virtual reality (AR/VR) or autonomous car, etc.) by reducing network latency and congestion.

In the following section, we will describe an overview of distributed cloud and its benefits. We then present some mobile applications that were emerged from distributed cloud. The open challenging research issues will be discussed in the later section.

2. Distributed cloud overview

2.1. Concept of distributed cloud

The purpose of distributed cloud is to bring a cloud service to a proximity of users geographically in order to reduce latency, network congestion, and guarantee QoS for mobile users. As mentioned above, a hierarchical network topology is considered to set up edge nodes or edge cloud (network node close to user) and to support mobility. The distributed cloud consists of core, regional, and edge cloud as shown in **Figure 1**. A core cloud is conventional cloud. A regional cloud and an edge cloud are improved from network node. An edge cloud is located in the proximity of users. A regional cloud locates between a core cloud and an edge cloud. Generally, a core cloud has the biggest scale, scale of a regional cloud is smaller than that of a core cloud, and the edge cloud has the lowest scale.

Three main layers of distributed cloud are described in detail as follows:

- Core cloud: It exists on the top layer of the distributed cloud. It has capabilities of whole works of conventional cloud including management and provision of cloud services and data. Examples of core cloud are Google Cloud, Amazon AWS, Microsoft Azure, and so on.
- Regional cloud: It exists on the middle layer of the distributed cloud. It has capabilities of proxy to support high mobility of mobile devices as well as self-deploy cloud service images and data caching. Examples of regional cloud are local mobility anchor (LMA) on Proxy Mobile IPv6 (PMIPv6), middle node on content delivery networks (CDN), etc.
- Edge cloud: It exists on the bottom layer of the distributed cloud and locates in the proximity of users. It has capabilities of being aware of mobile device, service, location, preferences, and so on, and then deploying and providing real-time cloud services. Examples of edge cloud are mobile edge computing (MEC), fog/edge computing (FC), cloudlet, and so on.

On the other hand, core, regional, and edge cloud can be managed on different administration domains because starting points of them are totally different. For example, a cloud service provider (CSP) (e.g., Google, AWS, Azure, and etc.) manages core cloud but a CSP does

http://dx.doi.org/10.5772/intechopen.75818

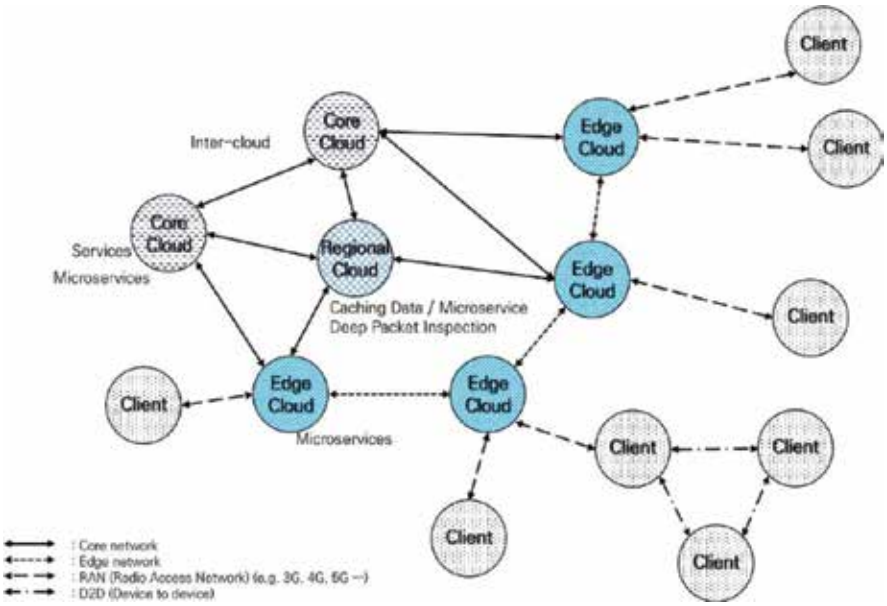

Figure 1. Concept of distributed cloud.

not have network infrastructure, so if regional and edge cloud are managed by a network provider (e.g., SK Telecom, Korea Telecom, LG Telecom, etc. in Korea), a CSP and a network provider may cooperate with each other. Besides, in case of a company has both core cloud

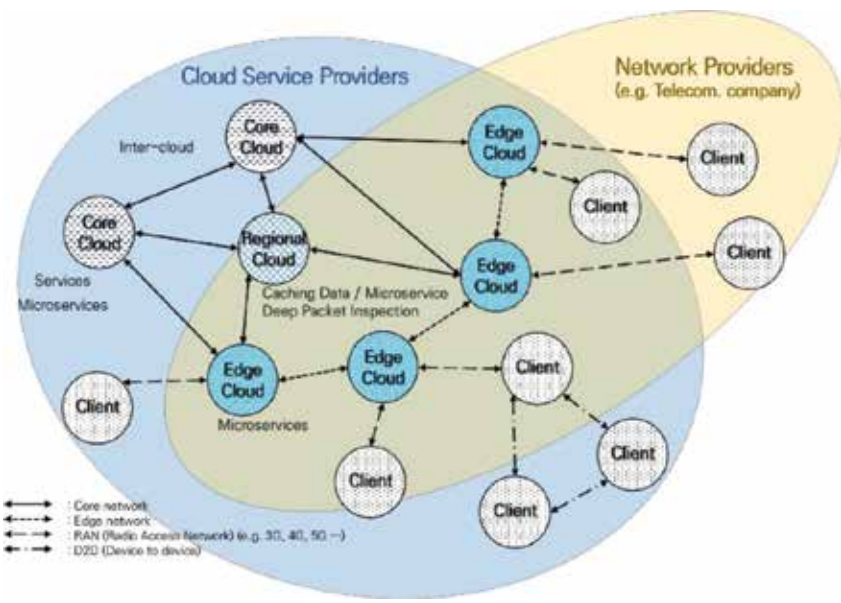

Figure 2. Administrator domains on distributed cloud.

and network infrastructure, the company can also cooperate with other CSPs for leasing their network infrastructure. Because the edge and regional cloud are deployed from the network node, they can provide two functions: cloud and network, managed by network providers and cloud service providers. Therefore, at least two administration domains exist on distributed cloud environment as in **Figure 2**.

2.2. Distributed cloud and conventional cloud

The concept of distributed cloud was started from an idea to combine computing resource on network node. In conventional cloud shown in **Figure 3**, between an end device and core cloud, there are many network nodes. Since network nodes cannot process cloud service and must forward to core cloud, it clearly causes congestion across the network, increases latency, and degrades cloud service performance. Therefore, conventional cloud environment is gradually changed to the distributed cloud environment as shown in **Figure 4**. The network nodes are enhanced with cloud functions, so-called edge cloud and regional cloud. Edge cloud is located on access/edge network where it is close to users. Regional cloud is located on distribution network. Core cloud is located on core network. All kinds of clouds can provide cloud services. As compared with the conventional cloud environment, the distributed cloud environment is able to provide a cloud service near cloud service customers. Therefore, network latency of distributed cloud is clearly lower than conventional cloud. As a result, the distributed cloud is proper to provide responsive cloud services.

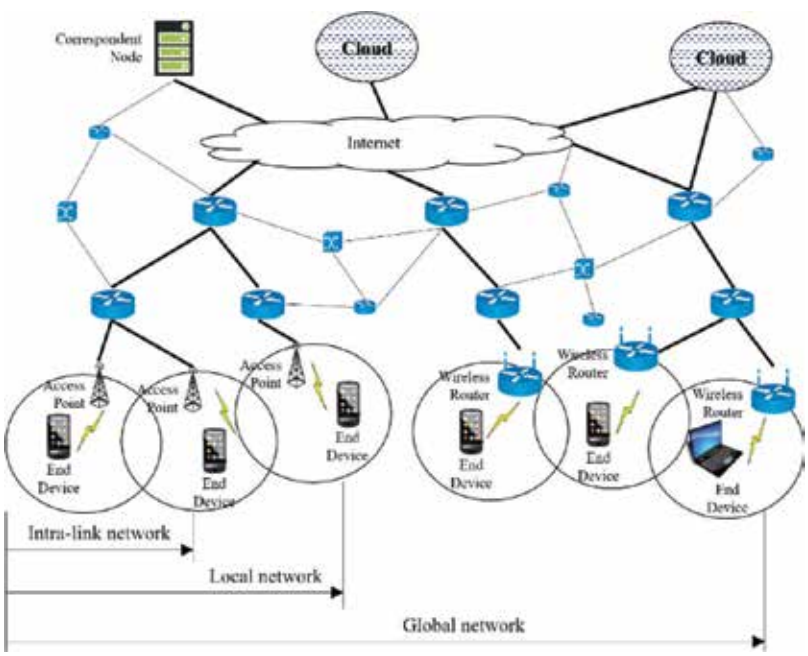

Figure 3. Conventional cloud.

http://dx.doi.org/10.5772/intechopen.75818

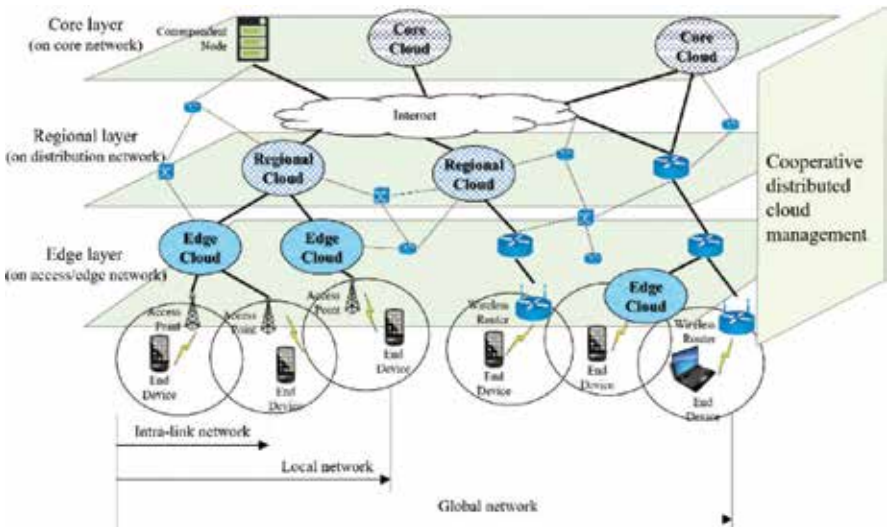

Figure 4. Distributed cloud.

2.3. Benefits of distributed cloud

The distributed cloud is emerged in order to support high real-time or high-mobility constraint of cloud services including augmented reality/virtual reality (AR/VR), autonomous car, and so on. The distributed cloud has some main benefits to provide cloud services as followings:

- Location transparency: A cloud service is provided from any cloud node, that is, an edge cloud, a regional cloud, a core cloud, or combination of them. Location of the cloud services can be hidden from a cloud service customer.
- High real-time: A cloud service on conventional cloud environment is provided through far from the network distance. It leads to nondeterministic network latency. However, a cloud service on the distributed cloud environment can be provided from edge cloud located in the proximity of the cloud service customers, so that it can guarantee the QoS of high real-time applications.
- High mobility: In order to support mobility in conventional cloud, the path of a cloud service is changed, but cloud service is still provided from a main cloud (i.e., core cloud). However, on the distributed cloud environment, cloud services and data can be floated by following target mobile devices.
- Load detachability: When a cloud service customer requests a cloud service to a core cloud on the distributed cloud environment, the edge clouds and the regional clouds between the core cloud and the cloud service customer can be aware of the cloud service and accomplish the cloud service instead of the core cloud. It leads to reducing network traffic in the vicinity of the core cloud.

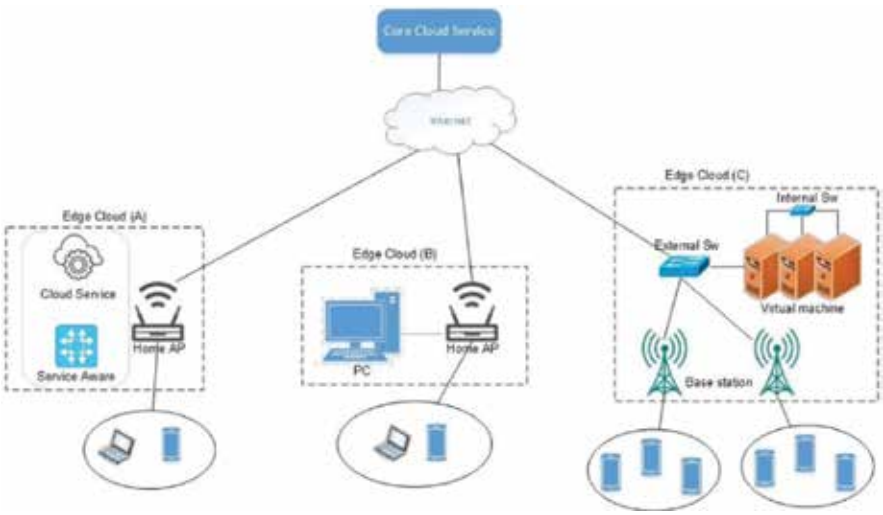

Figure 5. Edge cloud types.

2.4. Edge cloud deployment

In this section, we will take into account a network node in a distributed cloud, so-called regional cloud or edge cloud. Basically, an edge cloud consists of two main parts: network cloud service, and service aware module. Network function has responsibilities to receive/send signal through antenna and route packages to destination. A cloud service is packaged under an instance (e.g., docker instance) and automatically installed in edge cloud. A cloud service has its own local IP address. Service aware module helps edge cloud to filter cloud service packages (if cloud service exists) and forward to the cloud service in edge cloud. The other packages will route to destination normally. **Figure 5** describes three different types of edge cloud. Network of edge cloud can be a wireless access point at home or office or a base station to provide cellular network. To deploy cloud service, we present three ways:

1. A cloud service is packaged as an image and embedded inside network node (e.g. router).
2. A PC is used to run cloud service and connect to router through Ethernet port.
3. A virtual machine technology (e.g., docker, etc.) is exploited to execute cloud services. Virtual machines communicate with each other by internal switch and connect to base station (antenna) by an external switch. This way they can make edge cloud scalable depending on the number of incoming requests, and often use them for cellular network.

3. Distributed cloud-based applications

The trending edge cloud technology will leverage many new mobile services that are equipped with cloud media data. In this section, we introduce some significant applications which are assisted by distributed cloud.

3.1. VR-assist rental service

Finding an apartment/house to rent can be a time-consuming process. Virtual reality tours can help tenants to view a range of properties without ever needing to set foot in the actual house.

Deploying VR-assist rental service on conventional cloud, all databases of in/out-side house viewing are stored at core cloud, and it will cause a network congestion if many tenants access this service at the same time. Distributed cloud is a key solution to greatly mitigate network congestion and enhance performance of this service by distributing geography-based data to edge cloud. Each edge cloud will handle a part of the database from all houses located in same area (as shown in **Figure 6**). Tenants in every area tend to look up apartments/houses in their area instead of other areas. In case of looking for apartments/houses in another area, the edge clouds will communicate with each other to find matched data for tenants. We believe that this kind of application could be popular in forthcoming years as distributed cloud is mature.

3.2. Video streaming

Another application is video streaming in VANET. The application allows watching movie or enjoying multimedia in car/bus/limousine or even autonomous car while moving. The current popular solution is to use plugin DVD player to play in-car video resources. This solution is quite simple, but it requires given video sources and needs to update frequently. The second solution is to use phone/tablet to watch online favorite video through LTE connection. This solution can cause network congestion as many vehicles try to access this service at the same time. Transferring video data costs high bandwidth network that causes congestion easily and increases service delay. By applying distributed cloud, each base station will become edge cloud and can stream video by itself (as shown in **Figure 7**). With the benefit of mobility and caching, distributed cloud supports arranging edge clouds along user path in order to reduce latency, handover, and enhance quality of service.

Figure 6. VR-assist rental service.

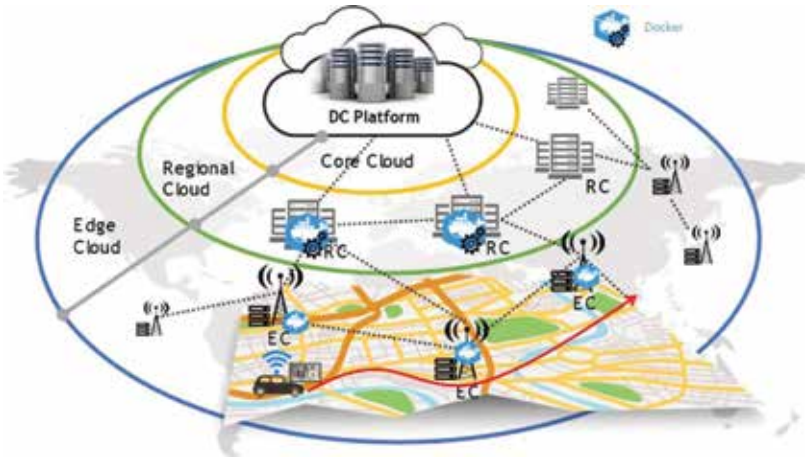

Figure 7. Video streaming.

3.3. Emergency care assistant

An Emergency Care Assistant (ECA) is a system that assists doctors to monitor and save patients during the journey to hospital (**Figure 8**). The challenge of this ECA is how to deliver symptoms and signs of disease to cloud so that the core service will analyze this information from medical knowledge and patients data to diagnose and give some treatments in real time. Besides, all symptoms and signs of patients are also sent to hospital so that doctors can monitor patients on the way to hospital and give the urgent treatment right away. Thus transferring information is very important and requires low latency. Therefore, with distributed cloud, edge cloud, and regional cloud will be arranged along the path to hospital to receive data from patients, analyze and respond right away.

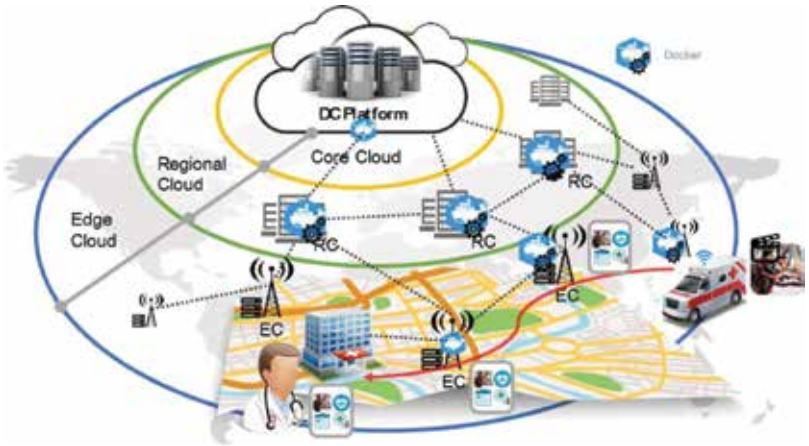

Figure 8. Emergency health assistant.

http://dx.doi.org/10.5772/intechopen.75818

4. Open research challenges

Although its promises and advantages as discussed above, the distributed cloud faces a new set of fundamental challenges that need to be addressed.

4.1. Context aware

The concept of context-awareness appeared initially in the field of ubiquitous computing, where devices and sensors are distributed pervasively within a given environment. In [19], context is defined as any information of environment that affects to user and the system, and even condition of user and the system (such as battery remaining, or number of user connecting to the system, etc.) can be considered as a context. Using the context information, the performance of services is much enhanced and the system could understand users' situation to satisfy quality of experience (QoE).

Contextual information in distributed cloud can be categorized as environmental context, application context, user context, device context, and network context. Environmental context includes location of user and time user request. Application context is related to latency of application and type of application (focus on computing or database). User context is mobility, activities, and even interactions with other users. Device context contains information of user devices including CPU, memory, and battery remaining. Network context is the bandwidth of the network and connection signal (Wi-Fi or cellular signal).

To efficiently deploy a distributed cloud, the system has to be aware of all the above contexts to distribute cloud services to edge nodes efficiently so that we can save energy and resources as well as guarantee quality of service.

4.2. Caching

Caching in many research areas has been proved beneficial [21]. In distributed cloud, caching is also a key challenge because cloud services and data are distributed to various places in the proximity of users instead of storing at core cloud far away from users. In distributed cloud, not only data are cached, but cloud services are cached as well. If the number of available cloud services is large, users will receive response from edge cloud in low latency. However, it is a trade-off between performance and economic benefit. The cloud resources of edge cloud are typically limited and costly. Accordingly, we need to achieve an efficient strategy to gain maximum benefit. Another issue related to caching is synchronization. How we can keep updating version of cloud service at all edge clouds and synchronizing data between edge/regional/core clouds of the same cloud service? This synchronization issues needs to be considered carefully.

4.3. Heterogeneous compatibility

Similar to mobile cloud computing, in distributed cloud, we have to deal with various types of heterogeneity, in which the following two main heterogeneity needs to be focused on:

- Heterogeneity in infrastructures: Core, regional, and edge clouds provide different services with custom-built policies, infrastructures, platforms, and APIs that make the cloud landscape heterogeneous. For example, Cloud service is running Linux-based core cloud, but if edge cloud uses ARM-based, the cloud service cannot execute properly and it needs to be recompiled in order to support compatibility.
- Heterogeneity in network: integrate a wide variety of network technologies for radio access with wireless and wired transport solutions interconnecting a huge number of vastly different end devices and users. For example, if the conventional cloud service is running on core cloud that is equipped with high bandwidth network, running cloud service on edge cloud that is equipped Wi-Fi or cellular signal needs to be considered.

In such a non-uniform environment like distributed cloud, facilitating interoperability, compatibility, and portability among heterogeneous clouds is considered.

4.4. Self-deployed service

The next generation of cloud systems will have the ability to automatically provision capacity and distribute workload dynamically in a cloud-to-region-to-edge continuum [22, 23]. Depending on number of incoming requests or history data, edge/regional cloud will be able to preemptively take reconfiguration and remediation actions in a fully automatic fashion to make distributed cloud more efficient.

On the other hand, distributed cloud needs to build a standard API so that core cloud can deploy their cloud services on edge/regional cloud by only a function call. It means that cloud providers just only build their cloud services into images and put them on cloud. Whenever edge cloud wants to execute a cloud service, it just downloads that image and runs quickly. Simplifying such self-deployed service will attract more and more cloud providers to join distributed cloud.

4.5. Service routing

In conventional cloud, mobile application has to ask where the cloud service server is at the initial time and then connect to that service. Contrarily in distributed cloud, mobile application just sends request to cloud service normally, and edge cloud will detect the nearest destination that is running the requested cloud service. To deploy this kind of service, we need to construct a good service routing mechanism so that edge/regional cloud can easily find the target cloud service in short time. A combination of information centric networking (ICN) [24] and software defined network (SDN) [17] can be considered to shape an efficient service routing mechanism for distributed cloud.

4.6. Cooperative orchestration

Integrating a cloud service into a mobile network environment brings a number of challenges related to service orchestration, mainly due to the fluctuation of resources and the evolving radio and network conditions caused by user mobility. Since user mobility across edge/regional clouds cooperating the same cloud service, or even different cloud service on different edge/regional clouds, emerges as an open challenge, how to share current status and variables between cloud services, how to let other clouds know timestamp of video streaming

when user moves to other areas, are things that need to be considered for assuring efficient network resource utilization, QoE, and reliability.

4.7. Hierarchical distribution

A hierarchical network topology enables mobility feature as well as provides the necessary computation power for latency-tolerant scenarios at the higher tiers and minimum delay for delay-sensitive applications at the lower tiers. To efficiently manage the whole system, it is necessary to build a good protocol between core-regional-edge cloud so that we can save more energy and avoid wasting cloud resources.

On the other hand, an open research challenge is using peer-to-peer network model for distributed cloud instead of hierarchical topology. By this way, managing cloud nodes will get more difficult as well as in routing for cloud service. However, peer-to-peer topology has its own advantage such as not depending on core cloud and reducing network congestion on backhaul link. Therefore, it has promising potential in distributed cloud.

5. Conclusion

The cloud computing is on the way to the next generation cloud, so-called distributed cloud. Distributed cloud is an emerging technology that brings forward the technical benefits of edge cloud computing with networking and support multi-tenancy allowing third party cloud providers to provision applications and services on-demand through standardized APIs. In this chapter, we have shown that the impact of distributed cloud in reducing latency, network congestion, and user mobility. Besides, some mobile applications are described to emphasize the necessity of distributed cloud. We have also introduced some significant open research challenges in distributed cloud in order to improve the next generation cloud. Eventually, considering its potential, it is obvious that distributed cloud will significantly leverage the mobile application and mobile communication.

Acknowledgements

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2017-0-00294, Service mobility support distributed cloud technology).

Author details

Tien-Dung Nguyen, Yunkon Kim, Xuan-Qui Pham, Tri D.T. Nguyen and Eui-Nam Huh*

*Address all correspondence to: johnhuh@khu.ac.kr

Department of Computer Science and Engineering, Kyung Hee University, South Korea

References

[1] Dinh HT, Lee C, Niyato D, Wang P. A survey of mobile cloud computing: Architecture, applications, and approaches. Wireless Communications and Mobile Computing. 2013;**13**. DOI: 10.1002/wcm.1203

[2] Sonam S, Sarv PS. A survey on latency reduction approaches for performance optimization in cloud computing. International Conference on Computational Intelligence & Communication Technology. (CICT). 2016;5. DOI: 10.1109/CICT.2016.30

[3] Chen X, Jiao L, Li W, Fu X. Efficient multi-user computation offloading for mobile-edge cloud computing. IEEE/ACM Transactions on Networking. 2016;**24**(5). DOI: 10.1109/TNET.2015.2487344

[4] Corcoran P, Datta SK. Mobile-edge computing and the internet of things for consumers: Extending cloud computing and services to the edge of the network. IEEE Consumer Electronics Magazine. 2016;**5**(4). DOI: 10.1109/MCE.2016.2590099

[5] Pathak A, Wang A, Huang C, Greenberg AG, Hu YC, Kern R, Li J, Ross KW. Measuring and evaluating tcp splitting for cloud services. In: PAM. Vol. 10. Springer; 2010;(10). DOI: 10.1007/978-3-642-12334-4_5

[6] Satyanarayanan M, Bahl P, Caceres R, Davies N. The case for vm-based cloudlets in mobile computing. IEEE Pervasive Computing. 2009;**8**(4). DOI: 10.1109/MPRV.2009.82

[7] Cai Y, Yu FR, Bu S. Dynamic operations of cloud radio access networks (c-ran) for mobile cloud computing systems. IEEE Transactions on Vehicular Technology. 2016;**65**:1536-1548. DOI: 10.1109/TVT.2015.2411739

[8] D'ıaz M, Mart'ın C, Rubio B. State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing. Journal of Network and Computer Applications. 2016;**67**:99-117. DOI: 10.1016/j.jnca.2016.01.010

[9] Wu D, Liu X, Hebert S, Gentzsch W, Terpenny J. Democra- tizing digital design and manufacturing using high performance cloud computing: Performance evaluation and benchmarking. Journal of Manufacturing Systems. 2017;**43**(10). DOI: 10.1016/j.jmsy.2016.09.005

[10] Bonomi F, Milito R, Zhu J, Addepalli S. Fog computing and its role in the internet of things. In: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing. 2012;(4). DOI: 10.1145/2342509.2342513

[11] ETSI. Mobile-Edge Computing. Introductory Technical White Paper. 2014

[12] Satyanarayanan M, Lewis G, Morris E, Simanta S, Boleng J, Ha K. The role of cloudlets in hostile environments. IEEE Pervasive Computing. 2013;**12**:40-49. DOI: 10.1109/MPRV.2013.77

[13] Bahl V. Microsoft. Emergence of Micro Datacentre (Cloudlets/Edges) for Mobile Computing [Internet]. May 13, 2015. Available from: http://research.microsoft.com/en-us/um/people/bahl/Present/Bahl_mDC_Keynote_May.pdf [Accessed: Dec 6, 2017]

[14] Taleb T, Samdanis K, Mada B, Flinck H, Dutta S, Sabella D. On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration. IEEE Communications Surveys & Tutorials. 2017;**19**:1657-1681. DOI: 10.1109/COMST.2017.2705720

[15] Wu J, Zhao Q, Yang N, Duan J. Augmented reality multi-view video scheduling under vehicle-pedestrian situations. In: International Conference on Connected Vehicles and Expo (ICCVE); Shenzen. 2015. pp. 163-168. DOI: 10.1109/ICCVE.2015.75

[16] Nekovee M. Radio technologies for Spectrum above 6 GHz – A key component of 5G. In: 5G radio technology seminar. Exploring Technical Challenges in the Emerging 5G Ecosystem (Institution of Engineering and Technology); London. 2015. pp. 1-46. DOI: 10.1049/ic.2015.0030

[17] Benzekki K, El Fergougui A, Elbelrhiti Elalaoui A. Software-defined networking (SDN): A survey. Security and Communication Networks. 2016;**9**:5803-5833. DOI: 10.1002/sec.1737

[18] Bi Y, Zhou H, Xu W, Shen XS, Zhao H. An efficient pmipv6- based handoff scheme for urban vehicular networks. IEEE Transactions on Intelligent Transportation Systems. 2016;**17**:3613-3628. DOI: 10.1109/TITS.2016.2584079

[19] Perera C, Zaslavsky A, Christen P, Georgakopoulos D. Context aware computing for the internet of things: A survey. IEEE Communications Surveys & Tutorials Journal. 2013;**16**(40). DOI: 10.1109/SURV.2013.042313.00197

[20] Redowan Mahmud, Ramamohanarao Kotagiri, Rajkumar Buyya. Fog computing: A taxonomy. Survey and future directions. Internet of Everything. Internet of Things (Technology, Communications and Computing), Springer. 2017;(27). DOI: 10.1007/978-981-10-5861-5_5

[21] Zhang M, Luo H, Zhang H. A survey of caching mechanisms in information-centric networking. IEEE Communications Surveys & Tutorials. 2015;**17**:1473-1499. DOI: 10.1109/COMST.2015.2420097

[22] Östberg PO et al. Reliable capacity provisioning for distributed cloud/edge/fog computing applications. 2017 European Conference on Networks and Communications (EuCNC); Oulu. 2017. pp. 1-6. DOI: 10.1109/EuCNC.2017.7980667

[23] Landa R, Charalambides M, Clegg RG, Griffin D, Rio M. Self-tuning service provisioning for decentralized cloud applications. IEEE Transactions on Network and Service Management. 2016;**13**:197-211. DOI: 10.1109/TNSM.2016.2549698

[24] Ahlgren B, Dannewitz C, Imbrenda C, Kutscher D, Ohlman B. A survey of information-centric networking. IEEE Communications Magazine. 2012;**50**:26-36. DOI: 10.1109/MCOM.2012.6231276

Section 2

Latest Trends in Mobile Computing

Chapter 4

Reliable Web Service Consumption Through Mobile Cloud Computing

Amr S. Abdelfattah, Tamer Abdelkader and
El-Sayed M. El-Horbaty

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.74461

Abstract

The mobile intermittent wireless connectivity limits the evolution of the mobile landscape. Achieving web service reliability results in low communication overhead and correct retrieval of the appropriate state response. In this chapter, we discuss and analyze two approaches based on middleware approach, Reliable Service Architecture using Middleware (RSAM), and Reliable Approach using Middleware and WebSocket (RAMWS). These approaches achieve the reliability of web services consumed by mobile devices and propose an enhanced architecture that achieves the reliability under various conditions with minimum communication data overhead. In these experiments, we covered several cases to prove the achievement of reliability. Results also show that the request size was found to be constant, the response size is identical to the traditional architecture, and the increase in the consumption time was less than 5% with the different response sizes.

Keywords: web service, WebSocket, mobile consumption, middleware, timeout problem

1. Introduction

The evolution of the mobile landscape coupled with the Internet nature and the recent explosion of the cloud computing technology is facilitating the deployment of web services. The web services are the perfect way to provide a standard platform for mobile application communication through the Internet.

Smartphones are gradually becoming the effective client platform to consume the services and the pool of data and information [1].

The mobiles are uncomfortable because of their limited processing power and intermittent wireless connectivity. In terms of that, there is an uncertainty of whether the web service request was successfully received, was lost on the Internet before reaching the server, or was partially processed. In the case the application retries the operation and resends the request, it may be duplicated or cause an error, such as two orders entered or two credit card charges.

Most of the cloud-oriented services and data are deployed as web services that are network-oriented applications [2]. The synchronization between a mobile device and a web service is achieved through initiating a conversation in a request-response pattern.

Mobile cloud computing (MCC) is the combination of cloud computing, mobile computing, and wireless networking to bring rich computational resources to mobile users, network operators, as well as cloud computing providers. That enables the applications that could not run on the mobile device due to the mentioned constraints and can now be delivered to users of these devices [3].

The reliable web services through mobile cloud computing is achieved using approaches that focus on ensuring the request execution under the intermittent connectivity and service unavailability conditions, moreover ensuring the appropriate response according to the request state.

1.1. Web service types and limitations

The web service architecture (WSA) [4] proposed by W3 organization relies on a number of web standards, such as Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL), Extensible Markup Language (XML), and Universal Description Discovery and Integration (UDDI) that allow services to be searched, described, and integrated by any application [5].

There are two types of web services: Simple Object Access Protocol (SOAP) and Representational State Transfer (REST).

SOAP web service consumption requires extra effort mostly due to the lack of native support. The two major SOAP-based web service limitations are categorized mainly to communication and computation overhead [5].

In the REST architecture, standardized interface and protocol are used for representing resources between clients and servers. These principles encourage REST applications to be simple, be lightweight, and have high performance. Therefore, regarding the scope of reliability, RESTFUL services overcome SOAP service limitations and achieve better results especially in mobile communications [6–8]. Using REST as service architecture is preferred for mobile devices because REST services use HTTP request and response, which means that a mobile device connected with the Internet can access the service without additional overhead, unlike SOAP web services [9].

1.2. WebSocket

The WebSocket protocol is a full-duplex protocol over a TCP connection that typically provides bidirectional communication between web browsers and web servers. It is used to facilitate the real-time data transfer from and to the server [10].

1.3. Reliability and challenges

The challenges in web service consumption using mobile client are worth analyzed from two perspectives: mobile limitations and connectivity limitations from both mobile and cloud service provider sides. The challenges are listed in the following points:

- Connection loss: The smartphones perspective, clients have an intermittent connection because of their mobility. They can be momentarily removed from the connected network and later join the available network [9]. From the perspective of cloud/server, it may lose the connection, and their web services become unreachable from clients.
- Latency/bandwidth: The very limited bandwidth mobile cellular networks are often billed based on the data transferred amount.
- Timeout: The service timeout problem is one of the issues in the mobile experience, such that the service response size, database relations' communication time, and the required computations in the services are continuously increasing corresponding to the application usage during the time, which causes repeat timeout through the consumption of these services.

1.4. Reliable web service approaches

Middleware approach and mobile agent (MA) approach are the most recent approaches that are used for achieving the reliability of web services.

In this chapter, we will discuss, analyze, and focus on the middleware-based approaches.

2. Background

Different architectures are implemented based on the following the middleware approach. The middleware is applied in more than context with a different purpose.

2.1. Middleware approach

The middleware component is responsible for retrieving the response from the web service, such that it acts as a gateway that communicates lightly with the client.

The middleware solution for mobile clients mostly focuses on application and content adaptation. The proposed communication architecture introduces a gateway between the mobile client and the web service that takes the heavy load communication with the service. The mobile client will instead have to sustain a lightweight and simple client–server communication over a fast binary protocol [11].

a. Middleware architecture

The middleware architecture, as shown in **Figure 1**, consists of the following three components:

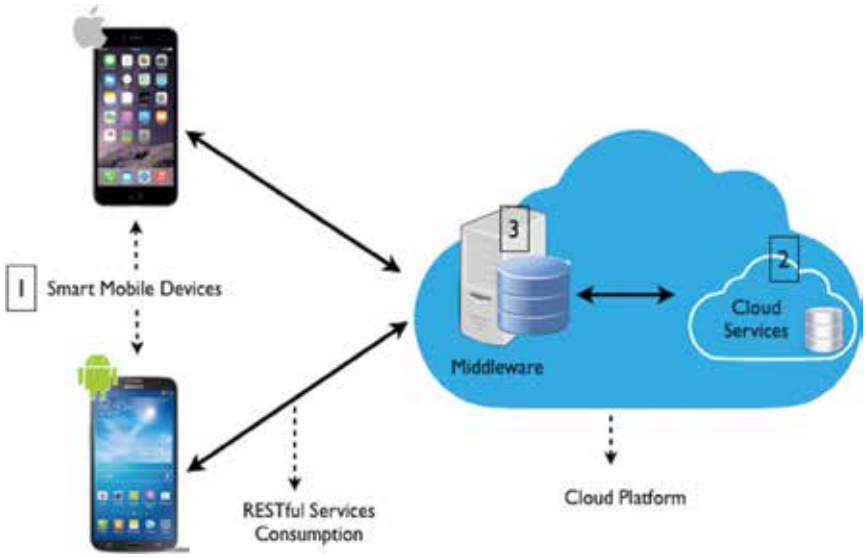

Figure 1. Middleware architecture and the communication process between its components.

1. A mobile device that has Internet access.
2. A web application that contains web services connected to a database.
3. A web application that represents the middleware contains web service consumption and data handling modules.

b. Middleware advantages

The middleware advantages are summarized as follows:

- Small bandwidth usage because of the light communication with the mobile client.
- This architecture brings more opportunities toward ensuring a more reliable communication with the web service such as:
  - The middleware will often run on dedicated hardware that will justify the search for solutions to ensure some kind of state of the communication with a web service.
  - Retry mechanisms can be explored in case of connection failures.
  - The middleware can retain the state of the overall communication and retry to continue when all parties come back online, such as in the case of communication failures (mobile device to middleware or middleware to web service).

c. Middleware limitations

The main middleware limitation is the increasing of the overall duration of a request to the web service and to process the data format, but deploying the middleware and the actual web service in the same server instance network can enhance this.

http://dx.doi.org/10.5772/intechopen.74461

3. Reliable service architecture using middleware (RSAM)

This section describes the proposed architecture RSAM [12]. In this section, the proposed architecture is presented and how it is used to improve the web service consumption reliability through mobile cloud computing. It also illustrates how the proposed approach overcomes the web service consumption and their reliability limitations mentioned in the above sections.

The enhanced middleware architecture focuses on the integration between the mobile client and service layer, so the architecture is defined as client middleware component and service middleware component. This integration increases the system awareness for each request state to be notified to the user appropriately.

3.1. RSAM architecture

In this architecture, the mobile consumer component has the permission to consume a cloud service directly without passing through the middleware, which creates the flexibility to customize system communications.

The enhanced architecture components and their communications are shown in **Figure 2**, which contain two main components:

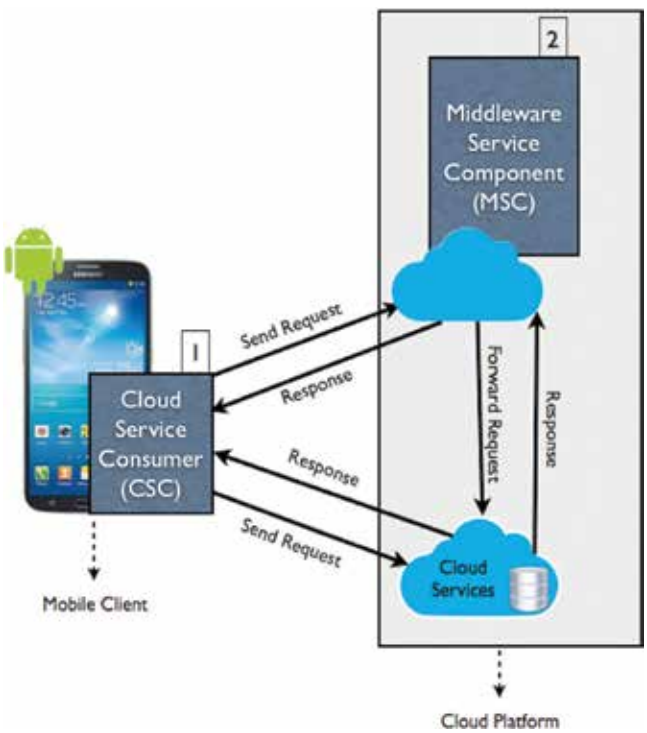

Figure 2. RSAM architecture and the communication process between its components.

- Cloud service consumer (CSC) is a mobile client middleware component responsible for:
  - Constructing the request with its appropriate attributes to be ready for sending
  - Handling the request communication cycle either to the cloud service directly or through the service middleware component
  - Receiving the response and notifying the client with the appropriate state
- Middleware service component (MSC) is a cloud service middleware component responsible for:
  - Receiving the request from the client service consumer and constructing the appropriate response regarding the sent request attributes.
  - Communicating with the required cloud service.
  - Caching the request and response states to be tracked for later usage.
  - A detailed description of the architecture is in the following subsections.

3.1.1. Cloud service consumer (CSC)

The CSC is the client middleware component responsible for handling the request communication cycle starting from the service call till the response notification.

There are attributes that affect the consumer and support it to track and handle the consumption process as a separate component, as shown in the following:

- Service descriptor is the module that contains the routing mechanism regarding the base cloud service URL and the middleware component URL and has the description of each service in the system. This descriptor consists of:
  - Basic attributes that describe the services, such as the service name, request type, parameter names, parameter types, and expected response type with its appropriate parser.
  - Semantic attributes that affect the behavior of the execution, such as:
    - The forced attributes that force the MSC to use the succeeded cached results or forward the request to the cloud service each time
    - The direct attribute that controls the request route to the middleware or to the cloud service directly
  - Request client Id is the unique identifier overall requests in the system; it consists of two main parts:
    - Basic part is the unique auto-generated key that consists of subparts that ensure its distinct property, such as mobile device manufacture number, the request timestamp, and the requested service name.

http://dx.doi.org/10.5772/intechopen.74461

- Additional attributes part is the other part that includes labeled attributes that change the MSC behavior regarding the received request such as the request number of trials and forced attribute.

The sequence of the client service consumer is illustrated in **Figure 3**. The following algorithm highlights its steps:

1. [Mobile User] select a specific service in the application to invoke.
2. [Client Id generator] generate valid client id as designed in the middleware validation method.
3. [Service Descriptor] construct the request with its specific attributes.
4. [Service Consumer] send the request to the middleware or direct to the specified cloud service as attributed. [Service Client Manager] update the request attributes in the persistence storage based on the response state.
5. [Service Client Manager] adapt the response based on its state.
6. [Service Client Manager] notify the user with the appropriate response that shows the suitable state.

This sequence is performed in the mobile client part in two stages:

1. Preparation: The CSC prepares and constructs the appropriated request with its attributes, in order to be fit in the middleware request validation agreement.
2. Consumption: The stage of consuming the specific web service using the constructed request through middleware and receiving the appropriate response to show it to the user in its reliable form.

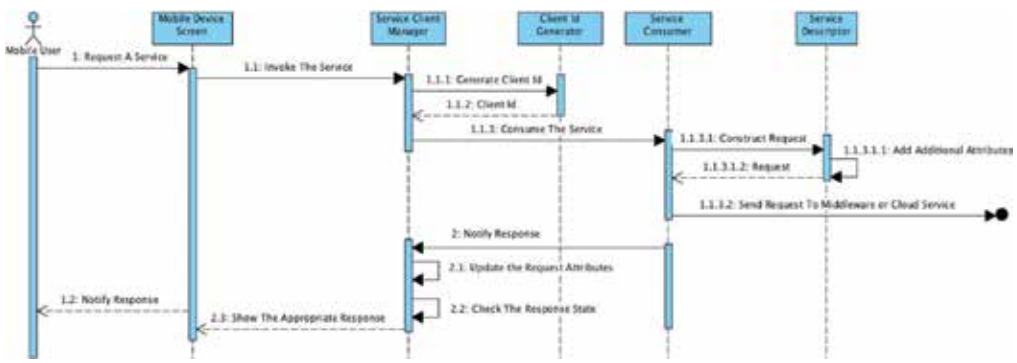

Figure 3. Cloud service consumer sequence.

3.1.2. Middleware service component (MSC)

The cloud middleware component is the other integrated part with the cloud service consumer, such that it is responsible for handling the received request and proceeding to the appropriate function to send the response back to the client consumer.

The middleware component sequence is shown in **Figure 4**. It begins acting once a client request is received.

The sequence of the service middleware component is illustrated in **Figure 4**. The following algorithm highlights its steps:

1. [Mobile device] sends a request.
2. [Middleware request filter] receive the request to ensure it is included in the allowed ones.
3. [Request manager] initialize the request holder.
4. [Parser Manager] parse the request to extract the attributes included in the request.
5. [Validation Manager] ensure the request and the attached client id attribute validity.
6. [Database Manager] save the validated request in the persistence storage.
7. [Web service Consumer] consume the attached cloud service using the sent request.
8. [Database Manager] save the response in the persistence storage with a relation with the saved request.
9. [Request Filer] respond to the mobile device with the response.

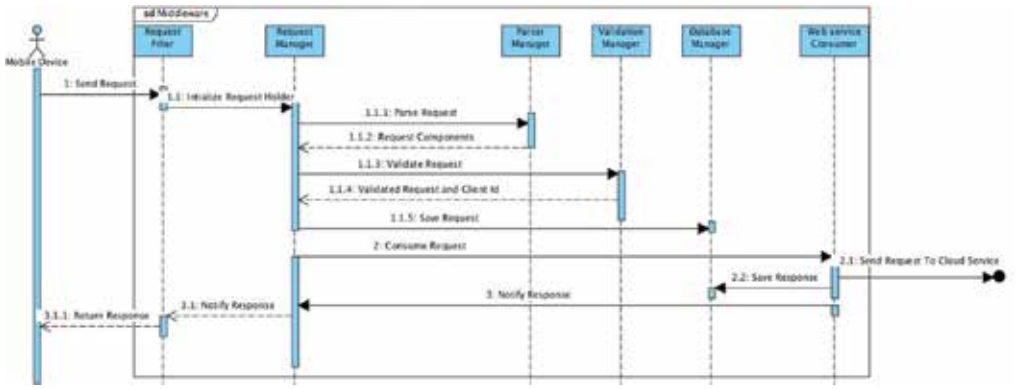

Figure 4. Service middleware component sequence.

This process follows three stages:

i. Pre-processing: The service middleware component parses the request to extract the client identifier and its attributes and then validates the request to ensure whether it is correct and secure or not.

ii. Processing: In the case of request validation success, the service middleware retrieves the similar previous cached request or saves this request details in its cache if it did not exist before; then there are two possible cases to follow regarding the different request cases:

a. Forward the request to the cloud service if:

   i. It was the first time invoking this request.

   ii. It is intended for the client consumer to retry this request.

   iii. The request is labeled with the forced attribute.

b. Return the middleware component cached results, if this request is invoked before with a successful result.

iii. Post-processing: The service middleware component constructs the appropriate response and caches its details according to the action taken in the processing step. It then sends this response to the mobile consumer to notify the user with the appropriate request state.

3.2. RSAM protocol

The enhanced architecture achieves cloud service reliability while considering the most affected cases illustrated in **Table 1**.

The architecture process flow shown in **Figure 5** covers these possible cases, such that it mainly depends on the middleware components in client consumer and middleware to track the state of each request to be able to notify the user with the appropriate response.

The possible response states shown in **Figure 5** contain the following:

- Succeeded and failed states that explicitly indicate the success and failure of the request execution.
- Cached state that informs the user that the response was cached in the middleware from previous execution for the same request. This state covers two cases:
  - Preventing the duplicated request execution
  - Optimizing the time of the request execution if it will get the same response from the cloud service because it no longer needs to be passed to the cloud service again with its complex query

  Doubt state counted as the most significant one that marks the request as doubt request to inform the user to contact the responsible one in order to ensure the request state to prevent its significant duplication

Mobile state	Middleware state	Cloud service state	Case number
Reachable	Reachable	Reachable	1
Reachable	Reachable	Non-reachable	2
Reachable	Reachable	Server error	3
Reachable	Non-reachable/server error	Any	4
Non-reachable	Any	Any	5
Timed out	Reachable	Reachable	6

Table 1. The covered state summary of enhanced middleware architecture.

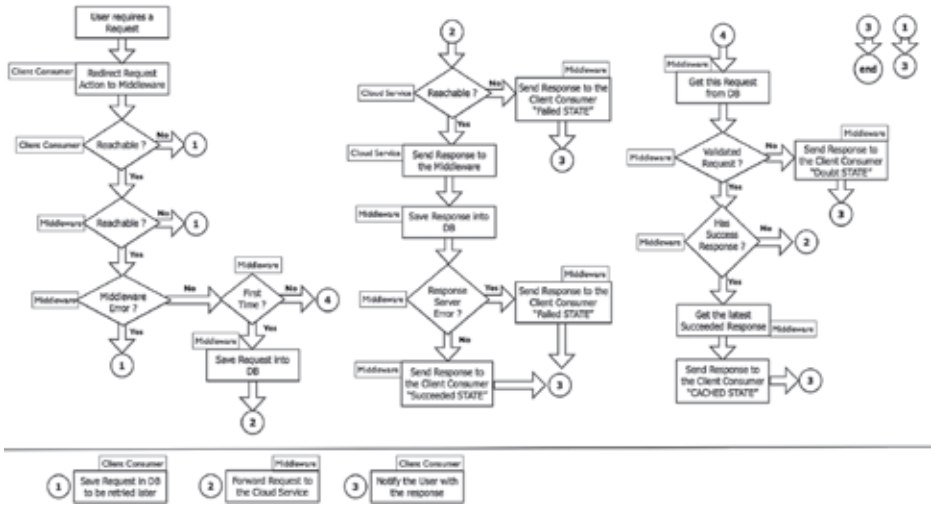

Figure 5. Enhanced architecture process.

Table 1 contains the following states:

- **Reachable**: available through Internet access.
- **Server Error**: The service responds with an internal server error.
- **Timed Out**: The Service execution takes more time than that allowed for the consumer to get the response.

4. Reliable approach using middleware and WebSocket (RAMWS)

This section describes the proposed architecture RAMWS. In this section the proposed architecture is presented and how it is used to achieve the reliable web service consumption in terms of overcoming the timeout problem. It achieves the immediate notification with the appropriate response once it is available to the user.

http://dx.doi.org/10.5772/intechopen.74461

The enhanced approach depends on the integration between the middleware approach and the WebSocket protocol. This integration enables the middleware to always represent the client with its arguments while consuming the required web service and enables the WebSocket to represent the open connection protocol between the server and the client.

4.1. RAMWS architecture

The proposed approach is based on the architecture that was shown in **Figure 6**, which contains three main components:

1. Integrated socket cloud service consumer (SCSC) is a mobile client middleware component responsible for:
   - Constructing the request with its appropriate attributes to send it to the integrated socket middleware service component (SMSC) layer
   - Handling the request communication cycle between the client and the SMSC
   - Managing the client side WebSocket communication with the SMSC

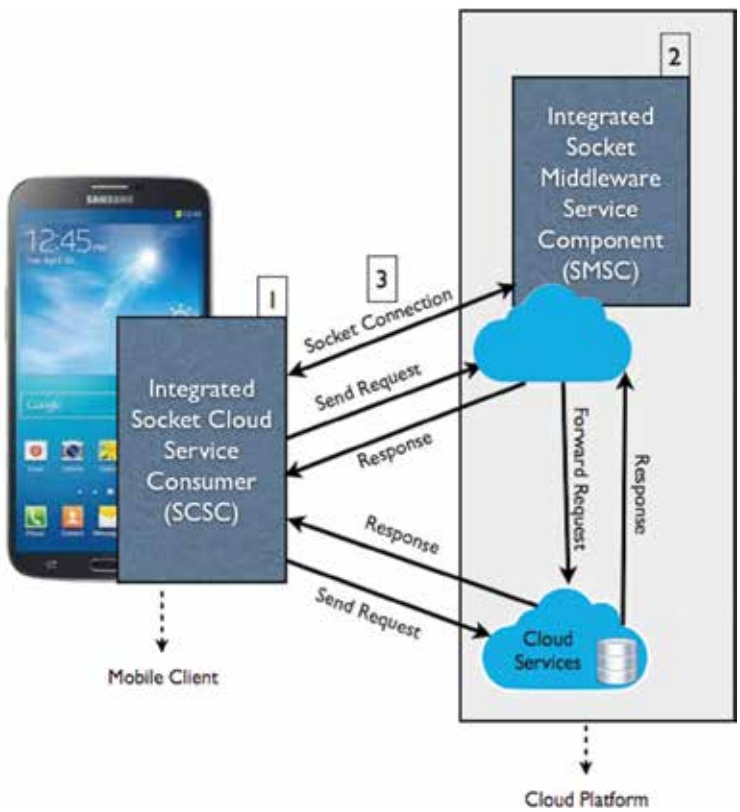

Figure 6. RAMWS architecture and the communication process between its components.

- Receiving the response and notifying the client with the appropriate result based on the service response and the WebSocket data connection

2. Integrated socket middleware service component (SMSC) is a cloud service middleware component responsible for:
   - Receiving the request from the integrated socket client service consumer (SCSC) and constructing the appropriate response regarding the sent request attributes
   - Consuming the required cloud service to get the requested response
   - Managing the server side WebSocket communication with the SCSC layer
   - Handling two-way response to the SCSC, such that short time response to the client request with an appropriate state and actual response through WebSocket connection contain the expected result from the cloud service when available
3. WebSocket protocol is a standard communication way for the server to send content to the client without being solicited by the client. It plays the most important role to overcome the request timeout problem, such that it allows messages to be passed back and forth while keeping the connection open during the service communication time, and it is responsible for:
   - Achieving the other way of communication between the SCSC and the SMSC besides the required REST service communication
   - Sending the actual response to the SCSC once this response returned from the cloud service

4.2. RAMWS protocol

The RAMWS architecture achieves reliability for the heavy cloud service while overcoming the timeout issue.

The architecture process sequence shown in **Figure 7** shows different experiences in mobile applications to prevent the heavy services from the timeout response, such that it mainly depends on two different responses: temporary response for the request and actual response through the socket connection.

The RAMWS protocol contains the following:

1. The request additional attributes: They are attributes that control some important behaviors in the service consumption process, such as:

a. request_id: Identifies the request, to distinguish the responses returned through the WebSocket connection from multiple requests.

b. temporary_response_type: The temporary response is the returned response to be shown in the mobile device till the actual response returns from the cloud service. There are some of the defined types that control the suitable temporary responses for each service, such as latest_cached_response, waiting_message, and limited_cached_response.

http://dx.doi.org/10.5772/intechopen.74461

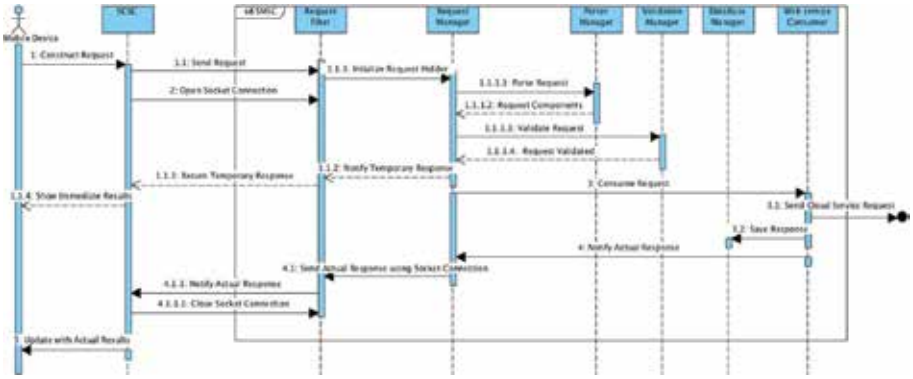

Figure 7. RAMWS protocol sequence.

c. socket_enabled: The is a flag attribute, which flags if this service is heavy and requires this proposed approach, or it is light enough to get the response within the defined timeout without additional overhead

5. Implementation and results

This section is the implemented part of the chapter; it converts that research trial from architecture to implemented framework and shows the used environment setup. The mentioned proposed architectures are applied in the following proof-of-concept case, such that a social network mobile application called "Social Contacts" allows the user to send a message to his/her mobile contacts and show customized messages regularly during the whole day.

5.1. Social contacts application

This application includes the combination of the different request types and different amounts of data flow as shown in **Table 2** and will be clarified below.

The social contacts modules included in this use case are shown in **Figure 8**.

1. Cloud services modules.

- Authentication module: Allows the user to login to an existed account or register a new account.
- Feeds module: Shows the user the frequently post feeds from his contacts and allows the user to send a new post to the contacts.

2. Middleware module:

- Request states module: This is a middleware-related module that gains the benefits from the enhanced architecture and introduces a new mobile user experience to the sort of reliable applications, such that it contains all sent requests states to presented and be clear to the user action.

Module	Function name	Flow	Request type	Data size	Forced attribute
Authentication module	Login	• The user enters his phone number and chosen password • If the user authorized, the cloud service gets the saved user account to the mobile response; otherwise an error response is sent to the mobile client	Get	Small (only phone number and password)	Yes
	Register	• The user enters his phone number and chosen password • The application takes the permission to read the contacts info and sends them to the server • If the user doesn't exist, the cloud service posts and inserts the user account with its related contacts into database; otherwise an error response is sent to the mobile client	Post	Medium (phone number, password, and all contact numbers)	No
Feeds module	**Send Post**	• The user enters/choses his/her contact phone number to send the post to • The user writes his post text • The cloud service adds this post in the database related with the sent contact	Post	Small- large (regarding the post content size)	No
	Delete Post	• The user choses to delete any post that appears in his/her posts • The cloud service deletes this post from the database	Delete	Small (the post id)	No
	Get Posts Feed	• The user opens or refreshes the screen to show the posts that are sent from and to him • The cloud service gets all the related posts to this user	Get	Medium-large (regarding the number of returned posts)	No
	Get Posts Feed Timeout	• The same functionalities as Get Posts Feed, but it does heavy computations and enlarge the response to consume a lot of time to make a timeout response	Get	Very large response	No

Module	Function name	Flow	Request type	Data size	Forced attribute
[Middleware component]—request states module	**Show Requests**	• The user opens or refreshes the screen to show the requests states and the request details sent from him using any of his/her mobile devices • The middleware service gets all the related requests with their exact states to this user			
	Retry Request	• The user chooses to retry the execution of specific failed request The middleware service posts the request in its reliable corresponding scenario			
	Delete Request	• The user chooses to delete any of his/her sent requests The middleware service deletes this request from the database, in order not to be shown or retried from the user another time			

Table 2. Social contact application functions.

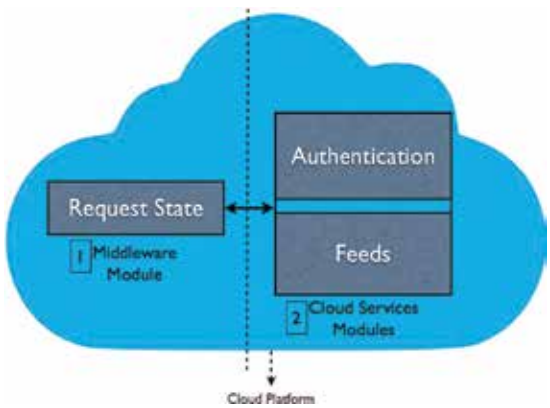

Figure 8. Social contacts modules.

5.2. Environment setup

The working environment of the programming languages and the tools that are used in building these enhanced architectures are explained as follows in **Table 3**.

The following notes regarding making the architecture as a standalone solution:

- The S/CSC is built as an independent and separate android library.
- The middleware and backend components are deployed in a separate cloud instance than the other. This ensures that middleware remains available in the case of any down issues that occur in the backend cloud instance.

Cloud services and middleware service component				
Language and frameworks	**Web service**	**Web socket**	**Mobile agent**	**Database and text format**
Java enterprise edition and hibernate framework [13]	RESTFul (JAX-RS)	Java API for WebSocket (JSR 356) [14]	Java Agent Development Framework (JADE) [15]	MySQL [16] and JSON [17]
Cloud service consumer (client)				
Platform		**Response caching**	**Web socket**	**Text format**
Android		File system	Java WebSocket Client [18]	JSON [17]
Service deployment				
Cloud service provider		**Application server**		
Openshift Cloud [19] for both middleware and backend each on separate instance		Glassfish application server [20]		

Table 3. Environment setup.

5.3. Result assessment

The proposed middleware architecture handles the effective cases that covered the corresponding reliability concept. Moreover, the performance comparison will be considered to ensure the usability and discuss the trade-off factors related to the middleware approach. The following factors are considered as the most affective factors regarding the mobile computing scale:

- Request size: The size of the request body that constructs and sends in the mobile client; it will be affected because of the additional attributes necessarily added for the middleware to ensure the reliable request.
- Response size: The size of the response body that it is received in the mobile client; it will not vary because there are no additional properties that need to be added from the middleware for any reason.
- Consuming time: The time taken from sending the request till receiving its response may be longer because of the additional middleware connection rather than the direct cloud service connection.

5.3.1. RSAM results

Regarding the social contact application functions, **Table 4** and **Figures 7–11** show the performance measurements through middleware component and through direct cloud service connection.

The measurements indicate the middleware component cost performance. Regarding the request size factor, the middleware adds additional 226 bytes to the request. These bytes used for the required attributes discussed in its description section. Although this number of

		Direct cloud connection			Middleware connection		
Module	**Function name**	**Request size (byte)**	**Response size (byte)**	**Consuming time (s)**	**Request size (byte)**	**Response size (byte)**	**Consuming time (s)**
Authentication module	Login Success	55	5	> 0	281	5	> 0
	Failure	53	315	> 0	279	315	> 0
Feeds module	Send Post	20217 (~20K)	3072 (~3 kB)	1	20443 (~20.5K)	3072 (~3kB)	1
	Get Posts Feed	25	191745 (~191kB)	2	251	191745 (~191kB)	2
		25	2167000 (~2MB)	21–26	251	2167000 (~2MB)	22–28
		25	4592949 (~4.5MB)	44	251	4592949 (~4.5MB)	44
		25	6828571 (~7MB)	65–66	251	6828571 (~7MB)	65–67

Direct cloud service results **timeout in each time** in the large response size with heavy required computations; the middleware connection results **timeout for the first time** but returns the response successfully from the middleware storage in the next retry.

Table 4. Social contact application performance measurements.

additional bytes in the request is considered a cost to the mobile client, this cost is low, and its complexity is O(1) as it is constant regardless the original request size.

The response size is the factor that is not changed, as shown in **Table 4**, response size columns. The middleware forwards the response exactly as returned from the cloud service. Therefore, we avoid using transformation overhead and additional cost from the middleware or the mobile client.

The consuming time is one of the most important factors for the mobile applications, which require quick responses. The measurements show that time may slightly vary about 2 s in

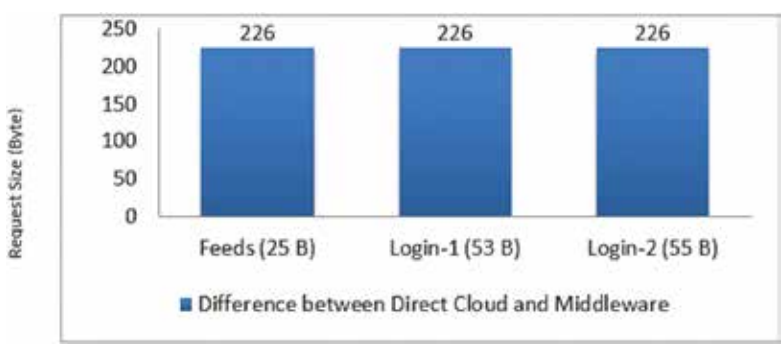

Figure 9. Request size factor of different services regarding the direct cloud and the middleware connection.

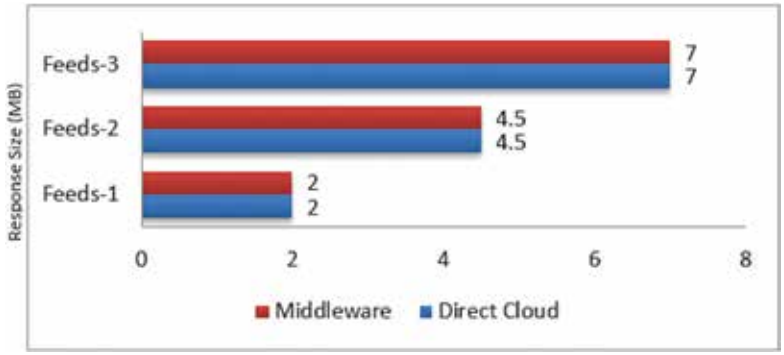

Figure 10. Response size of different services regarding the direct cloud and the middleware connection.

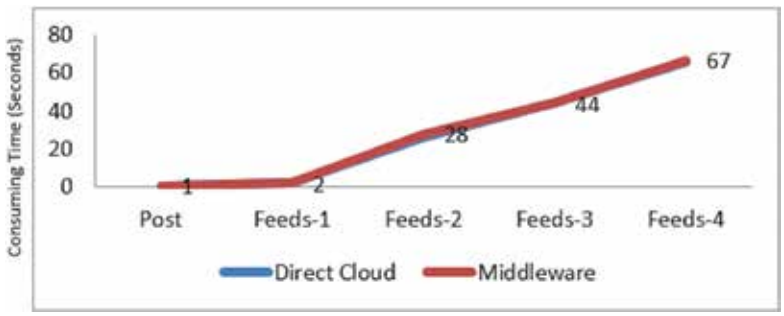

Figure 11. Consuming time for different services regarding the direct cloud and the middleware connection.

the medium response size (~2 MB) data class and the very large response (~7 MB) data class. However, it is exactly the same in the other cases without any overhead due to the middleware connection. This is because the middleware uses lightweight connections and lightweight data formats. In addition to its concerns with the request and response states resolving, the middleware does the minimal efforts in data transformation and data conversion.

5.3.2. RAMWS results

The proposed approach overcomes the timeout problem that threats the cloud service reliability. The two main concepts used to achieve this objective are middleware and the two-response technique. Each of them participates in solving the problem and has some aspects in the other side to be measured and compared with their responsibilities.

As shown in the above sections, the middleware achieves the availability and reliability in responding to the mobile client with the appropriate actions and data. From the cost perspective, it requires extra resources for this middleware handling and deployment, such that it is deployed in an application server, which is located in another cloud instance.

Regarding the two-response technique, it is managed by the middleware using two different protocols: the HTTP REST protocol and the WebSocket protocol, respectively. The first

http://dx.doi.org/10.5772/intechopen.74461

response is a temporary one that enhances the mobile experience by showing appropriate results to the user. After that, in the second response, the user receives the actual results on time through the WebSocket open connection. This technique costs an extra data communication because of the temporary response that is transferred using the REST response. **Figure 12** shows this cost regarding the different temporary response types. The waiting message response type (waiting_message) is the minimum data required to transfer simple message to the mobile client and has constant response size. The limited cached response type (limited_cached_response) sends limited size from the previously cached responses and has small variable response size relative to the size of the single object of data required. The latest cached response type (latest_cached_response) is the latest cached response in the middleware and has a size that is identical to the actual response size of the whole required data. **Figure 13** shows the network consumption difference between the proposed technique and the traditional one, such that the traditional technique consumes the network resource about 2–7 times depending on the network strength and the response size, compared with the proposed technique that uses the network resource without waiting time.

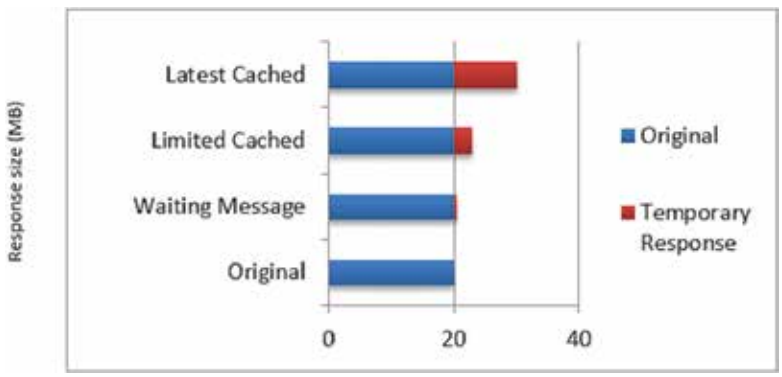

Figure 12. Different temporary response data size with its types compared with the original actual response size.

Figure 13. Network usage time for consuming three different services regarding the traditional and the proposed techniques.

6. Conclusion

In this chapter, we proposed and discussed two middleware-based approaches that achieve reliable web services through mobile cloud computing. These approaches focus on ensuring the service request execution under the different mobile environment conditions including intermittent connectivity and service unavailability conditions. Moreover, it ensures returning the appropriate response according to the request state.

The proposed Reliable Service Architecture using Middleware (RSAM) achieves the reliability by focusing on the request behavior rather than the request structure.

The RSAM focuses on ensuring the request execution and preventing the duplicate request execution as a result of the intermittent mobile connection. In addition, they consider the most important factors for the mobile client such as the request size and the response size for mobile client data transmission limitations and the service consuming time, which is critical for the mobile applications and their usability.

The service response size, database query time, and the required computations in the services are continuously increasing corresponding to the application usage, which causes timeout during the consumption of these cloud services. The timeout problem degrades the mobile usage experience and waste network usage time and threats the cloud service reliability. This problem requires restructuring the cloud service itself or the database schema to overcome such problem.

The RSAM solves this issue in the perspective of getting the response, such that the timeout problem occurs in the first consumption, but it retrieves the ready stored response in the middleware storage in the next retry. The proposed Reliable Approach using Middleware and WebSocket (RAMWS) achieves the web service consumption reliability in the aspect of overcoming the timeout problem. The RAMWS integrates the middleware approach and the two-response technique to achieve the cloud service consumption reliability. The two-response technique is used between the mobile client and the middleware, such that the first response is temporary and shows the appropriate results to the user till the actual results are received on time. The actual response is retrieved through WebSocket open connection once it is fetched using middleware service.

The proposed approach proved to overcome the timeout problem, which occurs in requests through mobile cloud computing, reduces the network usage time, and enhances the mobile experience to be more usable.

Author details

Amr S. Abdelfattah*, Tamer Abdelkader and EI-Sayed M. EI-Horbaty

*Address all correspondence to: amr.elsayed@cis.asu.edu.eg

Faculty of Computer and Information Sciences, Ain-Shams University, Cairo, Egypt

References

[1] Lomotey Richard K, Deters R. Reliable consumption of web services in a mobile-cloud ecosystem using REST. In: IEEE 7th International Symposium of Service Oriented System Engineering (SOSE); 2013. pp. 13-24

[2] Christensen JH. Using RESTful web-services and cloud computing to create next generation mobile applications. In: 24th ACM SIGPLAN Conference Companion Object Oriented Program; 2009. pp. 627-634

[3] O'Sullivan MJ, Grigoras D. Delivering mobile cloud services to the user: Description, discovery, and consumption. In: IEEE International Conference on Mobile Services (MS); USA. IEEE; 2015. pp. 49-56. DOI: 10.1109/MobServ.2015.17

[4] W3C Working Group. Web Services Architecture [Internet]. February 2004. Available from: http://www.w3.org/TR/ws-arch [Accessed: November 2017]

[5] Cobârzan A. Consuming web services on mobile platforms. Informatica Economica. 2010; **14**(3):98-105

[6] Chen M, Zhang D, Zhou L. Providing web services to mobile users: The architecture design of an m-service portal. International Journal of Mobile Communications. 2005; **3**(1):1-18

[7] McFaddin S, et al. Modeling and managing mobile commerce spaces using RESTful data services. In: 9th International Conference on Mobile Data Management, MDM'08; IEEE. 2008. pp. 81-89

[8] Kleimola J. A RESTful Interface to a Mobile Phone [Internet]. January 2008. Available from: http://www.researchgate.net/publication/228974867_A_RESTful_Interface_to_a_Mobile_Phone [Accessed: November 2017]

[9] Gonsai AM, Rushi RR. Enhance the interaction between mobile users and web services using cloud computing. Oriental Journal of Computer Science & Technology. 2014; **7**(3):416-424

[10] Yasumoto K, Yamaguchi H, Shigeno H. Survey of real-time processing technologies of iot data streams. Journal of Information Processing. 2016;**24**(2):195-202

[11] He Y, Salih OS, Wang C-X, Yuan D. Deterministic process-based generative models for characterizing packet-level bursty error sequences. Wireless Communications and Mobile Computing. 2013;**15**(3):421-430

[12] Amr S, Abdelfattah TA, EI-Horbaty EI-SM. RSAM: An enhanced architecture for achieving web services reliability in mobile cloud computing. Journal of King Saud University—Computer and Information Sciences. 2017:1-11. DOI: 10.1016/j.jksuci.2017.03

[13] Redhat. Hibernate Framework [Internet]. Available from: http://hibernate.org [Accessed: November 2017]

[14] Oracle. JavaTM API for WebSocket [Internet]. Available from: https://jcp.org/en/jsr/detail?id=356 [Accessed: November 2017]

[15] Jade (Telecom Italia SpA). JAVA Agent Development Framework [Internet]. Available from: http://jade.tilab.com [Accessed: November 2017]

[16] Oracle. MySQL Documentation [Internet]. Available from: https://www.mysql.com [Accessed: November 2017]

[17] JSON Data format [Internet]. Available from: http://www.json.org [Accessed: November 2017]

[18] Java WebSocket [Internet]. Available from: org.java-websocket: Java-WebSocket [Accessed: October 2016]

[19] Redhat. OpenShift Cloud Provider [Internet]. Available from: https://www.openshift.com [Accessed: November 2017]

[20] Oracle. GlassFish Java Server [Internet]. Available from: https://glassfish.java.net [Accessed: November 2017]

Chapter 5

Validating Activity-Based Travel Demand Models Using Mobile Phone Data

Feng Liu, Ziyou Gao, Bin Jia, Xuedong Yan,
Davy Janssens and Geert Wets

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.75810

Abstract

Activity-based travel demand models predict travel sequences on a day for each individual in a study region. These sequences serve as important input for travel demand estimate and forecast in the area. However, a reliable method to evaluate the generated sequences has been lacking, hampering further development and application of the models. In this chapter, we use travel behavioral information inferred from mobile phone data for such validation purposes. Our method is composed of three major steps. First, locations where a user made calls on a day are extracted from his/her mobile phone records, and these locations form a location trajectory. All the trajectories from the user across multiple days are then transformed into actual travel sequences. The sequences derived from all phone users are further classified into typical patterns which, along with their relative frequencies, define travel profiles. These profiles characterize current travel behavior in the study region and can thus be utilized for assessing sequences generated from activity-based models. By comparing the obtained profiles with statistics drawn from conventional travel surveys, the validation potential of the proposed method is demonstrated.

Keywords: mobile phone data, travel sequences, activity-based travel demand models, travel surveys, travel behavior, travel sequence classification

1. Introduction

Activity-based travel demand models view travel as demand of activity participation. In this modeling framework, travel is analyzed in relation to daily activity behavior, the context of land-use and transportation networks, as well as personal background information

(e.g., socioeconomic conditions) [1]. *Travel surveys*, which collect full daily activities and travel of a small sample of individuals during one or a few days, are also required as training sets. Once the models are built, they can generate *travel sequences* (i.e., chains of activities and travel conducted by a person during a day) of each person in the study area using the Monte Carlo simulation approach. The individual travel sequences are then accumulated across the entire population, resulting in an origin-destination (OD) matrix. In this matrix, each element describes the number of trips between each pair of the corresponding locations of the area. This matrix is further assigned to the road network based on a traffic assignment algorithm, and the number of assigned trips on each road can subsequently be used as important input for mobility-related studies in the region (e.g., travel demand prediction, emission estimate, and transport policy evaluation). **Figure 1** demonstrates the entire process of an activity-based model.

Despite the comprehensive process of activity-based models, a reliable method has been absent to validate the simulated travel sequences. Traditionally, the model results are examined at both internal and external stages of the development process (see **Figure 1**). In the *internal validation*, the statistics aggregated from the simulated sequences (e.g., the average number of trips per day) are compared with those drawn from the expanded survey data that is not used as the training set of the model development but usually collected in the same survey period. Thus, the internal validation suffers from a number of limitations that are intrinsic to the shortcomings of the survey data [2]. In contrast, the *external validation* indirectly evaluates the model results at the traffic assignment stage. The assigned traffic volumes are compared against data from external sources (e.g., traffic counts) on a number of specified roads. However, good outcomes of the compared results might have resulted from the extra processes of OD matrix aggregation and traffic assignment, thus providing no convincing evidence of the accuracy of the model itself.

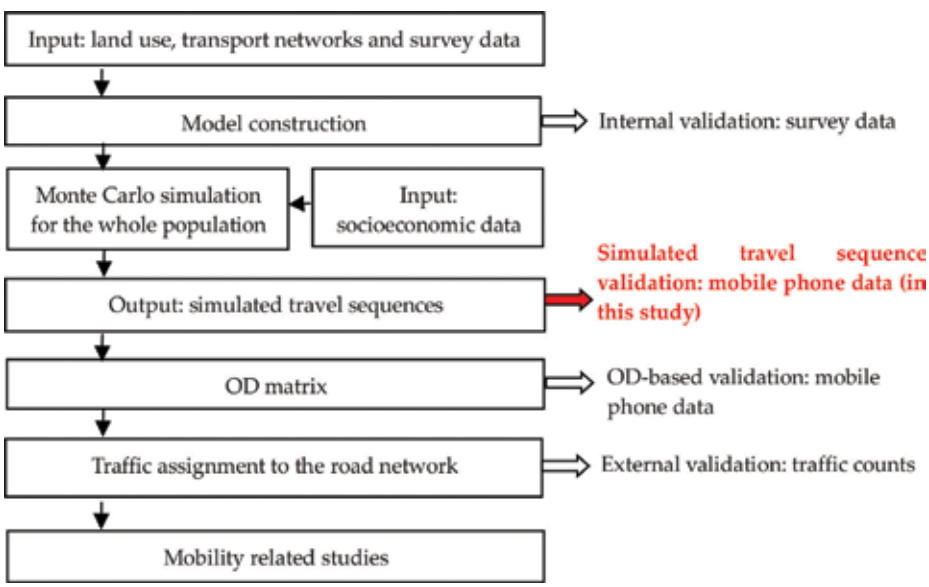

Figure 1. The entire process of an activity-based travel demand model.

Furthermore, if problems are found, it is also challengeable to trace these errors back to the model-building process. Nevertheless, despite the limitations, both internal and external validations are commonly adopted, as no methods have been developed for more accurately validating the model results [3].

The wide spread of mobile phones has offered an opportunity to use the devices as a new data collection method in transportation research. Call location data collected from the devices reflects the latest travel behavior of users, enabling up-to-date travel behavior studies on a large scale. Particularly, mobile phone data has been investigated for validating travel demand models; the study [4] can typify the state-of-art of such research. In this paper, based on the mobile phone data of 1 month recorded from 0.3 million users in Lisbon of Portugal, the two most frequent call locations (cell towers) for each of the users are first identified as home and work locations. An OD matrix is then built, aggregating commuting trips in the morning from home to work over all the users. This matrix is subsequently extrapolated according to a census survey to account for the total number of workers (1.3 million) in the city. Finally, the scaled matrix is compared with the morning travel demand that is predicted by a travel demand model developed in the study area. The results demonstrate the potential and feasibility of mobile phone data in benchmarking travel demand models (see **Figure 1**).

However, despite its advancement by adopting mobile phone data, the OD-based approach does not take into account the sequential information encoded in the call location patterns. It has been well documented that the choices of activities on a day are dependent on each other [5], as shown by the observation that the activity chain of having breakfast, travel, and working is often performed together on a working day. Furthermore, while the activity of bringing/getting people (e.g., bringing/getting children to/from schools) is usually conducted on the commuting ways, leisure is more executed in the evening. The interdependencies and temporal sequencing of daily activities have been regarded as a key factor in travel decision-making processes. The examination of how the simulated travel sequences are compatible with the sequential characteristics observed from the call location patterns is therefore important [6].

Addressing the above described limitations, our study proposes a new approach that utilizes the sequential characteristics of activity and travel behavior. Specifically, this approach first derives actual travel sequences from mobile phone data of all users. A set of *typical patterns* are then defined, each of which represents a certain class of the actual travel sequences. Profiles consisting of relative frequencies of the typical patterns in the travel sequences are subsequently computed. These profiles represent current workers' travel behavior and can thus be used to directly evaluate the simulated travel sequences yielded from activity-based models, by comparing them against the profiles obtained from the simulated sequences.

In relation to the existing OD-based method, the new approach offers the following major advantages. (1) It directly evaluates the predicted travel sequences, leading to more objective assessment and easier identification of the problems of the model system. (2) It examines the distribution of the sequences over the typical patterns, while the OD-based method looks into the number of trips across different OD pairs. In the new approach, the locations that are visited by an individual on a day are analyzed as a whole, while in the OD-based method, these locations are treated as unrelated individual activity participation. These two methods

have different perspectives and provide a complementary means of validating travel demand models.

The remainder of this chapter is organized as follows. Section 2 introduces the mobile phone data and Sections 3–5 detail the validation method. A case study is conducted in Section 6, and the obtained results are compared against real travel surveys in Section 7. Additional analysis on parameter sensitivity is performed in Section 8. Finally, Section 9 has discussions for future research and Section 10 draws major conclusions.

2. Mobile phone data

The mobile phone data consists of complete mobile communication patterns of 5 million anonymized users in Ivory Coast (i.e., 25% of this country's population) over 5 months between December 1, 2011 and April 28, 2012 [7]. The data contains the location (represented by cell ID) and time when each user conducts a call activity, including initiating or receiving a voice call or message. To address privacy concerns, the original data was divided into consecutive two-week periods; in each period, 50,000 users were randomly selected and their call records were extracted. This leads to a total of 10 datasets being generated. One of the datasets is used for this research. **Table 1** illustrates the typical call records of a user on Tuesday, December 20, 2011.

Call Time	**12:50:00**	**14:30:00**	**17:30:00**	**18:20:00**	**22:10:00**
Call location (Cell ID)	998	1520	982	956	956

Table 1. Call records of a user on a day.

3. Call location trajectory construction

A *call location trajectory* (i.e., *call-seq*) from a mobile phone user on a day can be described as a sequence of $l_1 \rightarrow l_2 \rightarrow \ldots \rightarrow l_n$, where l_i ($i = 1,\ldots,n$) is the cell ID and n is the *length* of the sequence, that is, the total number of locations that the user has reached and called that day. The *call frequency* (i.e., *call-fre*) of a set of consecutive calls made at each location l_i is denoted as k_i ($k_i > 0$), and the time for each of these calls is $T(1),\ldots, T(k_i)$. The *call interval* (i.e., *call-int*) between the first and last call time of these calls is $T(k_i) - T(1)$. Integraing the time signatures of the multiple calls, a *call-seq* can be formulated as $l_1(T(1),\ldots,T(k_1)) \rightarrow \ldots \rightarrow l_n(T(1),\ldots,T(k_n))$. From all the *call-seqs* of a user, home and work locations are first identified, and stop locations where the user has stayed for doing activities are then predicted.

3.1. Home and work locations

Two temporal points including *work start time* (i.e., *work-st*) and *work end time* (i.e., *work-et*) are estimated from the mobile phone data. The time when calls begin to substantially increase in

http://dx.doi.org/10.5772/intechopen.75810

the morning on weekdays is considered as the *work-st*. Likewise, the moment when call activities reach the second climax in the late afternoon is selected as the *work-et*. Around this time, we assume that people start to communicate for after-work activities. Based on these two time points, a location is regarded as a home if it is the most frequent call location during the night period on weekdays between the *work-et* and *work-st* as well as throughout the entire weekend period. In contrast, a location is chosen as a work place if it is the most frequent call place in the work period between the *work-st* and *work-et* on weekdays, and if it is different from the previously identified home location for the user. In addition, the calls at a work location should occur at least 2 days a week.

3.2. Stop locations

After the home and work locations are identified for each worker, the remaining places in the *call-seqs* are either *stop locations* (i.e., *stop-locs*) where people pursue activities other than home and work activities or *non-stop locations* (i.e., *nonstop-locs*). The *nonstop-locs* can be further divided into *trip locations* (i.e., *trip-locs*) where the user travels or *false locations* (i.e., *falso-locs*) that are wrongly recorded because of location update errors. When call traffic is high in a user's location area, this location is shifted to less crowded cells for a short period of time, leading to location updates, but the user does not actually move. Furthermore, even for the previously identified home or work locations, some instances of these two locations could also be due to non-stop reasons, for example, people traveling in their work area while calling. Therefore, each location instance in the *call-seqs* should be differentiated between a *stop-loc* and *nonstop-loc*, irrespective of its activity types.

The two scenarios where *nonstop-locs* could occur can be demonstrated with the call data of two users. The first user (User298) has a trajectory of l_1(17:06,17:43) → l_2(17:51) → l_3 (17:56,19:41) → l_4 (21:55), where four locations are observed and the *call-int* is 37, 0, 105, and 0 (min) respectively. Each of these locations needs to distinguished between a *stop-loc* and a *trip-loc*. The trajectory of the second user (User64) is l_1(13:21,20:11) → l_2(22:00) → l_3 (22:02) → l_4 (22:05) → l_2 (22,07,23:12). This user has five location updates, with the *call-int* as 410, 0, 0, 0 and 65 (min), respectively. It is noted that the time difference between the first and second visits to l_2 is only 7 min. Although a small chance exists that this user may have moved to l_3 and l_4, the occurrences of these two places in such short time is most likely caused by location update errors.

In order to recognize *stop-locs* from all the possible *nonstop-locs* in a *call-seq*, a method is proposed as follows. For each l_i in the *call-seq*, the *call-int* is examined. If it is longer than a threshold $T_{call\text{-}int}$, l_i is regarded as a *stop-loc*. Otherwise, if the *call-int* is shorter than (or equal to) $T_{call\text{-}int}$ (e.g., in the case of a single call made at l_i), and if this location appears in the middle of the call trajectory, the time interval between the last call time at the previous location of l_i (i.e., $T(k_{i\text{-}1})$ at $l_{i\text{-}1}$) and the first call time at its next location (i.e., $T(1)$ at l_{i+1}), defined as the *maximum time boundary* (i.e., *max-boundary*), is examined. If this interval is longer than a threshold $T_{max\text{-}boundary}$, l_i is considered as a stop. However, if l_i is the first or last location of the trajectory, all the distinct stop locations already identified according to the previous steps from the user are aggregated. If l_i is among these locations, it is predicted as a stop. Otherwise, l_i is treated as either a *trip-loc* or *false-loc* and thus deleted. After the elimination of all the *nonstop-locs*, the

remaining places from the *call-seq* are stored into a *stop trajectory* (i.e., *stop-seq*). Each l_i in these trajectories is annotated with its *activity type* (i.e., *act*(l_i)), categorized into home (*H*), work (*W*), and other (*O*) activities. Travel is implicit in between each two consecutive locations. For instance, based on the above process, if 30 and 60 min are used for $T_{call\text{-}int}$ and $T_{max\text{-}boundar}$ as adopted in our case study, the obtained *stop-seqs* for User298 and User64 are $l_1 \rightarrow l_3 \rightarrow l_4$ and $l_1 \rightarrow l_2$, with the *activity types* of these locations as l_1 (*W*) $\rightarrow$ l_3 (*O*) $\rightarrow$ l_4 (*H*) and l_1 (*W*) $\rightarrow$ l_2 (*H*), respectively.

4. Trajectory transformation

Mobile phone data is event driven, in which locations are recorded only when the devices connect to the GSM network. Users' call behavior affects the number of trips and activity locations that are captured by the call data. The more active a user is in using the phone, the better his/her travel behavior is revealed by the device. The call locations can be regarded as the observed behavior at certain temporal points on a day, based on which real travel behavior of the users can be deducted. A method is therefore developed to transform the *stop-seqs* into *actual travel sequences* (i.e., *actual-seqs*) that represent real travel paths of the users on those days. This method is composed of the following steps. (1) For each user, two variables including the *actual activity duration* at a location l_i (i.e., *actual-dur*(*user*, l_i)), and the *call rate per minute* at all call locations of the user (i.e. *CallRate*(*user*)), are derived. (2) These two obtained variables are converted into a *call probability* that the user makes at least one call at l_i (i.e., *CallP*(*user*, l_i)). (3) Given a real travel sequence on a day, various *stop-seqs* could be possibly observed depending on the user's call behavior. The *conversion probability,* at which a certain *stop-seq* is generated from the *actual-seq* (i.e., *ConvertP*(*user, actual-seq, stop-seq*)), is calculated. (4) Based on the observed frequencies of all the *stop-seqs* from the user, a linear equation is built and the frequencies of the actual travel sequences are inferred.

4.1. *CallRate*(*user*) and *actual-dur*(*user*, l_i)

The *CallRate*(*user*) describes the probability that a user makes calls each minute, and it is calculated as follows:

$$CallRate(user) = \frac{\sum_{day} total - number - calls(user, day)}{\sum_{day} time - span(day)} \quad (1)$$

where, *total-number-calls* (*user, day*) and *time-span*(*day*) denote the total number of calls each day for the user and the time interval (min) of these days. *Actual-dur*(*user*, l_i) is the actual duration (min) that a user spends at l_i. Since no information on the actual stop duration is provided from the phone data, we approximate this variable using the average of the duration of all stop locations with the same activity types over all respondents that are obtained from a travel survey (see Section 7.2). The derived *average duration of locations per activity type* is referred as *actual-dur*(*act*(l_i)).

http://dx.doi.org/10.5772/intechopen.75810

4.2. *CallP*(*user, l_i*)

Given a user's call rate and the *actual-dur*(*user, l_i*) that the individual has spent at l_i, the probability that the user makes at least one call during the visit to l_i (i.e., *CallP*(*user, l_i*)) is cumputed based on the following steps. (1) The *actual-dur*(*user, l_i*) is first divided into a number of equal-length intervals. Each of these intervals is regarded as an experiment, and its length (i.e., *EpisodeL*) can be decided by the average duration that people spend on the phone each time they connect the GSM network (e.g., 2 min in our case study). (2) Under the assumption that users make calls independently in each interval and that the likelihoods of calling across different intervals are identical, *CallP*(*user, l_i*) then follows the binomial distribution. The *actual-dur*(*user, l_i*) decides the total number of intervals (i.e., independent experiments), and the call rate provides the probability of calling in each interval (i.e., the success for each experiment result). (3) *CallP*(*user, l_i*) can be computed according to Formula (2), as the probability of making at least one call (i.e., having at least one success) over the entire *actual-dur*(*user, l_i*) (i.e., the total number of experiments). *CallRate*(*user*) and *actual-dur*(*act*(l_i)) are used as the approximation of the call rate and the *actual-dur*(*user, l_i*) for a particular activity type of l_i.

$$CallP(user, l_i) = 1 - \{1 - EpisodeL \times CallRate(user)\}^{actual-dur(act(l_i))/EpisodeL} \tag{2}$$

4.3. *ConvertP*(*user, actual-seq, stop-seq*)

Let $l_1 \rightarrow l_2 \rightarrow \ldots \rightarrow l_n$ represent the *actual-seq* on a day for a user. Based on *CallP*(*user, l_i*), the probability that a certain *stop-seq* could be observed from the original sequence, that is *Convert* (*user, actual-seq, stop-seq*), is calculated. For instance, the probability of generating the *stop-seq* $l_1 \rightarrow \ldots l_{i-1} \rightarrow l_{j+1} \ldots \rightarrow l_n$ (i ≤ j) is

$$\begin{aligned} ConvertP\ & \left(user, l_1 -> l_2 \ldots -> l_n, l_1 -> \ldots l_{i-1} -> l_{j+1} -> \ldots l_n\right) \\ & = \prod_{m=1}^{i-1} CallP(user, l_m) \times \prod_{m=i}^{j} \overline{CallP(user, l_m)} \times \prod_{m=j+1}^{n} CallP(user, l_m), \\ & \overline{CallP(user, l_m)} = 1 - CallP(user, l_m) \end{aligned} \tag{3}$$

Where, we assume that no calls were made during the visits to the locations from l_i to l_j. We also hypothesize that users make calls independently across different location visits.

The conversion process can be demonstrated by the call records of User302. The probabilities that this user makes at least one call at home, work, and other locations are 0.81, 0.90, and 0.42, respectively. Assuming that this individual has an *actual-seq* of *HWOH* on a certain day, a total of 15 different *stop-seqs* could be possibly generated from this original sequence. The sum of the conversion possibilities of these *stop-seqs* is 1. For instance, the possibility of generating *HWH* is *ConvertP*(*user, HWOH, HWH*) = *0.34*.

4.4. *Actual-seq* derivation

Let $y_1, y_2, \ldots, y_k$ represent the frequencies of all k different *stop-seqs* of $s_1, s_2, \ldots, s_k$ constructed from a user's call records. These *stop-seqs* are sorted by their *length* in a descending order, with

s_1 having the largest number of locations. Assume that the corresponding *actual-seqs* of the user also occur among $s_1, s_2, \ldots, s_k$; the frequencies of the *actual-seqs* (i.e., $x_1, x_2, \ldots, x_k$) can be estimated based on Formula (4). Note that the parameter user in *ConvertP* is left out.

$$\begin{aligned} & x_1 \times ConvertP(s_1, s_1) = y_1 \\ & x_1 \times ConvertP(s_1, s_2) + x_2 \times ConvertP(s_2, s_2) = y_2 \\ & \ldots \\ & x_1 \times ConvertP(s_1, s_k) + x_2 \times ConvertP(s_2, s_k) + x_2 \times ConvertP(s_k, s_k) = y_k \end{aligned} \tag{4}$$

An additional constraint $\sum_{i=1}^{k} x_i = \sum_{i=1}^{k} y_i$ is added to the above formula, in order to ensure that the total number of the derived sequences and that of the observed trajectories are equal. This leads to a model with $k + 1$ equations and k unknown variables $x_1, x_2, \ldots, x_k$. To find the optimal solution to the unknown variables, the Linear Least Square Method is employed. This method searches for the answer by minimizing the sum of the squares of *residuals* that are the differences between the observed frequencies and the corresponding estimated frequencies by the model. Specifically, let the estimators of $x_1, x_2, \ldots, x_k$ as $\widehat{x}_1, \widehat{x}_2, \ldots, \widehat{x}_k$; the *residual* for the ith equation (i.e., $residual_i$, $i = 1,\ldots,k$) is calculated as follows.

$$\begin{aligned} & residual_1 = \widehat{x}_1 \times \mathrm{ConvertP}(s_1, s_1) - y_1 \\ & \ldots \\ & residual_k = \widehat{x}_1 \times \mathrm{ConvertP}(s_1, s_k) + \widehat{x}_2 \times \mathrm{ConvertP}(s_2, s_k) + \ldots + \widehat{x}_k \times \mathrm{ConvertP}(s_k, s_k) - y_k \end{aligned} \tag{5}$$

With $\widehat{x}_k$being replaced with $\widehat{x}_k = \sum_{i=1}^{k} y_i - \sum_{i=1}^{k-1} \widehat{x}_i$, the last equation is converted as

$$residual_k = \widehat{x}_1 \times \mathrm{ConvertP}(s_1, s_k) + \ldots + \left(\sum_{i=1}^{k} y_i - \sum_{i=1}^{k-1} \widehat{x}_i \right) \times \mathrm{ConvertP}(s_k, s_k) - y_k \tag{6}$$

The total sum of the squared residuals (i.e., *residual-sum*) is computed, and the minimum of the *residual-sum* is found by setting its partial derivatives to zero as follows.

$$residual - sum = \sum_{i=1}^{k} (residual_i)^2, \quad \frac{\partial (residual - sum)}{\partial \widehat{x}_i} = 0, i = 1, \ldots k - 1 \tag{7}$$

This results in the computation of $\widehat{x}_1, \widehat{x}_2, \ldots \widehat{x}_{k-1}$; $\widehat{x}_k$ is obtained as $\widehat{x}_k = \sum_{i=1}^{k} y_i - \sum_{i=1}^{k-1} \widehat{x}_i$.

In the case of User302, the *stop-seqs* derived from his/her call records of 10 days are *HWOH, WH, OH, W,* and *H*, with the frequencies as 1, 3, 2, 1, and 3, respectively. The original frequencies of these sequences are estimated as $\widehat{x}_1 = 1.17, \widehat{x}_2 = 3.74, \widehat{x}_3 = 4.89, \widehat{x}_4 = 0.07, \widehat{x}_5 = 0.14$, and the *residual-sum* is 0.74.

During the above described process, we assume that the actual travel sequences *actual-seqs* could only occur within the set of the observed location trajectories, that is, *Set* = $\{s_1, s_2, \ldots, s_k\}$. This is

based on the well-established findings that human activity and travel behavior exhibit a high degree of spatial and temporal regularities as well as sequential ordering. A limited variety of travel sequences for a user can be observed during a certain time period. In addition, the optimal solution of the frequencies of the *actual-seqs* would be most likely found within the *Set*. This is due to the fact that if an *actual-seq* s_p is not in the *Set*, the optimal estimator $\widehat{x}_p$ for the frequency of s_p would be a value less than or equal to zero. For instance, for User302, if s_p is longer than any trajectory in the *Set*, e.g. s_p = *HWOWH*, the equation $x_p \times ConvertP(HWOWH, HWOWH) = 0$ is obtained. From this equation, we obtain $\widehat{x}_p \approx 0$. Similarly, if s_p is shorter than certain trajectories in the *Set*, for example, $s_p = HWO$, the equation $x_1 \times ConvertP(HWOH, HWO) + x_p \times ConvertP(HWO, HWO) = 0$ is constructed, leading to $\widehat{x}_p < 0$.

5. Workers' travel sequence classification

Figure 2 describes travel sequences for workers, in which a sequence is divided into four parts, including before-work, commute, work-based, and after-work parts. They respectively represent the activities and travel undertaken before leaving home to work (indicated by the arrow a, e.g., *HOH*), between home and work commutes (by b and d, e.g., *HOW* or *WOH*), work-based (by c, e.g., *WOW*), and after arriving home from work (by e, e.g., *HOH*).

A *home based tour* (i.e., *tour*) is defined as a chain of locations that starts and ends at home and accommodates at most two work location visits. For a working day, a *tour* can be classified into the patterns of *HWH, HOWH, HWOH, HWOWH, HOWOH, HOWOWH, HWOWOH,* and *HOWOWOH*, where *O* represents one or a chain of visits to several other different locations. On a non-working day, a *tour* is described with *H* or *HOH*. All the above 10 patterns characterize the *tours* for worker's travel behavior, and they are defined as *home-based tour classification* (i.e., *tour-class*). Each pair of these patterns is then merged, leading to 81 combinations that can be used to classify an entire day's sequences containing two *tours*. For instance, the combination of *HWH* and *HOWH* results in the class *HWHOWH*. The daily sequences that have more than two work location visits in a *tour* (e.g., *HWOWOWH*) or that contain more than two *tours* (e.g., *HWHWHWH*) are each formed into one additional category. Thus, all the combinations along with the original 10 *tour* patterns that describe daily sequences with only one *tour* lead to a total of 93 patterns. These patterns underlie workers' daily travel behavior,

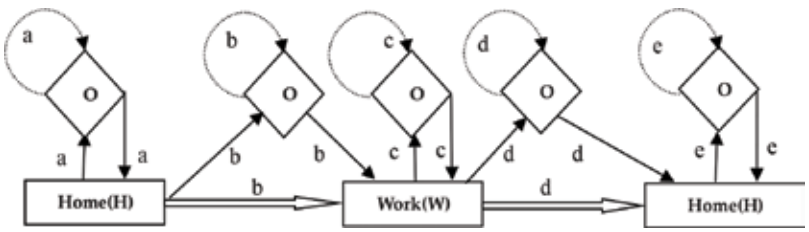

Figure 2. Workers' travel sequence representation. Note: H and W denote home and work locations respectively, while O refers to all other places. Each arrow from the start and end locations represents the related travel between these two places, and the arrow from the location O to itself indicates a chain of consecutive visits to different O locations.

and they are denominated as *day sequence classification* (i.e., *day-class*). Given a group of users, their travel sequences can be categorized according to the *tour-class* and *day-class*, respectively. The relative frequency of each of the patterns over the total occurrences of all the corresponding patterns among the travel sequences forms a *home-based-tour-profile* (i.e., *tour-profile*) and *day-sequence-profile* (i.e., *day-profile*) among these individuals.

Based on these two types of classification, all the *stop-seqs* and *actual-seqs* previously constructed from the mobile phone data are grouped. During this process, a home location H is added at the beginning and end of a sequence if it is absent from the sequence, under the assumption that each user starts and ends a day at home. After classification, two types of profiles, that is, the *tour-profiles* and *day-profiles*, are derived from both the *stop-seqs* and *actual-seqs*, respectively.

The Pearson correlation coefficient r (see Formula (8)) is used to measure the relation between the corresponding profiles derived from the different sets of sequences. It reveals the strength of relationship between the compared sets; the closer r is to 1, the stronger the relationship is.

$$r = \frac{\sum_{i=1}^{d}\left(\frac{A_i - \overline{A}}{S_A}\right)\left(\frac{B_i - \overline{B}}{S_B}\right)}{d-1}, \overline{A} = \frac{\sum_{i=1}^{d} A_i}{d}, \overline{B} = \frac{\sum_{i=1}^{d} B_i}{d}$$
$$S_A = \sqrt{\frac{\sum_{i=1}^{d}\left(A_i - \overline{A}\right)^2}{d}}, S_B = \sqrt{\frac{\sum_{i=1}^{d}\left(B_i - \overline{B}\right)^2}{d}} \tag{8}$$

where A and B denote the two compared profiles, A_i and B_i are the frequencies of the pattern i in these two profiles and d denotes the total number of the patterns in each profile.

6. Case study

In this section, adopting the proposed approach and using the mobile phone data described in Section 2, we carry out a case study. In this process, *stop-seqs* are first constructed and *actual-seqs* are then derived.

6.1. *Stop-seq* construction

Figure 3 describes the distribution of the number of calls in each hour of the weekdays, showing that the peaks of calls in the morning and in the afternoon occur at 9 am and 18 pm, respectively. These two temporal points are chosen as the *work-st* and *work-et*. Based on the criteria for home and work locations, 49,421 (i.e., 98.8% of the total) users have their home identified, and 8016 users (i.e., 16.2% of the total) are screened out as employed people who work between 9 am and 18 pm at least two weekdays per week. All the call records of these workers on weekdays are extracted, resulting in a total of

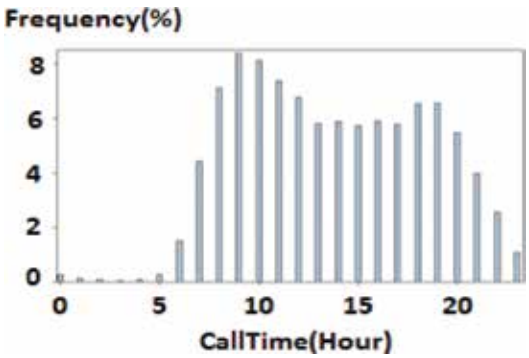

Figure 3. The distribution of call time.

69,536 *call-seqs*. From these sequences, 40.3% of the locations are removed as non-stop ones, using the thresholds $T_{call\text{-}int}$ and $T_{max\text{-}boundary}$ set as 30 and 60 min respectively. The remaining locations form the *stop-seqs*, with the average length of these sequences as 3.3 (locations).

6.2. *Stop-seq* transformation

The *time-span* (*day*) is specified as the period from 6 am to 12 pm. From each user, all the calls made during this period each day across the two survey weeks are counted, and the *CallRate*(*user*) is computed according to Formula (1). The average *CallRate*(*user*) over all the workers is 0.0073. The *actual-dur*(*act*(l_i)) is estimated from the travel survey conducted in Belgium which will be described in Section 7.2. From this survey, the duration is 222, 317 and 75 (min) for home, work, and other activities, respectively. The variable *EpisodeL* specifies the time window by which the *actual-dur*(*act*(l_i)) is split into a number of intervals, that is, experiments. To obtain this interval length, the total number of voice calls and the total duration of these calls over the 5-month mobile phone data in Ivory Coast are extracted. The ratio between these two variables leads to the average call duration as 1.92 min, and 2 min is thus taken as the estimation of *EpisodeL*. Based on all the above parameter settings, the call probabilities *CallP*(*user,* l_i) at home, work, and other locations for each user are respectively derived based on Formula (2). The average of *CallP*(*user,* l_i) over all the users for these three types of locations is 0.81, 0.88, and 0.41, respectively. The obtained call probabilities of each user, combined with the observed frequencies of the *stop-seqs* for the person lead to the calculation of the number of the *actual-seqs*, using the method described in Sections 4.3 and 4.4.

7. Result comparison with travel surveys

To examine the validation ability of the proposed method, the results inferred from the phone data are compared with the statistics drawn from real travel surveys. However, no

official surveys have been conducted in Ivory Coast, necessitating the use of data collected from other countries including South Africa and Belgium for this purpose. We acknowledge that the travel behavior in Ivory Coast and that in these two travel-surveyed countries are most likely different. Consequently, the comparison is to examine the validation potential of this approach but not to infer travel behavior relationship among these countries.

7.1. The two travel surveys

The South Africa National Household Travel Survey (*NHTS*) was based on a sample of 50,000 households over a period of 2 months between May and June in 2003 [8]. The collected information includes the number of trips on a typical weekday and the travel time and purposes of these trips. According to the survey results, the majority of the respondents can access most of the activity services in this country (e.g., train and bus stops as well as shops and post offices) within half an hour, and the average number of activity locations visited by a worker on a day is between 3.46 and 4.06.

The Belgian survey (*SBO*) was carried out on 2500 households between 2006 and 2010. This survey collects trip information of the respondents during the course of 1 week, such as trip origin and destination (i.e., activity locations), trip start and end time, and purposes of the trips (activity types). Activity locations are described with statistical sectors; the size of these sectors varies from a few hundred meters to a few thousands in radius, comparable to the spatial granularity of cells in a GSM network. Based on this survey, the average travel time is 24 min, 6 min shorter than a typical travel in South Africa. **Table 2** illustrates a representative diary of the respondent 'HH4150GL10190' on Tuesday, May 9th, 2006.

From all the respondents in the *SBO* survey, 342 individuals who work at least 2 days a week are selected, and the corresponding travel sequences are constructed. The duration of each location in the travel sequences is estimated as follows. If the location is not the first and the last one of the day, the duration is between the arrival time of the current trip at the location and the leaving time of the next trip from the location. Otherwise, the time of 6 am is used as the start time of the location if it is the first one of the day, and the time of 12 pm is adopted as the end time of the location if it is the last one on the day. For instance, the respondent demonstrated in **Table 2** has a sequence of *HWOH*, with the location duration as 165, 540, 25, and 255 (min), respectively. All the obtained location duration is averaged per activity type over all these individuals, leading to the estimate of the *actual-dur*($act(l_i)$), which has been used to derive the *actual-seqs* in the case study in Section 6.

Trip ID	Start Time	End Time	Origin	Destination	Purpose
1	08:45:00	09:00:00	34,137	34,145	Work
2	18:00:00	18:15:00	34,145	34,849	Shopping
3	18:40:00	19:05:00	34,849	34,637	Home

Table 2. Diary data.

7.2. Average length of sequences

Table 3 summarizes the average length of the sequences including *call-seqs*, *stop-seqs*, and *actual-seqs* derived from the mobile phone data as well as of the sequences constructed from the *NHTS* and *SBO* diaries. It shows that the average length of the sequences first decreases from 5.69 for the *call-seqs* to 3.3 for the *stop-seqs* and then rises back to 4.02 for the *actual-seqs* that is the closest to the length of both the *NHTS* and *SBO* diaries. The length differences suggest the importance of the process from the identification of stop locations to the inference of complete travel sequences, when travel behavior is analyzed based on mobile phone data.

7.3. *Tour-profiles*

Based on the classification method described in Section 5, two types of profiles, including the *tour-profiles* and *day-profiles*, are derived from the *stop-seqs*, *actual-seqs* and *SBO* diaries, respectively. **Table 4** shows the frequency of each pattern in the *tour-profiles*; the differences in the frequencies of the corresponding patterns between the *stop-seqs* and *actual-seqs* as well as between the *actual-seqs* and *SBO* diaries are also presented.

Due to the data collection nature of mobile phone data, when the *stop-seqs* are converted into the *actual-seqs*, two important characteristics are expected. (1) A *stop-seq* is generated not only from an *actual-seq* that is identical to this trajectory (e.g., when calls were made at each of the locations actually visited), but more likely from a sequence that is longer than this observed one (i.e., some of the real locations being missed if no calls were made there). Thus, when the *stop-seqs* are transformed into the *actual-seqs*, the number of long patterns increase, while that of short patterns decrease. (2) If the probability of making calls at a location is lower, the frequency for the derived *actual-seqs* that contain this location tends to be higher. These two features are well observed in **Table 4**. For instance, when the *actual-seqs* are compared with the *stop-seqs* (see the 4th row), the frequencies for the short patterns *H* and *HWH* decrease by 4.6 and 11.2%, while the frequency for the long one *HWOWH* increases by 0.7%. Furthermore, as the average call probability at the location *O* is the lowest among all the three activity types, an 8.3% rise is obtained for the pattern *HOH* among the *actual-seqs*. This forms a contrast with the pattern *HWH* that has an 11.2% decrease.

When the profiles drawn from the *actual-seqs* and *SBO* diaries are compared, a correlation coefficient of 0.99 is obtained (i.e., higher than the coefficient of 0.93 between the *stop-seqs* and *SBO* diaries). The high coefficient shows an overall high level of similarities across the patterns in the *tour-class* between these two types of sequences. Nevertheless, as previously indicated, due to the contextual deviations between Belgium and Ivory Coast, the real travel behavior between two countries can be very different. According to **Table 4** (see the 5th row), the differences in the frequencies over all the patterns between the *actual-seqs* and *SBO* diaries range from −6.2 to

Call-seqs	Stop-seqs	Actual-seqs	NHTS	SBO
5.69	3.30	4.02	3.46–4.06	3.96

Table 3. Average length of sequences.

	H	HWH	HOH	HOWH	HWOH	HWOWH	HOWOH	HOWOWH	HWOWOH	HOWOWOH	More than 2 W
SS	9.0	50.3	18.0	5.1	8.2	3.4	2.5	0.7	1.4	0.5	1.0
AS	4.4	39.1	26.3	6.7	10.3	3.8	4.1	1.0	2.1	0.8	1.3
SBO	6.4	42.9	32.5	3.1	10.8	1.6	1.9	0.2	0.5	0.1	0.2
AS - SS	−4.6	−11.2	8.3	1.6	2.1	0.4	1.6	0.3	0.7	0.3	0.3
AS – SBO	−2.0	−3.8	−6.2	3.6	−0.5	2.2	2.2	0.8	1.6	0.7	1.1

Note: The column represents the patterns, while the row denotes each single set of sequences (in the first three rows) and the differences in frequencies between pair sets of sequences (in the last two rows). SS, AS, and SBO refer to the stop-seqs, actual-seqs, and SBO diaries respectively.

Table 4. Tour-profiles (%).

3.6%. In addition, the increases in frequencies for short patterns *H*, *HWH* and *HOH* from the *SBO* survey by 2, 3.8 and 6.2%, respectively, could also be caused by the under-reporting of short trips or short-duration activities that typically occur in travel surveys. This leads to travel sequences obtained from the surveys being shorter than they actually are [1].

7.4. *Day-profiles*

Figure 4 describes the correlation between the frequencies of corresponding patterns in the *day-profiles*, where the x-axis represents the frequencies for the *stop-seqs* (**Figure 4a**) and *SBO* diaries (**Figure 4b**), respectively, while the y-axis denotes the frequencies for the *actual-seqs*. The line of $y = x$ is presented as a reference. From **Figure 4(a)**, it is noted that the majority patterns follow similar frequency distributions, with a coefficient as 0.91. However, there exist a few outliers that can be further divided into two groups. (1) The group of *HWH, H* and *HWHWH* with 14.3, 5.7, and 1.5% increases in frequencies for the *stop-seqs*, respectively. (2) The other group of *HOH, HOWOH*, and the patterns composed of more than two *tours*, showing 3.5, 2.4, and 2.2% higher for the *actual-seqs*. This further demonstrates that, in relation to the *stop-seqs*, the *actual-seqs* are likely to have a high percentage for long patterns and for patterns that contain locations featured with low call probabilities (e.g., the location *O*). In contrast, a lower proportion is expected among the *actual-seqs* for short patterns and for patterns consisting of locations featured with a high call rate (e.g., the location *W*).

In **Figure 4(b)**, the *day-profiles* obtained from the *actual-seqs* and the *SBO* survey are compared. It shows that the majority of the patterns have higher frequencies for the *actual-seqs* than for the *SBO* data (i.e., the points above the line). Nevertheless, a few patterns show higher occurrences for the *SBO* diaries (i.e., the points below the line) and they mainly consist of short patterns, for example, *HWH, HOH* and *HWOH* with 7.3, 7.1, and 3.2% increases, respectively. Apart from the deviations in travel behavior between these two countries, this figure demonstrates again the possibly missing records for short-duration trips or activities in travel surveys, resulting in a high frequency for short patterns. On top of that, further investigation reveals that out of all 93 patterns in the *day-profiles*, 57 (i.e., 61.3%) have zero frequencies for the *SBO* data; while for the *stop-seqs* and *actual-seqs*, only 18 patterns (i.e., 19.4%) are not present. It indicates that the sequences built from the mobile phone data are more diverse and representative in travel

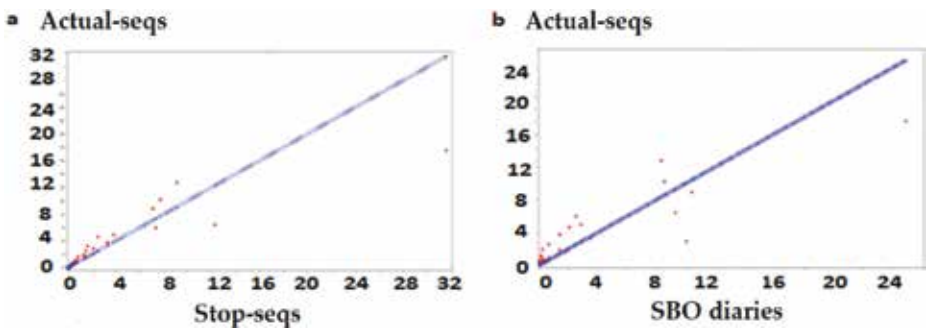

Figure 4. Correlation between the frequencies of corresponding sequences.Correlation between the frequencies of the actual-seqs and those of the stop-seqs (a) and SBO diaries (b).

behavior than the ones from the survey, further underlying the potential value of using mobile phone data for travel behavior analysis.

Despite the differences in each particular pattern, the coefficient is 0.89 between the whole *day-profiles* obtained from the *actual-seqs* and *SBO* survey. It shows that the profile inferred from the mobile phone data is comparable to the one extracted from a real travel survey. This further suggests that the derived profile can sufficiently represent workers' travel behavior in a study area, and therefore capable of validating the sequences generated from activity-based models.

8. Sensitivity analysis

In the proposed approach, several parameters including $T_{call\text{-}int}$, $T_{max\text{-}boundary}$ and *actual-dur*(*act*(l_i)) have been defined. Final investigation into how these parameters affect the derived results is conducted in two aspects. (1) The average length of the *stop-seqs* and *actual-seqs* (i.e., *SS-length* and *AS-length*). (2) The coefficients between the *stop-seqs* and *actual-seqs* as well as between the *actual-seqs* and *SBO* diaries (i.e., r_1 and r_2 respectively).

8.1. $T_{call\text{-}int}$ and $T_{max\text{-}boundary}$

$T_{call\text{-}int}$ defines the minimum time duration above which a call location is considered as a stop. The larger this value, the longer the duration of a stop is required, and the shorter the average length of the obtained sequence tends to be. This is well reflected in **Figure 5(a)**. However, the length of the sequences decreases slowly and enters into a constant level when this parameter passes the 30-min threshold (i.e., the value adopted in our case study). Similarly, r_1 and r_2 reach a stable level at the same 30-min threshold.

The parameter $T_{max\text{-}boundary}$ specifies a minimum value, such that given the current location under investigation and the trip to this location, if the time interval between the start of its next trip and the end of its previous trip is longer than this threshold, the current location is predicted as a stop. From **Figure 6a**, it is observed that as $T_{max\text{-}boundary}$ increases, both *SS-length*

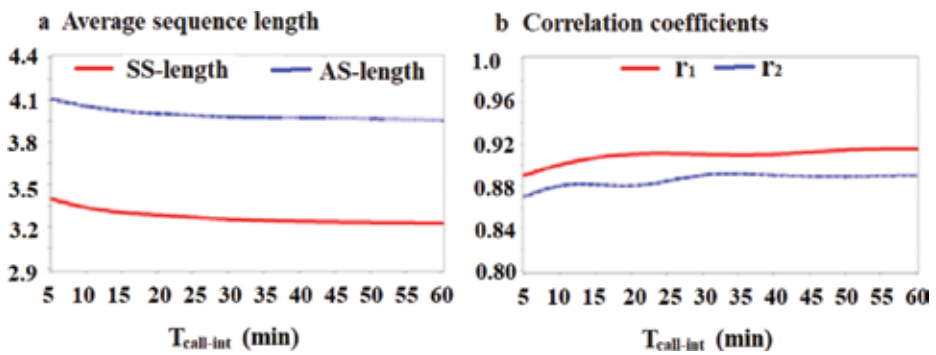

Figure 5. The relation between $T_{call\text{-}int}$ and the average sequence length (a) and the coefficients (b).

and *AS-length* continuously decrease without stopping at a certain value. In contrast, r_1 and r_2 fall into a steady state when $T_{max\text{-}boundary}$ passes to a certain value (e.g., 60 min) (see **Figure 6b**). This can be due to the possibility that the dismissed stop locations caused by the increase in $T_{max\text{-}boundary}$ are likely distributed randomly across various types of patterns, leading to the relative frequencies of these patterns remaining unchanged.

8.2. *Actual-dur*(*act*(l_i))

Figure 7 depicts the relation between the threshold *actual-dur*(*act*(l_i)) for work activities and the derived results. It shows that, when this parameter passes a certain point, e.g., 317 min specified in this study, the changes in *AS-length* as well as r_1 and r_2 disappear. This phenomenon can be explained by the binomial model used to estimate the call probability *CallP*(*user*, l_i). According to this model, when the *actual-dur*(*act*(l_i)) is longer, *CallP*(*user*, l_i) becomes larger. But the call probability eventually enters into a constant value of 1 at a certain point of the *actual-dur*(*act*(l_i)) (see **Figure 8**).

The above sensitivity analysis shows that, except $T_{max\text{-}boundary}$ that exhibits a certain level of influences on the average length of the sequences, a certain amount of changes in these

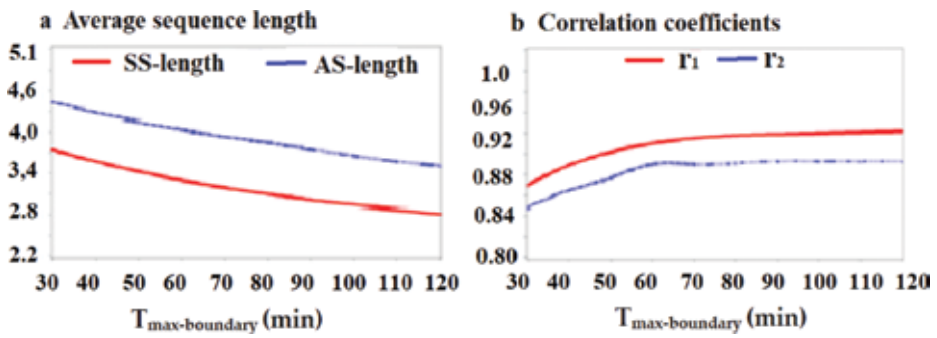

Figure 6. The relation between $T_{max\text{-}boundary}$ and the average sequence length (a) and coefficients (b).

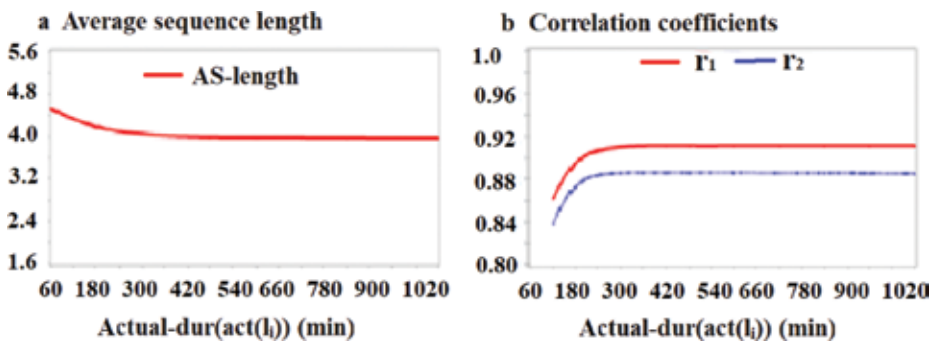

Figure 7. The relation between the *actual-dur*(*act*(l_i)) for work activities and the AS-length (a) and the coefficients (b).

parameters does not cause a substantial discrepancy in the sequence length and the profiles. This indicates that the profiles constructed from the mobile phone data are stable and consistent in characterizing workers' travel behavior; a minor change in these parameters will not result in significantly different outcomes.

9. Discussions

A number of areas can be enhanced in the future research. (1) In the prediction of home and work locations, a fixed work period (i.e., the time interval between 9 am and 18 pm on weekdays) is assumed. Under this assumption, people who work in night shifts or at weekends are ignored. The prediction of these two places could be improved by first deriving possible work regimes of the users. Flexible work periods can then be adopted corresponding to the different regimes. (2) During the process of stop location identification, rather than using general thresholds of 30 and 60 min for $T_{call\text{-}int}$ and $T_{max\text{-}boundary}$, these two parameters should be tailored to particular cells and individuals' travel speeds. For instance, smaller values than the current settings should be used for cells in a smaller size and for individuals with a higher travel speed (e.g., by car or by train). (3) With respect to the process of converting *stop-seqs* into *actual-seqs*, the estimation of the *actual-dur*($act(l_i)$) should be separated among different social-economic groups, as the work duration for full-time workers is longer than that for part-time ones. Moreover, rather than using an identical *CallRate*(*user*) for all activities conducted by a user, the call rate could be differentiated across different activity types, as the likelihood of making calls may vary depending on the activity context. (4) When examining the validation potential of this method, travel surveys stemmed from different geographic areas than that of mobile phone data are used. However, as discussed in Section 7, deviations exist in terms of land use, transport networks and social-economic conditions of individuals across different regions and countries, and travel behavior is shaped by all of these factors. Thus, in the future, the proposed method must be validated using a real travel survey performed in the same or similar context to where the mobile phone data is recorded. Such surveys will provide more relevance to the current method by identifying the optimal parameters as well as assessing the results. (5) With the rapid

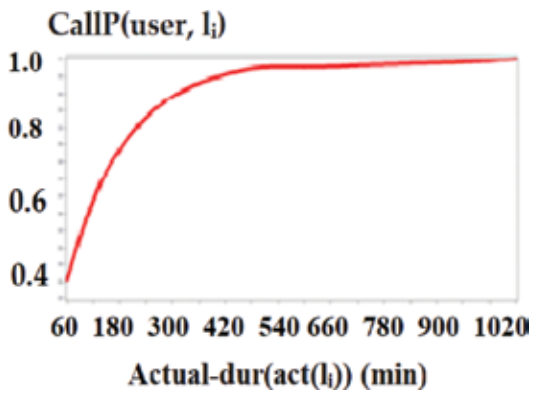

Figure 8. The relation between the *actual-dur*($act(l_i)$) for work and *CallP*(*user*, l_i).

development of mobile application, mobile phone data will be collected not only when people make calls but when they use the application. The location data will thus reveal more activities and travel episodes, enabling more accurate travel sequence derivation based on the proposed approach. This will lead to an even more reliable activity-based model validation method as well as to improved travel behavior analysis in general.

10. Conclusions

The proposed method can be integrated into activity-based models at the stage of 'simulated travel sequences' (see **Figure 1**). Specifically, during this validation, the set of simulated travel sequences yielded from the activity-based models is compared with the set of the *actual-seqs* derived from the mobile phone data in the following two aspects: (1) The average number of location visits per day, that is, the average length of the sequences. (2) The sequential order of the activities, reflected by the correlation between the corresponding profiles (i.e., the *tour-profiles* or *day-profiles*) constructed from each set of the sequences. If a large difference in the average length of the sequences or a low coefficient between these profiles is found, it would suggest mismatches between the simulated results and the travel patterns represented by the call data. This thus signals possible problems and calls immediate action into the examination of the activity-based models, prior to the utilization of the model results for further traffic assignment and mobility-related analysis.

Apart from the initial goal of developing a new validation method, this study also designs a novel process for stop location identification and *actual-seqs* derivation. This process integrates daily activities and travel with call activities; both types of activities occur concurrently and are performed by the same individuals. This method presents a solution to the challenge that is pertinent to mobile phone data research and application in a variety of fields, for example, in urban planning and location-based services. Due to the event-driven nature of mobile phone data collection, the locations are recorded only when a user connects the GSM network. What the user is doing (e.g., traveling or doing activities) is not known. Moreover, the places, where the person has stayed but without calling, are also dismissed. The location update errors which result in wrong documentation of user's actual locations raise another issue on the data collection. The results from our case study suggest a decrease by 42% in the number of location visits per day, when *stop-seqs* are constructed from the raw phone data. This number increases by 22% when the dismissed places are interpolated and the complete *actual-seqs* are formed. Such scales of changes signify the importance of the integration between the existing research using mobile phone data and this process in this study.

Acknowledgements

The authors would like to acknowledge the support of the European Union through the project of DataSim. We also thank the Orange Data for Development (D4D) challenge committee for the provision of mobile phone data.

Author details

Feng Liu[1,2]*, Ziyou Gao[1], Bin Jia[1], Xuedong Yan[1], Davy Janssens[2] and Geert Wets[2]

*Address all correspondence to: feng.liu@uhasselt.be

1 MOE Key Laboratory for Urban Transportation Complex Systems Theory and Technology, Beijing Jiaotong University, Beijing, China

2 Transportation Research Institute (IMOB), Hasselt University, Diepenbeek, Belgium

References

[1] Cui JX, Liu F, Hu J, Janssens D, Wets G, Cools M. Identifying mismatch between urban travel demand and transport network services using GPS data: A case study in the fast growing Chinese city of Harbin. Neurocomputing. 2016;**181**:4-18

[2] Cui JX, Liu F, Janssens D, An S, Wets G, Cools M. Detecting urban road network accessibility problems using taxi GPS data. Journal of Transport Geography. 2016;**51**:147-157

[3] Hartgen DT. Hubris or humility? Accuracy issues for the next 50 years of travel demand modeling. Transportation. 2013;**40**(6):1133-1157

[4] Shan J, Viña-Arias L, Ferreira J, Zegras C, González MC. Calling for validation, demonstrating the use of mobile phone data to validate integrated land use transportation models. In: Proceedings 7VCT. 2011. 2011. http://hdl.handle.net/2099/15544, ISBN: 978-972-96524-6-2, 978-989-97510-0-2

[5] Liu F, Janssens D, Cui JX, Wets G, Cools M. Characterizing activity sequences using profile hidden Markov models. Expert Systems with Applications. 2015;**42**(13):5705-5722

[6] Liu F, Cui JX, Janssens D, Wets G, Cools M. Semantic annotation of mobile phone data using machine learning algorithms. In: Smartphones from an Applied Research Perspective. Intech; 2017. DOI: 10.5772/intechopen.70255. https://www.intechopen.com/books/smartphones-from-an-applied-research-perspective/semantic-annotation-of-mobile-phone-data-using-machine-learning-algorithms

[7] Blondel VD, Esch M, Chan C, Clerot F, Deville P, Huens E, Morlot F, Smoreda Z, Ziemlicki C. Data for development: The D4D challenge on mobile phone data. Computer Science. 2012. https://www.researchgate.net/publication/231513146_Data_for_Development_the_D4D_Challenge_on_Mobile_Phone_Data

[8] Department of Transport. Key Results of the National Household Travel Survey – Final Report. Department of Transport, Pretoria, South Africa. 2005. Available from: http://www.arrivealive.co.za/pages.aspx?nc=household

Chapter 6

Adaptive Security Framework in Internet of Things (IoT) for Providing Mobile Cloud Computing

Feda AlShahwan

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.75190

Abstract

Internet of Things (IoT) has immense potential to change many of our daily activities, routines and behaviors. The pervasive nature of the information sources means that a great amount of data pertaining to possibly every aspect of human activity, both public and private, will be produced, transmitted, collected, stored and processed. Consequently, integrity and confidentiality of transmitted data as well as the authentication of (and trust in) the services that offer the data is crucial. Hence, security is a critical functionality for the IoT. Enormous growth of mobile devices capability, critical automation of industry fields and the widespread of wireless communication cast need for seamless provision of mobile web services in the **Internet of Things (IoT)** environment. These are enriched by mobile cloud computing. However, it poses a challenge for its reliability, data authentication, power consumption and security issues. There is also a need for auto self-operated sensors for geo-sensing, agriculture, automatic cars, factories, roads, medicals application and more. IoT is still highly not reliable in points of integration between how its devices are connected, that is, there is poor utilization of the existing IP security protocols. In this chapter, we propose a deep penetration method for the IoT connected set of devices, along with the mobile cloud. An architecture and testing framework for providing mobile cloud computing in the IoT that is based on the object security, power utilization, latency measures and packet loss rate is explained. Our solution is based on the use of existing security protocols between clients and the mobile hosts as well as a key management protocol between the individual mobile hosts implementing an out-of-band key exchange that is simple in practice, flexible and secure. We study the performance of this approach by evaluating a prototype implementation of our security framework. This chapter, in a preliminary manner, discusses the threats, hacks, misguided packets and over read sensor message. These packets are then translated by hardware and pushed through the web for later-on action or support. Our testing of a set of sensor-triggered scenario and setup clearly indicates the security threats from wireless connected small LAN environments and the overestimated sensor messages resulting from the initial set of the sensor readings, while we emphasize more on the security level of the web services serving the IoT-connected device. Also, we add a remark on how mobile web services

and their enabling devices are by far vulnerable to a 4G hack over the utilization of power pack and a serious battery use power draining issues.

Keywords: IoT, mobile cloud, adaptive

1. Introduction

1.1. History of applications

Internet protocols were introduced as safe and secure data packets. These are the solid protocols to capture the packets within the services provided and served by the Internet, as a **World Wide Web** (WWW), as **Fiber to the Premises** (FTTP), Mail, video streaming and serving other protocols, presenting an edge to the web. In the last decade, a new era of security was introduced while testing and applying the new sensors and web service to the scene of Internet as more and more automation was introduced and tested for manufacturing purposes, statistics, traffic jam monitoring, file processing within factories, energy saving techniques, medical sensitive data and alerts applications.

Alert application includes measuring on a remote site the systolic and diastolic pressure, patient's heart rate, pulse of a patient and sending out the sugar glucose far from hospital. In addition to this, testing results from a patient's home, while he or she performs the test. Smartphone scans and senses the result and sends it via the web (an Internet protocol), through a smartphone application.

While using a wireless network, WiMAX, 3G, LTE, or 4th generation communication protocol, networks or simply a fixed network such as **Asymmetric Digital Subscriber Line** (ADSL) services (medium) known as industrial services. For example, an immediate auto shut off power is required for an overheated oven in an electric factory or an overcrowded traffic road. This helps to send a report of a series of processed photos to the operation center periodically, which would alert either a police patrol or movement to the site to clear the crowd or to solve the accident caused by traffic jam. The same set of steps are presented to show a critical case or a critical disruption of the normal settled case or the normal routine and this is when an indict application for Internet came to the scene of technical processed signals and connected smart devices.

Internet security is a major research topic in the field of computing and parallel processing, networking and data network design. In the past decade, a lot of researches have been done into the investigation of IoT. There are many faces to how such entity or terminology is defined, mainly, the term IoT refers to Internet-connected objects that are smart in a computational and connectivity manner. These objects are able also to compute, detect and communicate while making measurements to various functions. Functions include civil, domestic, manufacturing, industrial applications, automation and medical applications that bring new protocols to life, such as **Time Division Multiple Access** (TDMA—collision-free protocol), **Carrier Sense Multiple Access** (CSMA—slow and low traffic level). These protocols are applied to sensors

of the IoT system and form the backbone for sensors' communication in IoT system. An energy efficient MAC protocol and appropriate routing protocols are required in the IoT networks with limited resources. Several MAC [3] protocols have been proposed for various domains with TDMA, CSMA and **Frequency Division Multiple Access** (FDMA). These protocols are collision free. However, they require additional complexity to the sensors. Moreover, none of these protocols are accepted as a standard. Therefore, the significance of this scenario requires further research.

The connection is not stable for a number of reasons. For instance, the battery of the sensor may drain out, the wireless communication can be interrupted or a sensor drops out. Consequently, a methodology for self-adapting to the IoT system must be applied that allows for multi-path routing scheme. Multi-path routing protocols are used in mobile ad hoc networks and terrestrial WSNs [4, 5]. They are mainly divided into three categories, namely data centric, location based and hierarchical [6]. This classification is based on different application domains. Data-centric protocols are query based and they depend on the naming of desired data, which helps in eliminating unessential transmissions. Location-based protocols utilize the position information to relay the data to the desired regions rather than the whole network [7]. Hierarchical protocols aim at clustering the nodes so that cluster heads can do some aggregation and reduction of data in order to save energy. The main challenge for the existing routing protocols is preserving energy. This is due to the scarcity of resources. Energy in the IoT network will dominate the number of hops in the multi-hop scenario.

Thus, next era in the field of networks will be outside the realm of the traditional static network. In the IoT system, many of the objects that surround us in our daily life like homes, medical centers, factories, hospitals or government processes areas and universities will be active via web services. IoT smart items or gadgets are simply objects of the network that can receive, send and translate information through a **Transmission Control Protocol** (TCP) protocol or using sensor elements that can convert their sensors into signals.

The information and communication systems of the IoT networks involve a significant amount of data that have to be stored, processed and interpreted in a seamless, efficient and easily presented form. New sensor network technologies will emerge to meet the enormous amount of data and the new challenges in this system. This model will consist of session (alert, emails triggers, reports, actions movement, stopping of movement, narrow down, widening, etc.) that is delivered in a seamless uninterruptible and efficient manner. Cloud computing can provide the virtual infrastructure for such computing model which integrates monitoring devices, storage devices, sensors, etc.

IoT requires the usage of the limited network resources. From the existing networks and context-aware computation emerge the application of smart connections. The instant presence of the data and the high-speed communication networks are a result of the growing presence of LAN, 3G, WiFi, WiMax, 4G and LTE wireless Internet access. However, for the successful emergence of the IoT vision, the computing system will need to convert from the traditional mobile computing scenarios that use smartphones and wireless network and move into connecting smart objects and embedded intelligent devices. This transition for the IoT demands the following [8]:

- A public or private accessible and shared environment that considers the context of its users and their appliances
- Software structure and the communication networks to process and transfer the relevant data, information to where it is related
- The analytic tools in the IoT that have the characteristic of automation and adaptive behavior

Smart connectivity and context-aware computation of the IoT system can be accomplished with the application of these abovementioned fundamentals. IoT is a representation of a **Network of Things** (NoT); more clearly, IoT has its own objects that are connected to the Internet, while NoT can be considered as **Local Area Network** (LAN), with none of its objects connected to the Internet. Social media networks, sensor networks are all versions of NoTs.

It is common to call NoT within a work environment as an enterprise-based application. The Information that is gathered and processed in these networks is tailored to be used only by the enterprise owners and the data may be revealed selectively [9]. An example of NoT applications is environmental monitoring, which is implemented to monitor and track the number of facility users and manage the utilities. For example, controls AC, electricity, alerts, heat, ventilation, and power.

The evolution of the current Internet into a network of interconnected objects, or gadgets, not only sense information from the surroundings and interacts with physical world but allows also using existing Internet standards to provide services for information transfer, analysis of data and web services. Web services are manipulated by the abundant devices and are accessed by the open wireless technology such as bluetooth, WiFi, **Global System for Mobile communication** (GSM), Wi-Max and **Digital Subscriber Line** (DSL) data access, along with tailored sniffers and sniff blocks [10]. Recently in 2012, the number of interconnected things invaded the lives of a visible number of individuals. As we are working on this chapter, there are 9 billion interconnected things and it is expected to reach 20 billion devices by 2020.

According to the above Gartner's IT Hype Cycle [4] (**Figure 1**), IoT has been identified as one of the most emerging technologies in IT. A Hype Cycle is a way of representing the emergence, maturity, adoption and impact on applications of specific technologies [11]. It is obvious that IoT will take 5–10 years for market adoption. Furthermore, mobile applications and IoT will be the most disruptive class of technologies over the next 10 years. Gartner explains IoT as a network of physical objects that contain embedded technology to communicate and sense or interact with their internal states or the external environment. But, for the successful spread of Internet of Things, the computing criterion need to go beyond traditional mobile computing and evolve into connecting everyday existing objects and embedding intelligence into our environment [12]. **Table 1** below describes the smart environment application domain, where smart is a reference of a direct intelligent part or component of the network or IoT.

Table 1 explains the type of the network against criterion. For the smart home, smart retail, network size is expected to be small and the numbers of users are few. While for smart agriculture and smart water the network size need to be large as the users are not few. The mode of energy in all the application domains are either rechargeable or energy harvesting.

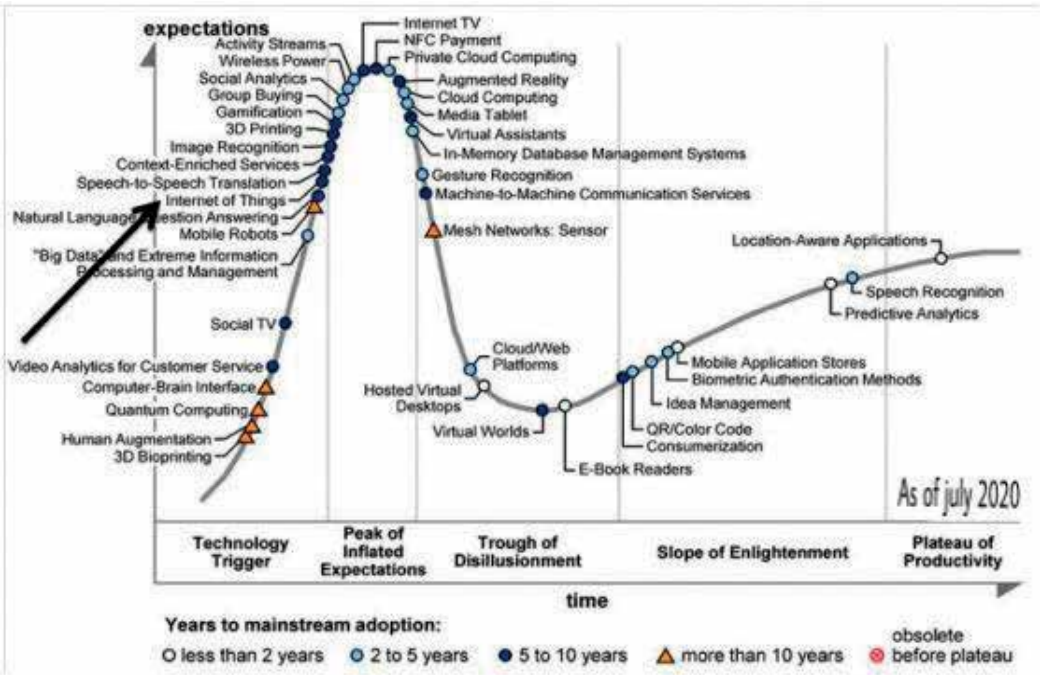

Figure 1. Cycle of technologies during the years.

	Smart Home/Office	Smart Retail	Smart City	Smart Agriculture/Forest	Smart Water	Smart transportation
Network Size	Small	Small	Medium	Medium/Large	Large	Large
Users	Very few, family members	Few, community level	Many, policy makers, general public	Few, landowners, policy makers	Few, government	Large, general public
Energy	Rechargeable battery	Rechargeable battery	Rechargeable battery, Energy harvesting	Energy harvesting	Energy harvesting	Rechargeable battery,Energy
Internet connectivity	Wi-Fi, 3G, 4G LTE backbone	Wi-Fi, 3G, 4G LTE backbone	Wi-Fi, 3G, 4G LTE backbone	Wi-Fi, Satellite communication	Satellite Communication,	Wi-Fi, Satellite Communication
Data management	Local server	Local server	Shared server	Local server, Shared server	Shared server	Shared server
IoT Devices	RFID, WSN	Smart Retail	RFID, WSN	WSN	Single sensors	RFID, WSN, Single sensors
Bandwidth requirement	Small	Small	Large	Medium	Medium	Medium/Large
Example testbeds	Aware Home [31]	SAP Future retail center [32]	Smart Santander[33], CitySense [34]	SiSViA [35]	GBROOS [36], SEMAT [37]	A few trial implementations [38,39]

Table 1. Smart environment application domain.

2. IoT and cloud internet in the coming years

Oxford defines the IoT as a proposed development of the Internet in which everyday objects have network connectivity, thus enabling them to send and receive data. The IoT is a term used to describe all connected objects nodes and computers that can and will perform a pre-defined set and a measurable set or group of functions and actions that send reports which react to a certain probe, indicating a signal as a response a to a unique sensor trigger.

The enormous spreading of smartphones and other handheld devices in the current decade has changed the computing environment. It becomes more autonomous, interactive and informative. Consequently, it motivated the researchers to focus on human-to-human interface in the late 1980s. As a result, the Ubiquitous Computing (UbiComp) technology has emerged. Mark Weiser, the forefather of UbiComp, defined a smart environment [13] as the physical world that is richly and invisibly interwoven with sensors, actuators, displays, and computational elements, embedded seamlessly in the everyday objects of our lives and connected through a continuous network.

The creation of the Internet has marked a foremost milestone toward achieving UbiComp's vision which enables individual devices to communicate with any other device in the world.

Caceres and Friday [14] discuss the progress, opportunities and challenges during the 20-year anniversary of UbiComp. They discuss the building blocks of UbiComp and the characteristics of the system to adapt to the changing world. More importantly, they identify two critical technologies for growing the UbiComp infrastructure *Cloud Computing* and the *IoT*.

The advancements and convergence of micro-electro-mechanical systems (MEMS) technology, wireless communications and digital electronics has resulted in the development of miniature devices having the ability to sense, compute and communicate wirelessly in short distances. These miniature devices called nodes interconnect to form a wireless sensor networks (WSN) and find wide application in environmental monitoring, infrastructure monitoring, traffic monitoring, retail, etc., [2, 15].

In all of the previous cases, there has to be a governing protocol that authenticates measures and monitors the amount of work needed to perform the predescribed functions; we still have to emphasize the need for new protocols to control and stabilize the IoT environments, such CoCa.

CoCa is an example of a service infrastructure for the IoT that provides pervasive services and supports connecting embedded objects, backend systems and mobile devices in a seamless manner.

In this chapter, we shall investigate the security issues and threats that are passed through a uniform and stable environment making use of a preset of works function and signals, reports, etc., working under an IoT rule. More importantly, we shall put and entail the environment to the security text needed to ensure that there is a stable solid system with a model for traffic and car, road monitoring system, enabled with 5 sensors for flow, speed, sudden jam and total size of roads. Here, we use a model for a cloud computing, connected to a net system named CIT—prepared and authored by Kuwait University undergraduate to manage and extract useful data and send it over to the control room by making use of its protocols. We shall narrow the testing or hypothesis of this research paper to tackle and modify the working conditions of a Wi-Fi operated medium of an IoT scenarios. This chapter aims measure the lack of security and proposes more protocols and shows integration effort to the standards that govern any IoT environment. We propose the chapter as a proof that the IoT is not secure enough to withstand critical application, industrial function and high security. In addition to this, the abundance of Wi-Fi protocols is not the only reason and rules that not only put the IoT critical applications at risk but also leaves a lot of questioning on how the procedure or the IoT is designed to operate.

http://dx.doi.org/10.5772/intechopen.75190

We propose that the IoT is not an optimal secure environment when critical applications are needed, whether in wireless connected machines or wired data networks.

3. Definitions, terminology, and elements

3.1. Definitions

As identified by Atzori et al. [1], IoT can be realized in three paradigms—Internet oriented (middleware), things oriented (sensors) and semantic oriented (knowledge). IoT can be more useful in applications where the three paradigms exist.

The RFID group defines IoT as the worldwide network of interconnected objects uniquely addressable based on standard communication protocols [16].

IoT has been defined by a group European research projects as [17] things that are active participants in business, information and social processes where they are enabled to interact and communicate among themselves and with the environment by exchanging data and information sensed about the environment, while reacting autonomously to the real/physical world events and influencing it by running processes that trigger actions and create services with or without direct human intervention.

Smart environment [18] utilizes information and communications protocols to make the critical, medical or emergency data and health hazard data structure components, services of a whole town management of traffic or, education, healthcare, public health, real estate and other utilities more aware, interactive and efficient.

In our definition, we make the definition more user-oriented and as close to real life as possible, thus generating data objects such as sensors, emails, physical alerts, triggers alarms, messages, emergency warning, actions, and so on, without restricting it to any standard communication protocol. This will provide significant amount of applications. Moreover, allow using the traditional existing protocols to deploy long lasting applications on the fly and at any time. Thus, our definition of IoT for smart environments is the communication of sensors and actuating devices to provide information across different platforms that can be accessed through a unified infrastructure, developing and enabling innovative applications. This is achieved by seamless large-scale sensing, data analytics and smart information interpretation using UbiComp and cloud computing.

3.1.1. IoT components

The components for IoT can be classified from high-level perspective into three categories that enables seamless UbiComp. Each category can be classified into more taxonomies as found in [1, 13, 19]. The three IoT components are:

a. Hardware—made up of sensors, actuators and embedded communication hardware

b. Middleware—on demand storage and computing tools for data analytics

c. Presentation—novel easy to understand visualization and interpretation tools which can be widely accessed on different platforms and which can be designed for different applications.

The major properties compose or make the security issues of IoT and also ask for more security reliable measure for IoT, as listed below:

Embedded utilization: Most IoT devices have only one single function or use (such as trigger light, turn on power, emerge alerts, message, control, sending datam monote house appliance, etc.); as a direct result, the recognition to a unique device makes up a pattern that gives an easy profile or can be filtered into a pattern.

Divers: The devices of IoT are able to work across multiple spanned devices of computing from low end low frequency RFID to full function computers, thus the privacy policy must encompass all the range of computations.

Scal: These devices, IoT functional ones, are easy to use and are added into the market with simply easy to use applications; thus it makes it hard for users to monitor the privacy and security issues at question.

Mobile: Most if not all IoT devices are mobile and usually connected to the Internet via a large multiple set of services or service providers.

Wireless: IoT devices, in most cases, get connected, and thus enabled, and become functional to the Internet via a large list of wireless protocol namely Wi-Fi, 802.11,WiMax, GSM, Bluetooth, etc.

A set of pervasive computing devices that monitors the technology applied to IoT is a characteristic that creates a set of challenges that need to be tackled. The challenges are listed in the following section.

3.1.2. Challenges and issues in IoT

There are some issues that act as a barrier against the spread of IoT:

1. Heterogeneity of devices and its management
2. Privacy and security of the data packets moved (transported) across these devices; thus a certain level of reliability must be built
3. Network knowledge and content of the packets needs to be known, measured and identified

The challenges will directly dictate a development of new algorithms that are encrypted in an efficient way to provide a minimum level of security for IoT connected devices and its environment, namely a need for a confidential and highly integral level of data communicating across such service level connected devices,

The following figure illustrates how to provide the level of security for IoT (**Figure 2**):

1. User privacy: Any user has to be secured enough not to have his/her unified information scattered across unwanted part of a private e-LAN or the Internet in general.

http://dx.doi.org/10.5772/intechopen.75190

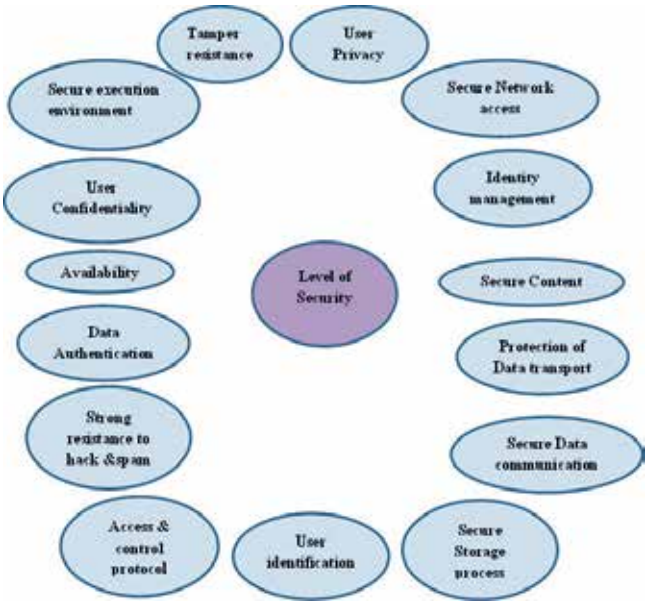

Figure 2. IoT security levels.

2. User confidentiality: This item controls the need and actions to be taken that the provider of services in an IoT is able to deduct from observing the use of the look ahead system concerned with a specific user; at least, this deduction should be extremely difficult to take place.
3. Authentication of data: All data and packet data with received information from IoT devices and user side or control units have to be authenticated.
4. Strong resistance to hack and spam attacks: The IoT systems should avoid having one point of failure and should be able to recover from nodes or multiple nodes failure, also avoid, if necessary, single points of failure.
5. Protocol for access and control: All information service providers must be able to adhere to access control protocols, to govern the way packets are retrieved and used within the network.
6. Protection of data transport: This part is usually deeply discussed and portrayed by telecom protocols and its supporting security levels.
7. User identification: It refers to the action by which the user validating users takes place before granting access to the system.
8. Secure storage process: This involves confidentiality and access control of critical and sensitive parts of the packets and information that are stored in the network.
9. Identity management: It is a wide look up area that handles identifying people and their connected things in an IoT system and controlling their access to libraries and services within that system by linking user profiles and their access levels and with the created user rights.

10. Secure data communication: This lists authenticating communicating peers, ensuring confidentiality and complex process of communication data, thus filtering loss of data transaction, hiding and protecting the user profile details of common communicating protocol.
11. Availability: Availability refers to complete allocation of authorized persons or systems only can access the system and to deny access or services to authorize users.
12. Secure network access: This provides a network connection or services that can work only if the device is linked and validated.
13. Secure content: Content secure transaction is the key to secure IoT, namely, using Digital Rights Management (DRM) protects the rights of the digital files moving across the IoT system or network.
14. Secure execution environment: It refers to a secure, managed-application environment that is dedicated with a set of rules for preventing the system from hacks and attacks or suspicious applications.
15. Tamper resistance: The full protection to the IoT system even if the logical part is down, it can resist hacks even with a physical attempt or if the device falls in the wrong hands or threats from outside parties or hackers.

4. Securities and threat taxonomy for IoT

4.1. Security and threat idiom for the IoT connected environment

Typically, an IoT connected set of devices will have it is own security threats; meanwhile, the new algorithms or technologies in IoT security might be able to put some reliable solutions. Security threat level of IoT is a main topic that needs to be addressed and standardized. One typical example is how infinite its applications can be as it is basically linking or enabling devices to be smart and connected to the Internet.

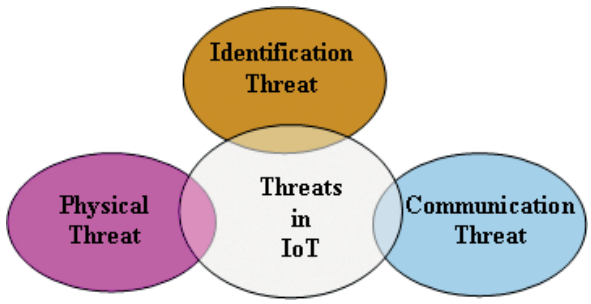

Figure 3. Threats taxonomy for IoT.

http://dx.doi.org/10.5772/intechopen.75190

We can safely state that the IoT is coupled with multiple security threats and alters overall information security risk profile, although the implementation of new protocols and restriction may help IoT fight against threats and vulnerabilities. IoT security is basically a data management topic and a highly rich research area. Effective management of these threats that are linked with IoT needs a deep and thorough evaluation of risk given the environment and development of a plan to go through clear and calculated risk.

The various threats associated with the use of IoT [13] are listed in the following paragraphs (**Figure 3**):

Identification threat covers determination of unique device/user/session with authentication, authorization, accounting, and provisioning.

Communication threat handles a d.enial-of-service attack (DoS) and it occurs when an attacker continually bombards a targeted AP (Access Point) or network with bogus requests, premature successful connection messages, failure messages, and/or other commands.

Physical threat includes micro probing and reverse engineering causing serious security problem by directly tampering the hardware components. Some types of physical attack require expensive material because of which they are relatively hard to perform. Some examples are de-packaging of chip, layout reconstruction, micro-probing.

Embedded security threat model will span all the threats at physical and MAC layer [20, 21]. Security threats such as device and data tampering, side channel analysis, bus monitoring, etc., will be the concerns at device level.

Storage management has crucial impact on the key management to achieve confidentiality and integrity. We must also be careful in choosing which cryptographic components to use as the building blocks since, for example, the cipher texts for some public key encryption.

5. Conclusion

The IoT [1] has immense potential to change many of our daily activities, routines and behaviors. The pervasive nature of the information sources means that a great amount of data pertaining to possibly every aspect of human activity, both public and private, will be produced, transmitted, collected, stored and processed. Consequently, integrity and confidentiality of transmitted data as well as the authentication of (and trust in) the services that offer the data is crucial. Hence, security is a critical functionality for the IoT [14].

Wireless data networks, are prone to a large number of attacks such as eavesdropping, spoofing, denial of service, and so on. Legacy Internet systems mitigate these attacks by relying on link layer, network layer, transport layer or application layer encryption and authentication of the underlying data. Though some of these solutions are applicable to the IoT domain, the inherently limited processing and communication capabilities of IoT devices prevent the use of full-fledged security suites.

Author details

Feda AlShahwan

Address all correspondence to: fa.alshahwan@paaet.edu.kw

College of Technological Studies, Public Authority for Applied Education and Training, Kuwait

References

[1] Atzori L et al. The internet of things: A survey. Computer Networks. 2010;**54**:2787-2805

[2] Woo A, Culler DE. A transmission control scheme for media access in sensor networks. In: Proceedings of the 7th Annual International Conference on Mobile Computing and Networking. 2001. pp. 221-235

[3] Juels A. RFID Security and Privacy. Springer; 2012

[4] Aoudia FA et al. A generic framework for Modeling MAC protocols in wireless sensor networks. IEEE/ACM Transactions on Networking; 2016

[5] Hasan MZ et al. A survey on multipath routing protocols for QoS assurances in real-time wireless multimedia sensor networks. IEEE Communications Surveys & Tutorials. 2017

[6] Bakht MP, Shaikh AA. Routing Techniques in Wireless Sensor Networks: Review and Survey. Journal of Applied and Emerging Sciences. 2016;**6**:18-23

[7] Amsalu SB et al. Design and performance evaluation of an energy efficient routing protocol for wireless sensor networks. In: 2016 Annual Conference on Information Science and Systems (CISS). 2016. pp. 48-53

[8] Botta A et al. Integration of cloud computing and internet of things: A survey. Future Generation Computer Systems. 2016;**56**:684-700

[9] Urzaiz G et al. The advanced network of things: A middleware to provide enhanced performance and functionality in IoT. In: Ubiquitous Computing and Ambient Intelligence: 10th International Conference, UCAmI 2016, San Bartolomé de Tirajana, Gran Canaria, Spain, November 29–December 2, 2016, Part II 10. 2016. pp. 284-294

[10] Negi V. (2014). Available: https://www.slideshare.net/vikrantnegi007/internet-of-things-seminar

[11] Hype Cycle. Available: https://en.wikipedia.org/wiki/Hype_cycle

[12] Sudarshan SK et al. A Comprehensive Study of Mobile Sensing and Cloud Services

[13] Streitz N, Markopoulos P. Distributed, Ambient and Pervasive Interactions: Proceedings of 4th International Conference, DAPI 2016, Held as Part of HCI International 2016, Toronto, ON, Canada: Springer; July 17-22, 2016. Vol. 9749. 2016

[14] Gao L-j, Chen Z-g. Security in Next-Generation Wireless Sensor Networks

[15] Herr DA et al. Protocol selection for transmission control protocol/internet protocol (TCP/IP). Google Patents; 2015

[16] Gubbi J et al. Internet of things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems. 2013;**29**:1645-1660

[17] Tomar GS et al. The Human Element of Big Data: Issues, Analytics, and Performance. CRC Press; 2016

[18] Li S et al. The internet of things: A survey. Information Systems Frontiers. 2015;**17**:243

[19] Crooks A et al. Creating smart buildings and cities. IEEE Pervasive Computing. 2017; **16**:23-25

[20] Shih E et al. Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks. In: Proceedings of the 7th Annual International Conference on Mobile Computing and Networking, 2001. pp. 272-287

[21] Ye W et al. An energy-efficient MAC protocol for wireless sensor networks," in INFOCOM 2002. Twenty-first annual joint conference of the IEEE computer and communications societies. Proceedings. IEEE, 2002. pp. 1567-1576

Edited by Mutamed Khatib and Nael Salman

Nowadays, mobile communication services are penetrating into our society at an explosive growth rate. Applications in mobile devices offer limitations, restriction, and guidelines on how mobile software can be used in order to simplify the mobile usage. As smart phones and tablets are becoming the daily computing device of choice for young ages, it is expected that mobile applications and services should be as flexible, high quality, and secure as the desktop systems. In this book, latest trends in mobile computing will be discussed. In the first section, cloud computing topics will be discussed widely into four chapters to give information to the reader about topics such as challenges, services, edge computing, and distributed clouds needed to integrate this promising issue into the next generation.

Published in London, UK

IntechOpen

ISBN 978-1-83881-504-2