Service-Oriented Architecture for Patient-Centric eHealth Solutions

Yohanes Baptista Dafferianto Trinugroho

Service-Oriented Architecture for Patient-Centric eHealth Solutions

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of Philosophiae Doctor (PhD) in Information and Communication Technology

University of Agder
Faculty of Engineering and Science
2014

Doctoral Dissertation by the University of Agder 92
ISBN: 978-82-7117-777-5
ISSN: 1504-9272

Printed by the Printing Office, University of Agder
Kristiansand

"Ask, and it shall be given you; seek, and ye shall find; knock, and it shall be opened unto you"

—Matthew 7:7

To those I love

Preface and Acknowledgements

This dissertation is a result of a research work carried out at the Department of Information of Communication Technology (ICT), University of Agder (UiA) in Grimstad, Norway, from November 2010 to December 2013. The research work is fully sponsored by UiA under PhD research fellowship project grant number 63730.

My interest in ICT dated back in late 1980s/early 1990s when my parents bought an Atari 130XE computer. Although I and my brother used it mostly for playing games that were recorded in cassettes, I began to understand a bit of BASIC programming language when my father invited a BASIC programmer to our apartment to teach him and my brother how to program. I decided to focus my career within the ICT domain after completing high school, and was enrolled in a computer science bachelor programme at the Gadjah Mada University in Yogyakarta, Indonesia. My interest in ICT grew even stronger after I obtained my bachelor's degree, and I continued pursuing a double master's degree at the RWTH Aachen University in Aachen, Germany, and the University of Trento in Trento, Italy, which was fully sponsored by the European Commission. I am very grateful to have had the opportunity to continue my formal education even further at doctoral level at the University of Agder in Grimstad, Norway.

I would like to express my sincere gratitude to my main supervisor, Professor Frank Reichert, for the precious time he has given me in directing and supervising my research work, and also for inviting me to join the Agder Mobility Lab (AML). His continuous support and encouragement have kept me motivated to finish what I started. I would also like to sincerely thank my co-supervisor, Professor Rune Fensli, for guiding and involving me in different projects through the course of my research work. I am very grateful to have joined the eHealth group where I learned many new things that I believe will be important in my future career. I thank all members of the AML and the eHealth groups for the stimulating discussions and interactions throughout the course of my research fellowship. I should also thank my supervisors at Ericsson Research in Aachen, Germany, Dr. Andreas Fasbender and Martin Gerdes for recommending me during the application process to the PhD programme.

I would like to thank the head of the ICT department, Professor Andreas Prinz, for all his help and generous facilities, especially when it comes to travelling to conferences as well as Christmas present surprises. I would also like to thank the ICT department's administrative staff who are responsible for helping PhD fellows, Trine Tønnessen, Tonje Sti, and Emma Elizabeth Horneman, for their kind help with day-to-day practicalities of the PhD programme. I also thank Professor Frank Li for the opportunity to be involved in the S2EuNet project for a research visitation.

The PhD research fellowship experience would not be complete without the companionship of other former and ongoing PhD research fellows, and thus I would like to thank my office mates, Ghislain Maurice Isabwe and Martin Gerdes, as well as all other PhD fellows that I cannot list one by one. I would also like to thank master students whom I supervised for the great collaborations.

Last but not least, I would like to sincerely thank my family and a special friend for their loving care. Their constant support has brightened up my days, and has kept the candle of hope lit through all the ups and downs of life during my stay in Grimstad.

Yohanes Baptista Dafferianto Trinugroho
February 2014
Grimstad, Norway

Abstract

The world is in shortage of about 7.2 million healthcare workers in 2013, and the figure is estimated to grow to 12.9 million by 2035, according to the World Health Organization (WHO). On the other hand, the median age of the world's population was predicted to increase from 26.6 years in 2000 to 37.3 years in 2050, and then to 45.6 years in 2100. Thus further escalating the need for new and efficient healthcare solutions.

Telehealth, telecare, and Ambient Assisted Living (AAL) solutions promise to make healthcare services more sustainable, and to enable patients to live more independently and with a higher quality of life at their homes. Smart homes will host intelligent, connected devices that integrate with the Internet of Things (IoT) to form the basis of new and advanced healthcare systems. However, a number of challenges needs to be addressed before this vision can be actualised. These challenges include flexible integration, rapid service development and deployment, mobility, unified abstraction, scalability and high availability, security and privacy.

This thesis presents an integration architecture based on Service-Oriented Architecture (SOA) that enables novel healthcare services to be developed rapidly by utilising capabilities of various devices in the patients' surroundings. Special attention is given to a service broker component, the Information Integration Platform (IIP), that has been developed to bridge communications between everyday objects and Internet-based services following the Enterprise Service Bus (ESB) principles. It exposes its functionalities through a set of RESTful Web services, and maintains a unified information model which enables various applications to access in a uniform way. The IIP breaks the traditional vertical "silo" approach of integration, and handles information dissemination task between information providers and consumers by adopting a publish/subscribe messaging pattern.

The feasibility of the IIP solution is evaluated both through prototyping and testing the platform's representative healthcare services, e.g., remote health monitoring and emergency alarms. Experiments conducted on the IIP reveal how performance aspects are affected by needs for security, privacy, high availability, and scalability.

Contents

List of Figures

List of Tables

Abbreviations

AAL	Ambient Assisted Living
AALIANCE	Ambient Assisted Living Innovation Alliance
AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
BPM	Business Process Modelling
BSN	Body Sensor Networks
CoAP	Constrained Application Protocol
COPD	Chronic Obstructive Pulmonary Disease
CRUD	Create, Read, Update, Delete
DSL	Digital Subscriber Line
EAI	Enterprise Application Integration
ED	Emergency Department
EDA	Event-Driven Architecture
EDGE	Enhanced Data Rates for GSM Evolution
EHR	Electronic Health Record
ESB	Enterprise Service Bus
ETSI	European Telecommunications Standards Institute
GP	General Practitioner
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ICT	Information and Communications Technology
IDE	Integrated Development Environment
IIP	Information Integration Platform
IoT	Internet of Things
ISDN	Integrated Services Digital Network
JDBC	Java Database Connectivity

JEE	Java Enterprise Edition
JPA	Java Persistence API
JSON	JavaScript Object Notation
M2M	Machine-to-Machine
MQTT	Message Queuing Telemetry Transport
NHN	Norwegian Health Network
NIHCM	National Institute for Health Care Management
OWL	Web Ontology Language
PHI	Paraprofessional Healthcare Institute
QoS	Quality of Service
REST	Representational State Transfer
RPC	Remote Procedure Call
SGML	Standard General Markup Language
SHOM	Smart Home Ontology Model
SIM	Subscriber Identity Module
SMS	Short Message Service
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SpO2	Pulse Oximetry
SWRL	Semantic Web Rule Language
TLS	Transport Layer Security
UDDI	Universal Description, Discovery and Integration
UiA	University of Agder
UML	Unified Modelling Language
UMTS	Universal Mobile Telecommunications System
UNFPA	United Nations Population Fund
URI	Uniform Resource Identifier
VLAN	Virtual Local Area Network
VPN	Virtual Private Network
WCDMA	Wideband Code Division Multiple Access
WHO	World Health Organization
WiFi	Wireless Fidelity
WSDL	Web Services Description Language
WS-BPEL	Web Services Business Process Execution Language
XML	Extensible Markup Language

Part I

Chapter 1

Introduction

1.1 Background and Motivation

Throughout the history of mankind, happiness and prosperity have been the main goals of many people in living their lives. Although the former is a very abstract concept and most often subjective, the latter can normally be quantified or measured based on certain agreed standards in the society. The metrics can differ in different time periods and in different cultures. The Legatum Institute[1], an organisation that researches global prosperity, revealed in its 2013 prosperity index report that inspite of tumultuous events during the last five years, global prosperity is actually on the rise [5]. Life expectancy normally increases in proportion to the rise of prosperity. Combined with the decreasing number of birthrate, this situation results in an increasing number of elderly population around the globe. Lutz et al. [6] predicted the median age of the world's population will increase from 26.6 years in 2000 to 37.3 years in 2050, and then to 45.6 years in 2100. According to the United Nations Population Fund (UNFPA)[2] report [1], the number of people aged 60 and up will hit 2 billion mark in 2050 globally. And quite surprisingly, the biggest portion of this population group, around 80%, is predicted to be contributed by developing countries, as shown in Figure 1.1. Nonetheless, the rapid growth of elderly population has been a major trend in many developed countries. In Canada, for example, the ratio projection of the population age 65 and over to the population of traditional working age (18 – 64) will rise from 20% in 2006 to 46% in 2050 [7]. Norway, which has been ranked first in the Legatum Prosperity Index for five consecutive years since 2009, had around 625,000 people older than 67 years old in 2010, and the figure is expected to double in 2060 [8]. This situation, however, is not balanced

[1] http://www.li.com/
[2] http://www.unfpa.org/

with an increasing number of healthcare workforce, which widens the gap between demand and supply within the healthcare sector [9]. A report issued by the World Health Organization (WHO)[3] stated that the world is in shortage of about 7.2 million healthcare workers in 2013, and the figure is estimated to grow to 12.9 million by 2035 [10]. According to the report, one cause of the healthcare worker shortage is an aging workforce, with staff retiring or leaving the profession, coupled with not enough young people entering the profession or being adequately trained. Paraprofessional Healthcare Institute (PHI)[4], a non-profit organisation based in New York City, estimated the demand for direct-care workforce will approach 5 million by 2020 in the United States (including nursing aides, home health aides, and personal care aides) [11]. It is reported that one of the main challenges is to make caregiving attractive as a profession while still providing affordable care. The limited number of healthcare workforce has contributed to the increase of healthcare costs as well. In the United States, the total expenditure was around $2.6 trillion in 2010 alone, and the amount is expected to double within five years [12]. Figure 1.2 shows Norway's total health expenditure per capita from 1995 to 2011 according to WHO data[5].

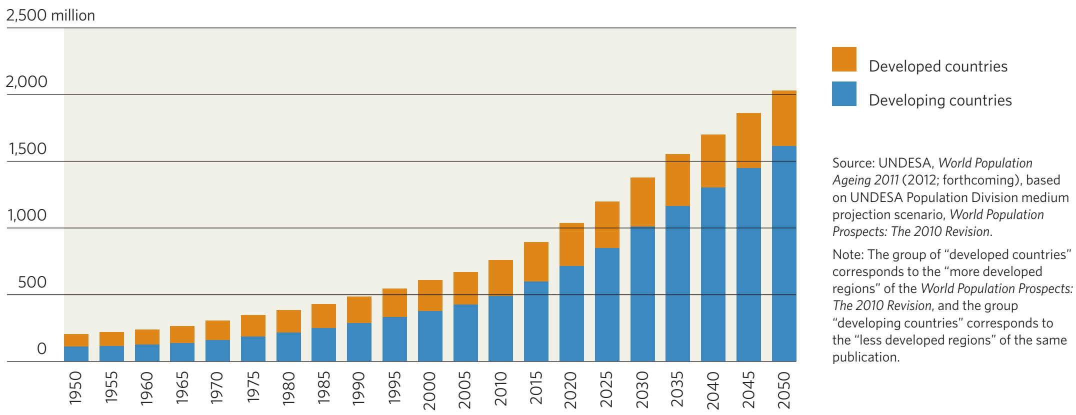


Figure 1.1: Number of people aged 60 or over [1]

Apart from adding more qualified healthcare workforce to handle the projected increasing number of elderly population, Information and Communications Technology (ICT) is expected to play a significant role in helping alleviate the burden of healthcare workers and optimise the use of available resources. Telehealth, for example, provides the capabilities to assist health maintenance and detection by utilising ICT, eliminating distance barrier between healthcare providers and patients

[3]http://www.who.int/
[4]http://www.phinational.org/
[5]http://www.who.int/countries/nor/en/

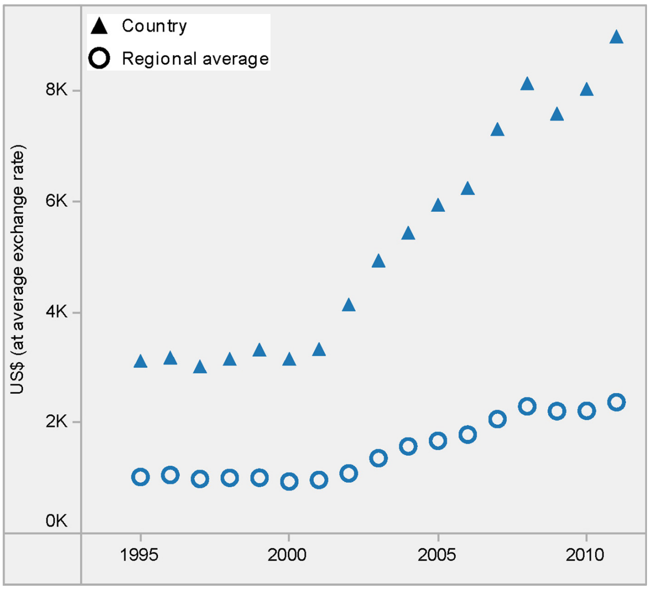


Figure 1.2: Norway's per capita total expenditure on health between 1995 and 2011

[13, 14, 15, 16, 17]. It can be further utilised to support emergency situation by means of remote alarming service that enables healthcare personnel to be notified in case of urgent situation occurred to the distantly-located patients. Novel ICT solutions will further reduce the time required for patients to stay at the healthcare premises, moving non-urgent treatments to the patients' homes. The patients can stay at their homes as long as possible where routine medical follow-ups can be achieved remotely as much as possible by utilising specialised home-care equipments. From this standpoint, not only will ICT make healthcare delivery more efficient, but also enables the patients to live more independently at their homes with high degree of self-care and self-management. It will of course require the patients to have a certain level of knowledge or ability to operate the healthcare equipments, but the next generation of elderly population, especially in developed countries, is expected to have more experience in interacting with ICT-related devices (e.g. smartphones, tablets, laptops) than the current generation's. In Norway, a public report used Mick Jagger as an icon representing the new generation elderly with active lifestyle [18]. This group of population is referred to as *digital immigrants*, who were not born into the digital world but have, at some later point in their lives, become fascinated by and adopted many aspects of digital technology [19]. They have been trained and got used to new ICT solutions such as the Internet, e-mail, and social media. In their elderly days, it is reasonable to expect them to require updated technology solutions, and this is one challenge that next-generation application and service developers should keep in mind.

The technologies and services being deployed at the patients' homes should support patient empowerment, which facilitates self-directed behaviour change of the patients. The greatest impact on the patients' health and well-being is a result of their self-management decisions and actions during the routine conduct of their daily lives. Healthcare professionals are responsible for ensuring their patients are making informed self-management decisions [20]. However, social interactions should not be limited to interactions between healthcare personnel and their patients. Involvement of relatives and colleagues is crucial for the encouragement of better lifestyle at home [21]. Healthcare social media can be utilised in realising this vision as a collaboration platform for virtual meetings [22] as shown in Figure 1.3.

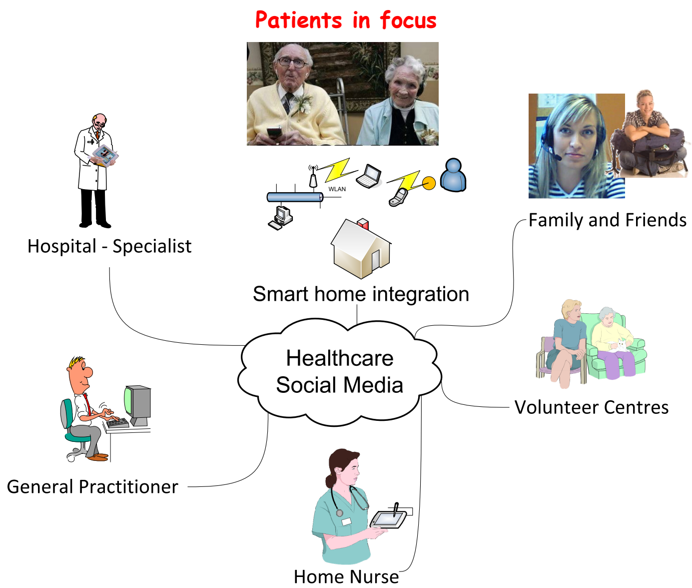


Figure 1.3: Healthcare social media for home patients

Future homecare technology should be as easy as possible for the patients to use, with high degree of automation. To realise this, intelligent networked sensors and actuators need to be installed at the patients' homes. Together with computer systems being deployed, the installed sensors and actuators transform the patients' homes into smart homes [23]. The current advancements in ICT have enabled the Internet of Things (IoT) vision to be realised by turning everyday objects to connected objects [24, 25, 26, 27, 28]. Off-the-shelf wireless wearable medical and fitness devices that can be found in consumer electronics stores nowadays play a big role in providing eHealth services to the patients as well. These devices can be used for monitoring the health condition of the patients both inside their homes and in outdoor environments. Computing resources in the smart homes should be able to collect all captured data and store them in either local data storages in the homes or remote data storage services in the "cloud".

However, most of the current available health-related devices are being designed to serve specific purposes with specialised applications/services being built exclusively on top. In other words, services are being closely integrated with devices in a vertical "silo" or "stovepipe" manner [29, 30, 31, 32], which limits the reusability of the gathered data by different applications/services. This makes novel service creation that combines data from different devices difficult, if not impossible, and thus similar data gathering functionalities are redundantly incorporated in different devices produced by different vendors. To make things worse, many deployed telehealth and telecare systems do not pay enough attention to the wider information systems architecture that could potentially generate unintended disbenefits in terms of creating data silos which may cause elderly people and patients harm or impede the ability of the clinicians and carers to treat them in an integrated fashion [33, 34, 35, 36, 37].

SOA (Service-Oriented Architecture) [38, 39, 40, 41, 42] has become a major trend in the last couple of years as a way to support business processes of organisations, especially by utilising Web services technology [43]. It is a paradigm that is well suited for tackling integration issues between different software components by separating implementation logics from the interfaces of services, and it can potentially be applied in homecare technology solutions to avoid vertical "silo" integration between devices and services. Automation that takes into consideration various types of data from different devices (including home appliances) can be realised much easier in a SOA environment than the tightly-coupled devices-services approach. New services with different algorithms or logics that utilise gathered data from different devices and sensors can be developed and deployed faster to support patients. By combining SOA and IoT in a smart home setting, it is possible to push forward healthcare services beyond the four walls of hospital and avoid long queues of physician appointments for non-critical treatments for patients in the future.

1.2 Challenges

Providing healthcare services that utilise ICT in a smart home environment and beyond has its own challenges. Some of the key challenges that this research work aims to address are described in the following points.

1. *Integration of various devices and services in a smart home environment*. In order to deliver healthcare services to the patients in their homes, different

devices and services from different vendors and healthcare providers (e.g. hospitals) with different technologies and standards need to be integrated in a common logical integration platform. This will enable new services to be developed by reusing the gathered data from different sensors and performing actions on exposed actuators' capabilities. This can be very challenging as many vendors use their own proprietary technologies and protocols for their products, and do not provide application programming interfaces (APIs) for either exposing the capabilities of the devices or the data that the devices gather or produce.

2. *Rapid service creation and modification*. Ambient assisted living (AAL) and eHealth services in a smart home environment may be designed to work locally without any dependency to external services. However, some services may require external services to be utilised, or the other way around, information gathered or produced by local services may be required by external services (e.g. hospital's remote health monitoring service). The requirements of these services may change frequently after their initial deployments, following the users' needs. Thus rapid service development and alteration should be supported for both in-home and external services. Deciding which data and to what degree it should be made available to external entities to be utilised in their services is a challenge by itself.

3. *Inside-outside smart home mobility of inhabitants and patients*. One of the main reasons for developing and deploying eHealth smart home solutions is to enable inhabitants and patients to live at their homes independently while still being in touch with remote healthcare personnel. Limiting them to always stay at home violates their independence as they may want or need to engage themselves in out-of-home activities (e.g. daily walk, shopping, visiting neighbours and family members, cultural events). Some portable wireless sensors and smart devices (e.g. smartphones, tablets) may be carried by the patients, but bulky home appliances that are used as part of eHealth services definitely stay inside their homes. eHealth services should be designed carefully to accommodate the inhabitants' and the patients' mobility inside and outside the smart homes.

4. *Information storage and modelling*. A plethora of services can be introduced in a smart home environment. The information that these services produce needs to be stored somewhere, either in a local data storage in the smart home or in a trusted third party "cloud" storage provider. To support man-

ageability (e.g. for matchmaking of services), metadata of every information should be stored and be made accessible, so that other services can accurately select which information to use according to their requirements. A unifying information model that integrates vendor-specific models is needed in order to enable consistent information view accross disparate business domains. In addition, to make the stored information more meaningful, the information should be managed in a certain way to maintain a knowledge base of various domains of interests that can be utilised for reasoning processes when needed.

5. *Dependability of eHealth services*. Littlewood and Strigini [44] suggested that dependability attributes include reliability, safety, security, and availability, while Avižienis et al. [45] defined attributes of dependability to comprise availability, reliability, safety, confidentiality, integrity, and maintainability. In the latter view, security is not seen as a standalone attribute, but rather as a combination of three attributes, namely confidentiality, integrity, and availability, which is in line with the CIA triad model [46]. Dissemination of health-related information of the patients to external entities out of their consent can be disastrous. While the information gathered from health-related devices should be made available to be utilised by different services, authentication and access control should be enforced to maintain high-level of security and privacy. However, usability and performance of eHealth solutions should not be completely compromised in favour of security and privacy. Striking a good balance between security and privacy on one hand, and usability and performance on the other hand, is another challenge when designing eHealth solutions in a smart home setting. Availability is another challenge that needs to be addressed in eHealth services which can be achieved by incorporating redundancy of service components.

1.3 Research Questions and Solution Requirements

This research work aims to contribute to the current healthcare delivery practices that involve ICT with the appropriate architecture and technical solutions which combine the current state-of-the-art principles, such as SOA, IoT, and cloud computing, for future-oriented more independently living patients. The output of this work is expected to provide an alternative guiding architecture that enables rapid development and deployment of new residential as well as out-of-home services for future elderly patients in particular and ordinary patients in general. Several requirements arose during the initial phase of this work:

1. **Flexible integration:** *How can different devices be integrated to support eHealth services and to avoid vertical "silos" in a smart home environment?*

2. **Rapid service development and deployment:** *How can eHealth services be developed and deployed rapidly to support changing needs of the inhabitants and the patients in a smart home environment?*

3. **Mobility:** *How should eHealth services be designed to support the inhabitants' and the patients' mobility inside and outside their homes?*

4. **Unified abstraction:** *How should information gathered from various devices be modelled in order to provide a consistent view across different domains?*

5. **Scalability and high availability:** *How can eHealth services accommodate an increasing number of users while minimising downtime?*

6. **Security and privacy:** *How can security and privacy aspects of eHealth services be maintained without compromising usability and performance?*

1.4 Methodology

Friedman and Wyatt [47] described nine important evaluation study types in biomedical informatics. In this research project, four of the nine study types were carried out, namely needs assessment, design validation, structure validation, and laboratory function study. Other study types such as usability testing, field-function study, and laboratory-user effect study of some parts of this work are planned to be conducted in different projects as continuation of this research project.

The author started the research work by conducting a scientific literature study and review on emerging technologies used in the smart home and eHealth domains to pinpoint potential uncovered areas that lead to the development of new solutions. An architectural design of the proposed solutions was then carried out to overcome the limitations found in the state-of-the-art approaches, followed by proof-of-concept implementations aiming to show the feasibility of the proposed solutions. Some aspects of the developed prototypes were evaluated in terms of performance, availability, and scalability, but due to time constraint of the PhD project, field trials are left aside for the continuation of the research work.

1.5 Limitation of Scope

This thesis focuses primarily on technical aspects of devices and services integration in the context of delivering flexible and efficient healthcare services to inhabitants in general and to patients in particular living at their homes. Medical and organisational aspects of the heathcare services are not covered in the thesis, and field trials of some parts of the proposed and developed solutions with real users (i.e. patients) are planned to be conducted in the continuation of the research work mainly due to time limitation of the PhD project. It is hoped that conceptual ideas and some proof-of-concept implementations as the main outcome of this research work will benefit further technology development in the area.

1.6 Structure of the Thesis

This thesis is organised as a compilation of scientific papers. It consists of two parts, where the purpose of the first part is to provide an overview of the research work, while the second part presents selected scientific articles rewritten as close to the original versions as possible in a chronological order throughout three years of research work.

Part I

Part I of this thesis provides an overview and summary of the thesis. Chapter 1 consists of the background and motivation of the research work alongside several key challenges and research questions which set the stage for this thesis. Chapter 2 describes the state-of-the-art concepts and technologies that lay the foundation of this work, including related works that have been conducted earlier by other researchers. Chapter 3 presents the proposed solutions which try to solve the issues addressed in chapter 1. Chapter 4 contains discussion on the results achieved throughout the research work, including an evaluation of the research questions presented in chapter 1. Chapter 5 presents a summary of the major contributions of this work with concluding remarks on future research directions.

Part II

Part II of this thesis consists of five selected peer-reviewed research articles (three conference proceedings articles and two journal articles) that have been published.

Three more published articles are listed but not included in Part II. The author of this thesis is the main author in all articles.

Paper I presents an idea of an integration platform that enables service developers to combine the capabilities of different devices to support the inhabitants' wellness and daily activities following a Service-Oriented Architecture (SOA) paradigm. Social media is proposed to be integrated in these services to enable collaborations with relatives, colleagues, and healthcare specialists in monitoring and encouraging a better lifestyle to the inhabitants, as well as to support remote administration of the developed and deployed services.

The integration platform is further extended in paper II to incorporate context-awareness by utilising an ontology-based model. A smart home ontology is presented in the paper following a Web Ontology Language (OWL) standard representation, and a Semantic Web Rule Language (SWRL) is used for reasoning on the ontology. Several usage scenarios are presented in the paper.

A more detailed layer-by-layer description of the proposed integration architecture is presented in paper III, and location-awareness to support the inhabitants' and the patients' mobility outside their homes is presented as well. A developed remote health monitoring prototype that utilises an open source Enterprise Service Bus (ESB) is presented in the paper as a service example of the proposed integration architecture.

Paper IV describes an Information Integration Platform (IIP) that enables everyday objects to communicate with Internet-based services following the Internet of Things (IoT) vision. It makes use of a publish/subscribe messaging pattern, and its functionalities are exposed through RESTful Web services. A prototype implementation of the platform is presented in the paper alongside its main features. Results of experimental comparisons between a brokered approach (via the implemented IIP) and a direct point-to-point approach are presented as well.

Additional functionalities of the IIP for security and access management to information are described in paper V. To support high availability and scalability, a clustering technology with load balancing is incorporated in the IIP, and is explained in the paper. Conducted performance evaluation results of both the clustered IIP version and its standalone counterpart are presented as well.

Chapter 2

State-of-the-Art

This chapter describes several non-exhaustive related works that have been carried out by other researchers and engineers, as well as the most current technologies and paradigms that enable the author to proceed with the design and development of the proposed solutions in the next chapter. The main aim of this chapter is to give the readers an overview of the concepts that act as the building blocks of the main contributions of this thesis.

2.1 Ambient Assisted Living, Telehealth and Telecare

Ambient Assisted Living (AAL) [48, 49, 50, 51, 52] is a term that refers to all kinds of technology solutions that aim to assist people with special needs in their daily activities and enable independent living at their homes. These solutions are based on intelligent environments surrounding the inhabitants, often unnoticed or being almost "invisible", that can adapt autonomously to the inhabitants' contexts and help them perform necessary actions to accomplish tasks that they wish to carry out. According to the European Ambient Assisted Living Innovation Alliance (AALIANCE) [53], technologies that are used for building AAL systems are generally expected to be:

- embedded (non-invasive, invisible devices);
- distributed in the physical environment or directly integrated into appliances;
- personalised, talored to the users' needs;
- adaptive (responsive to the users and the environment);

- anticipatory, able to anticipate users' desires as far as possible without conscious mediation.

AAL requires computing resources to be available in the users' surroundings instead of just single piece of hardware, connected with one another, and together cooperate assisting the users in their daily activities. This concept roots back to the notion of *ubiquitous computing* [54, 55, 56], where computers are embedded in everyday objects that human beings use or interact with. Several enabling technologies are fundamental to be present as the stepping stones for developing AAL systems, described as follows [53].

- *Sensing*: anything and anywhere, in-body or on-body, in appliances, and in the environment.
- *Reasoning*: collecting, aggregating, processing and analysing data, transforming them into knowledge. Reasoning engines could be implemented in sensors, home appliances, or servers connected to a network.
- *Acting*: automatic control through actuators and feedback (e.g. information, suggestions, guidance), which can be local or remote, instantaneous (e.g. emergency situation) or delayed (e.g. trend information, lifestyle recommendations), to relevant users utilising personalised multi-modal interfaces.
- *Communications*: sensors and actuators are connected to reasoning systems that may be connected to other reasoning systems with additional actuators.
- *Interaction*: intelligent interaction between users and systems, which is a very important aspect of AAL services will have specific requirements to adapt to the users' abilities.

AAL can be utilised to support healthcare service delivery by healthcare providers (e.g. hospitals) and government bodies (e.g. municipalities) to patients living at their homes, for example by means of telehealth. The term "telehealth" evolved from other term called "telemedicine", which is described as the use of audio, video and other telecommunications and electronic information processing technologies for the transmission of information relevant to the diagnosis and treatment of medical conditions at distant sites [14]. Telehealth tends to have a broader scope towards preventative, promotive, and curative aspects as opposed to telemedicine that solely focuses on the curative aspect in a regulated healthcare setting. Telehealth in general can be realised in two different modes: real-time and store-and-forward [57]. Real-time telehealth systems require instantaneous interactions between distantly located patients and healthcare workers. High data rate is often needed to deploy this type of telehealth systems. On the other hand, store-and-forward telehealth systems store the captured data in the patients' devices, then forward them at a later time to back-end servers for offline assessment by responsible healthcare personnel. "Telecare" is another important term which is commonly defined as the use of ICT to provide care directly to the users, especially elderly people [58]. Although the term is often used interchangeably with telehealth and telemedicine, telecare focuses more on the care aspect, and normally excludes the exchange of information solely between professionals for diagnosis or referral [59]. Typical functions of telecare systems include safety and security monitoring in the home, information and support provided via telephone and the Internet [59, 33]. The application of telehealth and telecare in home environments is widely known as "telehomecare". By utilising telehomecare, virtual home visit is possible to be carried out by healthcare personnel by means of interactive audio-visual communications as well as less-complex, non-interactive technology via the Internet, modem, or telephone. This may involve, for example, physical assessment of the patients' hearts, lungs, and obtaining vital signs such as blood pressures and pulse rates.

Polisena et al. [60] conducted a systematic literature review of the application of home telehealth for Chronic Obstructive Pulmonary Disease (COPD) compared with traditional care, and found out that home telehealth reduces the rates of hospitalisation and Emergency Department (ED) visits. However, the mortality rate was higher among patients with COPD using home telehealth, but the number of original studies were few and sample sizes were relatively small. Smith et al. [61] used 2-way interactive video technology to monitor medication compliance of several persons with mild dementia living at their homes, and discovered that it was feasible to be applied to avoid the need for home healthcare visits. The authors argue that video and simple phone monitoring help stabilise medication compliance in mild dementia patients who live alone and may be at risk for premature relocation out of their homes. A systematic review to identify studies on the effect of home telehealth on clinical care outcomes was conducted in [62]. The meta-analysis indicates that telehealth positively affects clinical outcomes, with effect sizes ranging from mild to moderate, which suggests that telehealth may be useful for conditions that require close monitoring, clinical assessment, and early intervention to avoid adverse events such as emergency visits and hospitalisation. Walsh and Coleman [63] initiated a pilot project of telehealth programme which was aimed to improve patient outcomes by augmenting patients' regularly scheduled in-home skilled nursing visits with video-conferencing encounters. In general, the authors have demonstrated consistent positive outcomes for all patient participants, which offer insights to the potential of telehealth technology as an effective tool to lower costs for the home care industry. They concluded that telehealth can help extend the services that have already been provided by the home health nurse and improve care, which means it can help home care agencies to do more with less.

From the technical perspective, Nourizadeh et al. [64] developed a patient-oriented telehomecare system to allow elderly to be medically monitored and assisted at their homes, that consists of five elements in its architectural design: Medical sensors and/or Body Sensor Networks (BSN), environmental sensors and home automation sensors network, a gateway, Web services technology, and Graphical User Interfaces (GUIs). The system is accessible from anywhere, displaying an adaptive remote sensing service combined with interactive telecommunications and an automated alert system. The authors in [65] introduced non-intrusive and non-invasive monitoring and assistance to the elderly directly at their homes. A general framework was defined in this work, and a semantic model called Smart Home Ontology Model (SHOM) was proposed to perform autonomic decision-making in the U-Health smart home. Pau et al. [66] proposed a design of personal health system to be integrated with smart home services platform supporting home-based e-care by using a common media server. As the interactions between people and technology is of specific importance in home eHealth applications, the system design takes into consideration the human factor guideline from European Telecommunications Standards Institute (ETSI). Another telehealth system that is focused on monitoring patients at home was proposed in [67]. The system was designed based on the AILISA project [68], employing off-line communications model by utilising emails for data exchange. The authors argued that data exchange using emails has numerous advantages, among others, email is widespread over networks and can work off-line. As email addresses are independent of machines (i.e. IP addresses), dynamic IP nodes are reachable, including nodes placed inside Virtual Private Networks (VPNs). A design of wireless body sensor system for monitoring the patients' physical activities in their homes is described in [69], where accelerometers are used for demonstration purpose to measure different types of human movements. There was wireless interference from nearby IEEE 802.11 signals and microwave ovens when IEEE 802.15.4 devices were used during the experiment conducted by the authors. Data delivery was, nevertheless, satisfactory and could be improved by selecting appropriate channels. The authors also found out that housing materials, home appliances, and even plants could attenuate the wireless signal at different scales.

Despite their positive traits, telehealth and telecare projects often focus on technologies to support specific diseases or social care problems which can result in fragmentation and information silos that impede integrated care of elderly people and patients [33, 34, 35, 36, 37]. For example, if clinical data (vital signs, assessments, medications, allergies) are captured in a telehealth or telecare system, but not integrated with the patient record in the General Practitioner (GP) or hospital system (or vice versa), then drug or treatment contra-indications could be missed [34].

2.2 Internet of Things

The term "Internet of Things" (IoT) was popularised at the MIT Auto-ID Center in 1999 where a group of people started to design and propagate a cross-company RFID infrastructure [24, 25]. The advancements in ICT have enabled IoT's vision to be realised by turning everyday objects to connected objects [26, 27], so that information gathered by these objects can be utilised in various different services. Everyday objects can be employed to capture and create information from the physical world instead of relying purely on people as normally done in traditional information systems [24]. This is mainly achieved by utilising RFID and sensor technologies. The ability to react automatically to events in the physical world opens up new opportunities to deal with complex or critical situations, as well as enables a wide variety of business processes to be optimised. The real-time interpretation of data from the physical world can lead to the introduction of various novel services and may deliver substantial economic and social benefits [27]. Figure 2.1 illustrates the technology roadmap for the IoT according to SRI Consulting-Business Intelligence (now Strategic Business Insights[1]).

The IoT is not made up of one single novel technology. Instead, it comprises of several complementary technologies that provide the capabilities to bridge the gap between virtual and physical spaces. These capabilities include, among others [27]:

- *Communications and cooperation.* Wireless technologies such as the GSM and UMTS, WiFi, Bluetooth, ZigBee, and other wireless networking standards currently under development enable objects to communicate with other objects and Internet resources.

[1] http://www.strategicbusinessinsights.com/

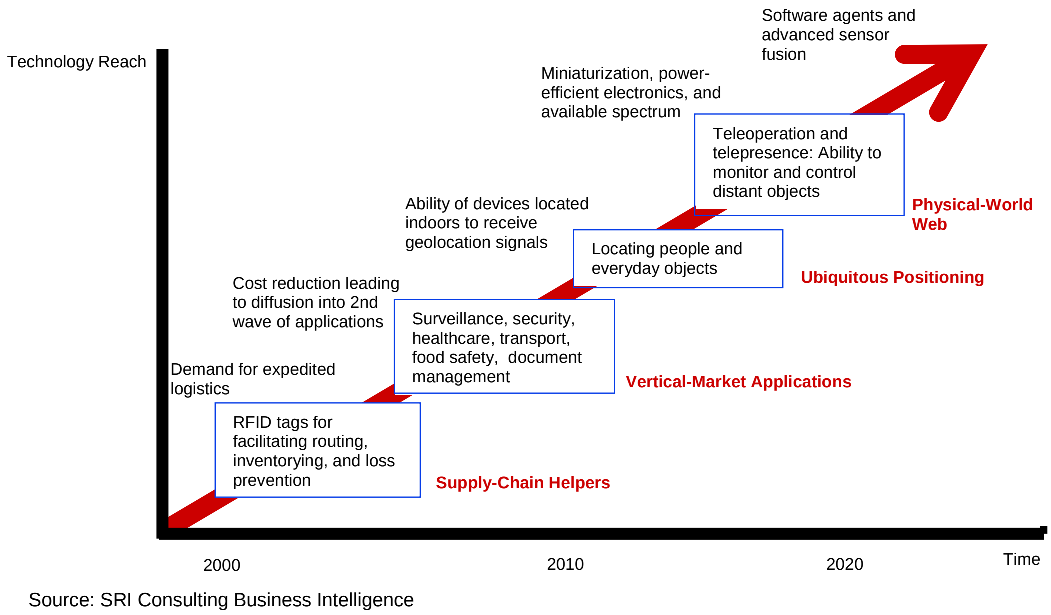


Figure 2.1: Technology roadmap of the Internet of Things

- *Addressability*. In IoT, objects can be located and addressed via discovery, look-up or naming services, so that they can be remotely interrogated or configured.

- *Identification*. RFID, NFC, and optically readable bar codes are examples of technologies that can be used to uniquely identify objects. Identification enables objects to be linked to information associated with those objects.

- *Actuation*. Objects may contain actuators that can be used to manipulate their surrounding environments. Such actuators can be utilised to remotely control real-world processes via the Internet.

- *Embedded information processing*. Objects that are equipped with processor or microcontroller and memory for storage can be used, for example, to interpret captured information.

- *Localisation*. GPS, GSM triangulation, radio beacons, ultrasound measurements, and optical technologies are some examples of technologies that can be used to locate physical objects.

- *User interfaces*. Innovative interaction paradigms such as tangible user interfaces, flexible displays, voice, image, and gesture recognition can be utilised to ease communications barrier between smart objects and their users.

2.3 Service-Oriented Architecture

The Service-Oriented Architecture (SOA) [38, 39, 40, 41, 42] aims to achieve loose coupling between interacting software components in a distributed environment. A service is a self-contained unit of functionality that is well-defined, can be discovered and composed, and does not depend on the context or state of other services [43, 70]. Services are described in a standard definition language, have a published interface, and communicate with one another requesting execution of their operations in order to collectively support a common business task or process [71]. As a result, each service is built as a discrete piece of code that is possible to be reused in different ways throughout the application by changing only the way an individual service interoperates with other services that make up the application. Figure 2.2 shows the basic components of a traditional SOA (SOA triangle), which consists of three main entities, namely *service provider*, *service consumer*, and *service registry* [40, 39, 72].

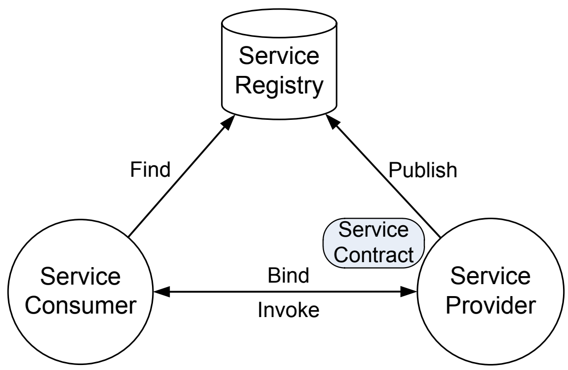


Figure 2.2: Basic components of a traditional SOA (the SOA triangle)

Services are implemented by a *service provider*, and their descriptions are published to a *service registry*. The *service registry* acts like yellow pages in the telecommunications domain, organises information about services and provides facilities for the *service provider* to publish descriptions of their implemented services as well as for the *service consumer* to discover available services that can be used. The *service consumer* queries the *service registry* to find a specific service, and if found, it retrieves the location of the service and binds to the service endpoint, then invokes the operations of the service. However, the *publish-find-bind-invoke* cycle, as shown in Figure 2.2, is not commonly applied in software systems [72], and the *service registry* is often being left as optional, leaving only the *service provider* and the *service consumer* interacting with each other in most cases.

Whether the *service registry* is used or not, the traditional SOA paradigm, as depicted in Figure 2.2, adopts a point-to-point integration model between the *service provider* and the *service consumer*. Although this approach is straightforward and simple to be deployed, an increasing number of services being involved can potentially create management and integration issues, as it introduces a tight-coupling between the sender and the receiver of the messages being exchanged which requires harmonisation in transport protocols, document formats, interaction styles etc. [23, 73]. To tackle the increasing complexity of integrating point-to-point services, a centralised service bus middleware is often utilised to avoid direct contacts between communicating services, which removes the hardwiring between the *service provider* and the *service consumer*. Figure 2.3 shows the general comparison of the two SOA approaches [23].

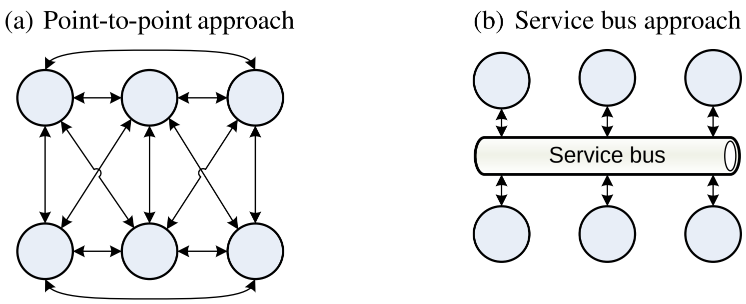


Figure 2.3: Two SOA approaches

2.3.1 Enterprise Service Bus

An Event-Driven Architecture (EDA) [74, 75, 76] defines a methodology for designing and implementing applications and systems in which events are transmitted between decoupled software components and services. An event is a change in state that merits attention from systems which may trigger some operations in different services. The combination of the SOA paradigm and EDA lays the foundation for an emerging technology that unites various conventional distributed computing, middleware, Business Process Modelling (BPM) [77, 78, 79], and Enterprise Application Integration (EAI) [30, 31] technologies. It offers a unified backbone on top of which enterprise services can be advertised, composed, planned, executed, monitored, and decommissioned [39]. This backbone is usually referred to as Enterprise Service Bus (ESB) [80, 2]. It acts as a loosely coupled, event-driven SOA with a highly distributed universe of named routing destinations across a multi-protocol message bus. Unlike the traditional point-to-point SOA approach, which could not avoid tight-coupling between the *service provider* and the *service consumer*, ESB provides a centralised approach for integration tasks, avoiding direct message exchanges between interacting services. Applications are abstractly decoupled from each other, and connected together through ESB as logical endpoints that are exposed as event-driven services. It is designed to provide interoperability between large-grained applications and other components via standard-based adapters and interfaces. The bus functions as both transport and transformation facilitator to allow distribution of services over disparate computing environments. In general, ESB has four major functions: message routing, message transformation, protocol mediation, and event handling [80]. Figure 2.4 illustrates the general architecture of ESB.

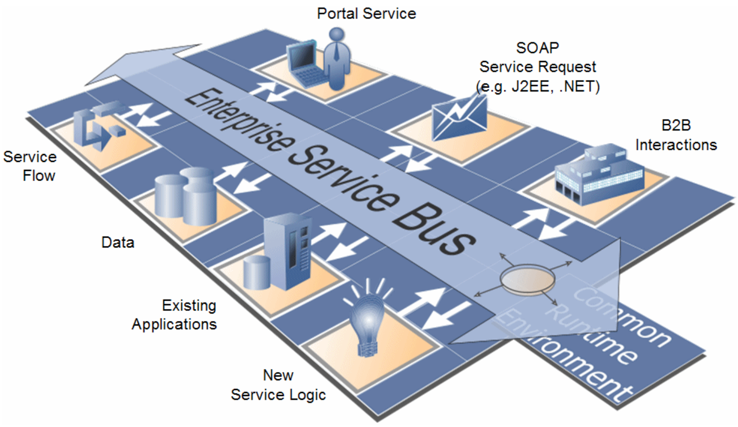


Figure 2.4: Enterprise Service Bus (ESB) general architecture [2]

There are several open source ESB implementations, and MuleESB[2] is one of the widely used products with a solid track record in organisations that have deployed it into production. It is a lightweight integration framework based on Java, and supports quite a number of transport protocols and connectors. XML-based configurations are used for creating message flows, which can be configured manually or through a friendly Integrated Development Environment (IDE) called MuleStudio. However, the learning curve is quite steep to understand the system thoroughly, and every modification requires a redeployment.

In Norway, a commercial product called the Shepherd[3] platform is used, nationwide, as an integration platform to interconnect sensors and applications within the healthcare sector. It exposes all of its services through RESTful APIs that are straightforward and simple to use. However, it supports only a handful of predefined observation types, where additional types can only be supported through custom response handlers that should be requested to the vendor. Applications can

[2]http://www.mulesoft.org/
[3]http://telenorobjects.com/shepherd/

only subscribe to devices instead of specific observation types for notifications. In addition, there is no permission granting scheme for an application to access observations that belong to other applications (i.e. cross-credentials access).

2.3.2 Web Services

SOA has been discussed quite frequently within the last decade, and the fundamental concept behind it has been adopted very rapidly in building complex software systems until recently, especially using Web services technology [43]. Web services technology promotes interoperability between various software applications running on disparate platforms by employing open standards and protocols. In addition, it supports the reuse of implemented services which further increases the speed of new service creation and deployment. Web services technology is currently divided into two major categories: the traditional "big" WS-* Web services [70, 81, 82, 3] and RESTful Web services [83, 82, 3]. Figure 2.5 shows the traditional "big" WS-* Web services and RESTful Web services in the context of application integration styles [3].

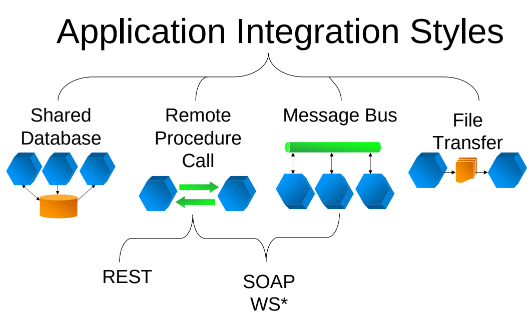


Figure 2.5: Traditional "big" WS-* vs. REST Web services in the context of application integration styles [3]

The traditional "big" WS-* Web services technology stack utilises Simple Object Access Protocol (SOAP) for exchanging structured information based on XML, Web Services Description Language (WSDL) for defining interfaces syntactically based on XML, and optionally Universal Description, Discovery and Integration (UDDI) for XML-based service registry. It is an XML-centric realisation of SOA, hence it is also known as XML Web services technology. This type of Web services provides interoperability for both Remote Procedure Call (RPC) and messaging integration styles [84]. One of its advantages is protocol transparency and independence, which enables the same SOAP message to be transported across a variety of middleware systems, which may rely on HTTP or other transports. This makes QoS aspects, such as encryption and reliable transfer, independent from the transports used along the path (i.e. end-to-end QoS) [3].

Representational State Transfer (REST) defines a set of architectural principles to design and develop Web services that focus on a system's resources. It ignores the details of component implementation and protocol syntax in order to focus on the roles of components, the constraint of interaction with other components, and their interpretation of significant data elements [85]. RESTful Web services technology is gaining increased attention mainly due to its simplicity for publishing and consuming a service [86]. Although REST was initially designed to be protocol-agnostic, its widespread use in the Web 2.0 domain has made HTTP as its de facto application-layer protocol. REST utilises HTTP methods explicitly with a one-to-one mapping between Create, Read, Update, and Delete (CRUD) operations and HTTP methods (i.e. POST, GET, PUT, DELETE). Every interaction with a resource is stateless (i.e. request messages are self-contained), and all session state is held in the client. Alternatively, session state can be transferred to another service such as the database. Resources in RESTful Web services are identified by Uniform Resource Identifiers (URIs), which provide a global addressing space for resource and service discovery. Resources are decoupled from their representations so that their contents can be retrieved in different formats (e.g. XML, JSON) [3].

2.4 Mobile Cloud Computing

Mobile applications targeted for mobile devices have become abundant in recent years due to the fast growth of mobile users. Figure 2.6 shows the growth of mobile subscribers worldwide according to GSMA[4]. These applications encompass a wide range of content categories, including entertainment, health, games, business, social networking, travel, and news. Digital distribution platforms for mobile applications such as the Apple App Store and the Google Play have made it even easier for mobile users to browse, search, download, and purchase applications of their interests and needs. One of the main drivers behind this trend is the ability of mobile computing to provide tools to the users at anytime, anyplace, irrespective of the users' movements, hence supporting location independence [87]. However, mobility faces some challenging issues such as limited resources, finite energy, and low connectivity, which may impede the execution of programs built for assisting the users and the creation of a pervasive environment [88, 89, 90]. Some mobile applications, such as location-based social networking, process captured data from

[4] http://www.gsma.com/

a mobile device's sensors. However, an extensive use of sensors in mobile devices is expensive in terms of energy consumption. In addition, high processing demands by certain applications, such as image processing, speech synthesis, augmented reality, and wearable computing, limit developers in implementing them for mobile devices. On the bright side, researchers have addressed this issue and have tapped into cloud computing technology to solve it.

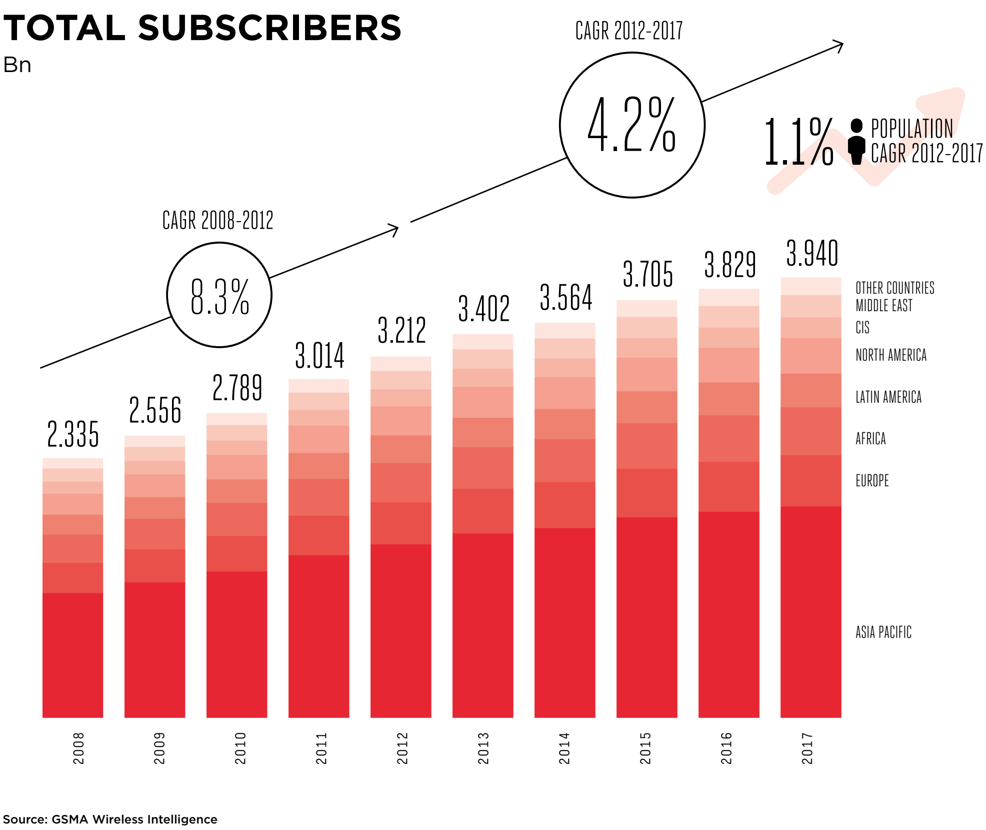


Figure 2.6: Total mobile subscribers worldwide [4]

Cloud computing is a paradigm in which computing resources, such as processing, memory, and storage, are not physically present at the user's location. Instead, a service provider owns and manages the physical resources, and the user accesses them through a network (e.g. the Internet) [91]. This type of computing provides many advantages for businesses, including low initial capital investment, faster deployment of new services, lower operation and maintenance costs, higher utilisation through virtualisation, easier disaster recovery, and flexibility to increase or to decrease computational resources on demand. The integration of cloud computing technology with mobile devices that aims to make the mobile devices more resource-capable in terms of computational power, memory, storage, and energy, is termed as “mobile cloud computing” [92]. It is the outcome of an interdisciplinary approaches comprising mobile computing and cloud computing [93]. The “outsourcing” procedure of migrating resource-intensive computations from mobile devices to the resource-rich cloud or server (i.e. nearby infrastructure) is also known as *computation offloading* [92, 91]. This paradigm alleviates the aforementioned issues faced by mobile devices through enhancement of application performance, reduction in battery power consumption, and enablement to execute heavy processes. In addition, cloud storage can be utilised to overcome storage constraints of the mobile devices.

2.5 Context-Awareness

According to Dey [94], context is defined as any information that characterises an entitiy’s situation, where an entity can be a person, place, or any object that plays a role in any type of interaction. Almost similarly, Schilit and Theimer [95] described context as location, identities of nearby people and objects, and changes to those objects over time. The word itself, derived from Latin *con* (with or together) and *texere* (to weave), describes context as an active process dealing with the way humans weave their experience within their whole environment, to give it meaning [96]. Although the definition of context within computer science communities rooted in user location, context encompasses many other things of interest including physical surrounding environments (e.g. lighting, noise level, temperature), network connectivity, communications costs, communications bandwidth, and social situation.

In computer science, context-awareness originated as a term from *ubiquitous computing* that intends to deal with linking changes in the environment with computer systems, so that computers sense and react based on their environments. Computation occurs in various situations and locations in contrast to the desktop computing style where computation takes place at a single location in a single context. Context-aware system is defined in [94] as a system that makes use of context to provide relevant information and/or services to the user. It aims at automatically personalising the user’s environment depending on the user’s context, and thus minimising user interaction with the system and the invoked services [97]. It should be sensible to user necessities, personalised according to the user’s profile, requirements, and context. In general, context-awareness can be considered as a prerequisite for adaptivity [98].

How contextual information is utilised by context-aware systems depends on the context model being used to represent the context. The formalism chosen for representing the context determines the reasoning methods the context-aware systems can use to perform adaptations. There are several context models that have been proposed and used for exchanging contextual information, and some of the most widely used are described as follows [99].

- *Key-value* model. This model is the simplest data structure for modelling contextual information. It is relatively easy to manage, but lack capabilities for sophisticated structuring and reasoning.

- *Markup scheme* model. This model makes use of a hierarchical data structure consisting of markup tags with attributes and contents. This type of context model is typically used in *profiles*, where a derivative of Standard General Markup Language (SGML), such as the XML, is commonly utilised.

- *Graphical* model. Unified Modelling Language (UML) is a widely used modelling instrument that has a strong graphical component (i.e. the UML diagrams). It can be utilised to model context as well.

- *Object-oriented* model. This model employs the main benefits from any object-oriented approach, encapsulating the details of context processing on an object level. Access to contextual information is provided through specified interfaces.

- *Logic-based* model. This model defines the context as facts, expressions, and rules. Contextual information is normally added to, updated in, and deleted from a logic-based system in terms of facts or inferred from the rules in the system respectively. This type of context model possesses a high degree of formality.

- *Ontology-based* model. This model makes use of ontologies to model contextual information as ontologies are a promising instrument to specify concepts and interrelations. Ontologies are particularly suitable to project parts of the information being used in daily life to a data structure utilisable by computers.

2.6 Chapter Summary

The main ideas behind AAL, telehealth, and telecare concepts were described in this chapter alongside several existing research and development works in the area. As the future trend of health and care services are moving towards the inhabitants' and the patients' homes which enable them to live independently through ICT-based assisted living technologies, the IoT vision comes into play, transforming everyday objects into connected objects to support such services. However, these services are traditionally designed in a vertical "silo" approach from the devices to the applications to serve specific diseases or social care problems. The SOA paradigm fits well to tackle this issue, enabling modular and reusable device capabilities to be utilised by different applications. Web services technology realises the SOA concept with a pragmatic approach through the use of open standards and protocols to enforce interoperability. ESB technology complements Web services to support event-driven communications as well as integration with other technologies. Mobile devices and wireless sensors are expected to play a big role in the future eHealth services as they support mobility for the users, enabling the inhabitants and the patients to freely move outside their homes. However, mobile devices, which are typically battery-powered, have limited operating lifetime as well as limited processing capabilities. The mobile cloud computing paradigm suits well to solve these issues by migrating resource-intensive computations from mobile devices to the cloud. In order to provide eHealth services, applications need to know the context of the users (e.g. location information) and perform reasoning processes upon it. Context-awareness is at the heart of such applications, and several context models were described in this chapter. The concepts and technologies presented in this chapter act as the stepping stones for the proposed solutions that will be presented in the next chapter, aiming to fulfil the requirements listed in the previous chapter.

Chapter 3

Proposed Solutions

This chapter presents the proposed solutions to the issues addressed in the previous chapters as the main contributions of the author in this thesis to the scientific community.

3.1 Guiding Integration Architecture

Telehealth and telecare have been widely used in recent years to provide health and care services at a distance as it supports eliminating space barriers as well as alleviates the cost burden of healthcare services by moving non-urgent treatments from healthcare premises to patients' homes. By adopting telehealth and telecare, patients and inhabitants will have more freedom in living their lives with minimum interventions from healthcare personnel. This requires the homes to be smart enough to facilitate healthcare services to be deployed and to assist the patients and the inhabitants in their daily activities. To achieve this, sensors, actuators, and various devices are necessary to be installed in their homes, which then become smart homes. There are available standards for home networks that support home automation, although many of them are proprietary. Some examples of these standards are X10[1], KNX[2], Echonet[3], UPB[4], and INSTEON[5]. However, most healthcare services require health-related devices to be present in addition to home appliances that the home networks standards support. On the flip side, AAL services will not work properly without the involvement of home appliances. A common integration platform is therefore necessary to be present in a smart home environment to

[1] http://www.x10.com/
[2] http://www.knx.org/
[3] http://www.echonet.gr.jp/english/
[4] http://pulseworx.com/downloads/upb/UPBDescriptionv1.4.pdf
[5] http://www.insteon.com/

accommodate a unified access to data and functionalities of the various types of devices.

A SOA-based approach is chosen in this thesis to be the prime driving paradigm to tackle integration issues in a smart home environment for AAL, telehealth, and telecare services, as it supports flexibility and reusability of service components, as well as rapid development and deployment of new services. By utilising SOA principles, value added services beyond basic utilisation of each device or service can be provisioned. Figure 3.1, which is based on [23], illustrates the proposed SOA-based home integration architecture for the realisation of novel AAL, telehealth, and telecare services in a smart home environment. This architecture is mainly presented in paper III and VI.

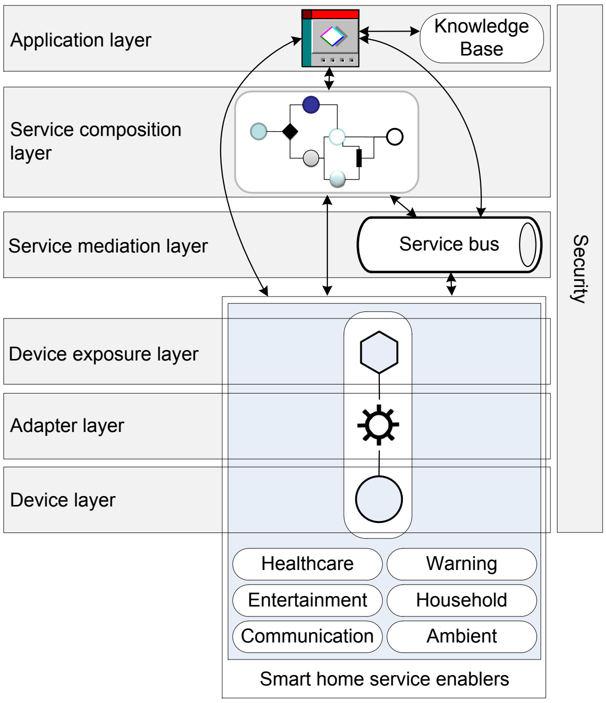


Figure 3.1: SOA-based home integration architecture

The proposed integration architecture follows a multi-layer design, consisting of seven main layers, described as follows [23, 100].

1. **Device layer**. This bottom-most layer consists of a multitude of devices in the smart home, including home appliances and health-related devices, which are used for gathering data (i.e. sensors) as well as to be controlled (i.e. actuators). Devices in this layer are not necessarily physical objects, but any data source that provides usable information as well as controllable resources

(i.e. virtual devices) that can be used by the services being deployed in the smart home.

2. **Adapter layer**. This layer is responsible for translating a service operation call from the adjacent upper layer to the device's native API. For incoming data from the lower layer, this layer is responsible for transforming raw data to the suitable format of the upper layers. The adapters can be implemented in the device itself if it is programmable, otherwise a separate box with listeners to incoming messages from the device should be utilised. If a mediation layer is present (e.g. a service bus), message translation functionalities can be performed by the mediation layer.

3. **Device exposure layer**. This layer is responsible for exposing capabilities of the devices as services, acting as the interface towards service consumers from a SOA perspective. This exposure may include service descriptions, such as the WSDL in traditional Web services, to ease the consumption process by the service consumers. If a mediation layer is used (e.g. a service bus), capabilities exposure of the devices can be performed by the mediation layer.

4. **Service mediation layer**. This layer is responsible for mediating communications between service providers and service consumers in a SOA environment, playing the role of a service broker. It decouples service providers from service consumers, avoiding direct point-to-point communications between them. This layer should have its own data storage for persisting incoming information from the devices to enable later retrieval by service consumers. Service bus technologies, such as the ESB, suit well in this layer.

5. **Service composition layer**. This layer provides the capability to combine existing services, either atomic or composite services, creating new value-added services. Service composition can be seen as in a part-of sense where a larger part encapsulates services and exposes itself as a service, or in a sequencing sense where an invocation order of existing services is defined. In traditional Web services, Web Services Business Process Execution Language (WS-BPEL) [101] is a widely used language for Web services composition following the part-of approach, and many ESB technologies support message flow for sequential invocation of services.

6. **Application layer**. This upper-most layer is responsible for hosting applications which are developed and deployed to provide services to the patients and

the inhabitants in the smart home, encapsulating different logics. The main intelligence in the smart home is envisaged to be residing in this layer by making use of the underlying basic and/or composite services, and therefore, a knowledge base, where all contextual information is stored, is expected to be present as well in this layer.

7. **Security layer**. Unlike the other layers, this layer is positioned vertically, spanning across all other layers, and is responsible for handling security issues. Different security mechanisms can be applied to the horizontal layers depending on the required level and type of security. Since devices' capabilities are expected to be published in terms of standardised APIs in this architecture (e.g. using Web services technology), messaging and transport security between service providers and service consumers will be one of the main focuses of concern.

The three bottom-most layers in this integration architecture form the smart home service enablers which provide the baseline capabilities of various devices to be utilised by more sophisticated applications in the application layer. In other words, they act as building blocks that enable rapid development and deployment of novel services. Six categories of service enablers are shown in Figure 3.1 that correspond to six different types of devices, namely healthcare, entertainment, communication, warning, household, and ambient. New categories may be added as new types of devices are introduced in the smart home. This architecture provides freedom to service designers and developers to either use the service enablers per se, combine their applications with composite services, or rely on the mediation layer's specific functionalities such as the mediation flow for sequencing the invocation of services.

The proposed integration architecture is aimed to guide system designers and developers to componentise their systems into smaller self-contained and reusable entities to avoid vertical "silo" systems that can only serve specific functionalities. The architecture itself can be realised and deployed in a closed smart home environment where all layers of the architecture are materialised inside the smart home's physical boundaries, including all services being deployed in home servers. It can also be realised in a hybrid manner, where some layers are implemented inside the smart home while some others are applied in trusted third party environments. A realistic example could be a hospital Electronic Health Record (EHR) application that gets updated as soon as its home patients perform daily measurements (e.g. blood pressure). In this case, the application layer is strecthed out to include applications deployed in hospitals, utilising the exposed data through the device exposure layer in the smart home.

3.2 Information Integration Platform (IIP)

Advancements of ICT in the healthcare sector have resulted in patients facing various portable wireless medical and fitness devices in their daily activities, which are being pushed to the consumer market by various vendors in recent years. This can be seen as a positive trend towards self-empowerment of healthier lifestyle, and enables healthcare personnel to more efficiently keep track their patients' health conditions via telehealth, if such feature is provided. Many vendors provide additional online services for devices they sell, enabling users to better visualise, store, and share the gathered information from the devices through the Internet. However, many of these services are integrated following a vertical "silo" approach (shown in Figure 3.2(a)), where the devices can only be used with the provided services, and other services have no or very limited possibilities to utilise such information, and thus it is quite common that similar information is redundantly gathered by different devices for their own services. The main disadvantage of this situation is the inability to combine information captured from different devices produced by different vendors for better reasoning and decision making [102, 103]. Open interfaces (e.g. Web services) are necessary to be provided by device vendors to overcome this issue, so that service developers can incorporate the information gathered from the devices in their services. However, the integration normally still follows a point-to-point approach, as shown in Figure 3.2(a), where each service is directly interacting with each device that provides the information. The downside of such point-to-point integration, from the devices' perspective, is that they have to send each newly gathered information to all services that are interested in using it. This is particularly an issue for battery-powered wireless devices.

As described earlier, the point-to-point SOA technique is simpler and more straightforward to be adopted compared to the service bus approach. However, the management of interactions between service providers and service consumers can be cumbersome when the number of services grows, and thus centralised service bus approach is more advantageous to be used. The integration architecture depicted in Figure 3.1 provides both mechanisms to be utilised, as there may be some scenarios where direct point-to-point interaction is more beneficial than using service bus, and vice versa. Nonetheless, a myriad of devices is expected to fill in the smart home environment to support the patients' and the inhabitants' well-being,

(a) Point-to-point vertical "silo" integration (b) Brokered converged integration

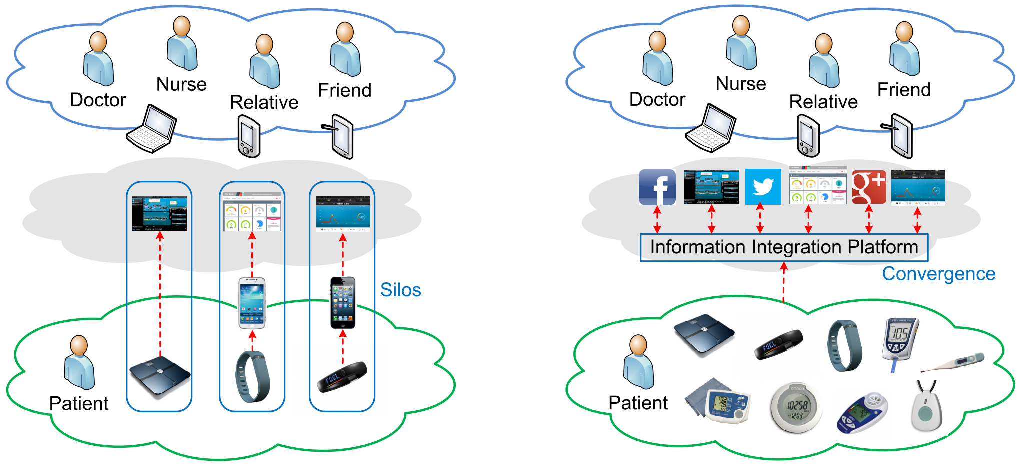


Figure 3.2: Point-to-point vertical "silo" vs. brokered converged integration approaches

and therefore, a service bus is essential to exist in the smart home.

The ESB concept that combines EDA and SOA paradigms fits well in the service mediation layer to simplify integration tasks, facilitating indirect interactions between service providers and service consumers in both synchronous and asynchronous manners. However, as it is not standardised, many ESB implementations provide various different sophisticated features in different ways, and many of these functionalities require steep learning curves for service developers to get used to. A service bus, called the Information Integration Platform (IIP) is proposed by the author with the primary aim to overcome this issue, offering a unified way of centralised brokering for message exchanges between service providers and service consumers. It utilises the publish/subscribe messaging pattern, where subscribers have the ability to express their interest in an event, and are subsequently notified of any event which is generated by a publisher and matches their registered interest [104]. The platform itself is designed to be flexible enough following the SOA paradigm to not only accept information from physical devices, but also from any type of information provider. It focuses on simplifying the information integration process while still follows the event-driven SOA concept as the ESB does, so that the platform can be used out-of-the-box without any internal programmatical modification. This is mainly achieved by enforcing a strong constraint that both service providers and service consumers should exchange messages through RESTful Web services. Gateways, which physically can range from dedicated servers to mobile devices (e.g. smartphones, tablets), are needed to encapsulate the captured information from the devices as HTTP requests to be sent to the platform in case the devices cannot send HTTP requests on their own. If gateways are used, they should have at least two network interfaces, one facing the devices and another one facing the IIP. Mobile gateways are particularly useful for scenarios that involve mobility of the patients [105]. The IIP is presented in detail in paper IV and V.

3.2.1 Conceptual Design

The IIP is proposed to bridge the communications between devices (i.e. information providers) and services (i.e. information consumers), acting as a service broker between the two entities. Information distribution task is delegated to the broker, so that information providers only need to send newly collected information once to the service broker. The service broker provides convergence for information gathering from various different devices that the patients encounter, as shown in Figure 3.2(b). RESTful Web service interfaces are proposed to be used facing both information providers and information consumers as they are widely used by cloud-based services, and adhere to SOA principles.

Information is shared through "information channels" that are managed internally by the IIP. Information channels belong to information providers, and each information channel contains a piece of information that can have several parameters. Information providers can publish new information to information channels that belong to them, and information consumers can subscribe to information channels containing information that they are interested to utilise. Information consumers will get notifications from the IIP whenever there is new information available in the information channels that they are subscribed to, following the standard publish/subscribe messaging pattern. An information provider can have many information channels, but an information channel can only be associated with one information provider (i.e. the owner). An information channel is attached to exactly one device, and a device can have many information channels. A device can only be associated with one information provider (i.e. the owner), but an information provider can have multiple devices. An information channel can be accessed/subscribed by many information consumers, and reciprocally, an information consumer can access/subscribe to many information channels. These relationships are shown in Figure 3.3.

In general, the IIP provides resources that are divided into two categories with two main base URIs, namely `https://root/provider/` for information providers (service providers from a SOA perspective), and `https://root/consumer/` for information consumers (service consumers from a SOA standpoint), where the `root` part of the URI refers to the domain of a specific IIP deployment. Following the

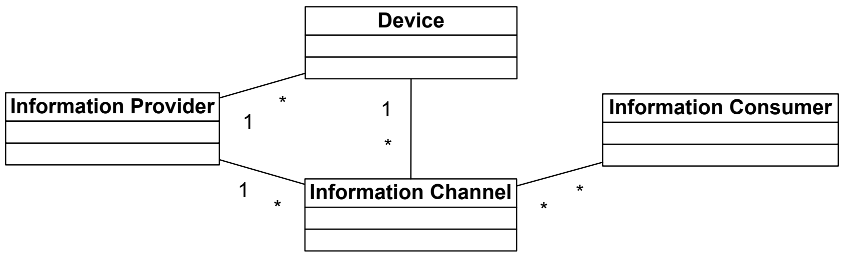


Figure 3.3: Relationships between information provider, device, information channel, and information consumer

REST approach, HTTP methods (i.e. POST, GET, PUT, DELETE) are utilised for exposing the services, where GET is used for nullipotent services, PUT and DELETE for idempotent services, and POST for non-idempotent services. Figure 3.4 shows the main functionalities ot the IIP.

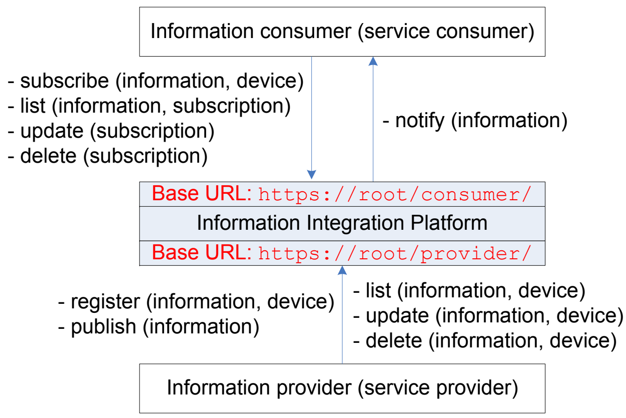


Figure 3.4: Main functionalities of the IIP

The main functionalities of the IIP with their REST resources for information providers are described as follows.

1. **Registration**. The IIP provides a resource with URI `https://root/provider/registration/information/` for information providers to register/add an information channel. This resource accepts HTTP POST request with Content-Type `application/x-www-form-urlencoded` and three predefined parameters, namely *deviceId*, *name*, and *description*. The *deviceId* parameter is a pointer to an existing device in the IIP that belongs to the information provider who sends the HTTP request. If this parameter is not specified in the request body, or specified but no such device exists in the IIP, or the device exists but belongs to other information provider, then a new device will be registered automatically. The *name* and *description* parameters are optional. The IIP will generate *infoId* as the identifier of the newly

registered information channel, *creationDate*, and *lastUpdateDate* for timestamping. In addition, the IIP will also associate the information provider's credentials with *ownerUsername* parameter, and adds it to *allowedUsers* parameter for access control. If other parameter names (apart from the mentioned ones) are included in the request body, they are treated as information parameters to store actual values of the registered information channel. For example, a location information can have two parameters, namely latitude and longitude. An XML representation of the registered information channel will be returned in the response body upon successful processing of a request to this resource.

The IIP also provides a resource for information providers to register a device which can contain different information channels that belong to the same information provider, accessible through URI `https://root/provider/registration/device/`. This resource accepts HTTP POST with Content-Type `application/x-www-form-urlencoded`, and two predefined optional parameters, namely *name* and *description*. The IIP will generate *deviceId* as the identifier of the registered device alongside *creationDate* and *lastupdateDate* for timestamping. An XML representation of the registered device will be returned in the HTTP response body.

2. **Listing**. The IIP provides a resource for information providers to retrieve all information channels that they have registered (i.e. belong to them) via HTTP GET request to URI `https://root/provider/registration/information/` and a resource to retrieve a specific information channel using its *infoId* via HTTP GET to URI `https://root/provider/registration/information/{infoId}/`. For retrieving all devices that belong to an information provider, the IIP provides a resource with URI `https://root/provider/registration/device/` that is accessible via HTTP GET request. A specific device can be retrieved using its *deviceId* at URI `https://root/provider/registration/device/{deviceId}/` through HTTP GET request. All resources in this category will return the representation of information channels and devices in XML format.

3. **Updating**. Two resources are provided by the IIP for updating a registered information channel and a device with URIs `https://root/provider/registration/information/{infoId}/` and `https://root/provider/registration/device/{deviceId}/`, respectively. Both resources accept HTTP PUT request with Content-Type `application/x-www-form-`

`urlencoded`, and require *infoId* and *deviceId*, respectively, as identifiers for updating. Accepted parameters are similar to the registration process for both information channel and device, and the *lastupdateDate* parameter will be updated automatically by the IIP. An XML representation of the updated information channel or device will be returned in the HTTP response body.

4. **Deletion**. The IIP provides a resource for deleting a specific information channel using its *infoId* with URI `https://root/provider/registration/information/{infoId}/`, and another resource for deleting a specific device using its *deviceId* with URI `https://root/provider/registration/device/{deviceId}/`. Both resources accept HTTP DELETE request for deletion, and 200 OK will be returned as response with XML content confirming that the deletion process has been successful. Deletion of an information channel will automatically delete all subscriptions to that particular information channel, while deletion of a device will result in automatic deletion of all information channels (including all subscriptions to them) attached to the device.

5. **Publication**. The IIP provides a resource for information providers to publish new information to their registered information channels via HTTP POST request to URI `https://root/provider/publication/{infoId}/`. This resource accepts Content-Type `application/x-www-form-urlencoded`, and the parameters in the request body can be a subset of information parameters of an information channel (i.e. partial information update). All subscribers to the information channel will get notification from the IIP via HTTP POST request with XML representation of the newly published information in its body. The same XML representation is returned to the publisher in the HTTP response body.

Information consumers can receive information that they are interested to use either via a pull mode, where they send requests to specific information channels in the IIP, or by a push mode, where they subscribe to selected information channels, and get notifications from the IIP whenever new information is published by information providers to the information channels they have subscribed to. The main functionalities of the IIP with their REST resources for information consumers are described as follows.

1. **Subscription**. The IIP provides a resource for information consumers to subscribe to a specific information channel that they have access to. This resource

accepts HTTP POST request with Content-Type `application/x-www-form-urlencoded` to URI `https://root/consumer/subscription/`. The request body can contain four predefined parameters, namely *infoId*, *notificationUrl*, *name*, and *description*. The *infoId* parameter is mandatory as it points to the information channel that the information consumer is interested to subscribe to. The *notificationUrl* parameter is also compulsory as it acts as the endpoint where the notifications should be sent to. The *name* and *description* parameters are optional. An XML representation of the subscription will be returned in the HTTP response body. The *infoId* parameter can be replaced by *deviceId* if the information consumer wants to subscribe to all information channels that are attached to a specific device.

2. **Listing**. The IIP provides a resource for information consumers to retrieve the latest information from all information channels that they have access to via HTTP GET request to URI `https://root/consumer/listing/information/`. This mechanism corresponds to the pull mode of information retrieval. Information consumers can also retrieve the latest information from a specific information channel using its *infoId* via HTTP GET request to URI `https://root/consumer/listing/information/{infoId}/`. A time-range information retrieval is also possible to be applied to this resource by using a query string with parameters *fromTime* and *toTime*. For example, if an information consumer wants to retrieve all information from a specific information channel between 23rd and 30th of October 2013, it can send an HTTP GET request to URI `https://root/consumer/listing/information/{infoId}?fromTime=2013-09-23T00:00:00&toTime=2013-09-30T23:59:00`. The parameters' values, which are the actual information being passed through information channels, will be returned in the HTTP response by the IIP in XML format.

   In addition, the IIP provides a resource for information consumers to list all of their subscriptions via HTTP GET request to URI `https://root/consumer/subscription/`. An information consumer can list a specific subscription using its *subscriptionId* at URI `https://root/consumer/subscription/{subscriptionId}/` and list all subscriptions to a specific information channel using its *infoId* at URI `https://root/consumer/subscription/{infoId}/`. The IIP will return a list of subscriptions in XML format in the HTTP response body.

3. **Updating**. A resource for updating a subscription to an information channel is provided by the IIP for information consumers via HTTP PUT request to URI `https://root/consumer/subscription/{subscriptionId}/` with Content-Type `application/x-www-form-urlencoded`. *subscriptionId* is mandatory to be included in the URI as the identifier of a specific subscription, and accepted parameters are similar to the subscription process, namely *infoId*, *notificationUrl*, *name*, and *description*. Partial parameters update is possible (i.e. only a subset of the accepted parameters are included in the request body), and the *lastupdateDate* parameter will be updated automatically by the IIP with the most current time. An XML representation of the updated subscription will be returned by the IIP in the HTTP response body.

4. **Deletion**. The IIP provides a resource for information consumers to delete/remove their subscriptions to information channels either by using *subscriptionId* as the identifier at URI `https://root/consumer/subscription/{subscriptionId}/`, by using *infoId* as the identifier at URI `https://root/consumer/subscription/{infoId}/`, or by using *deviceId* as the identifier at URI `https://root/consumer/subscription/{deviceId}/`. The resource accepts HTTP DELETE request, and 200 OK will be returned in the response with XML content confirming that the deletion process has been successful.

5. **Notification**. There is no specific REST resource for notification of newly published information to an information channel provided by the IIP. Instead, information consumers should provide endpoints (indicated by *notificationUrl* parameter in a subscription) for handling notifications sent by the IIP.

3.2.2 Security and Privacy

Information is exchanged between information providers and information consumers through the IIP, and thus communications between them should be secured. Since HTTP-based REST services are used by the IIP to expose its resources, the communications security relies heavily on the HTTP protocol's security. In contrast to the traditional Web services technology that has a solid standardised security mechanism such as WS-Security [106], RESTful Web services do not have any standardised security. Transport Layer Security (TLS), in the form of HTTPS, has been the main security measure for message exchange in RESTful Web services, which provides a secure point-to-point communications channel on top of the transport layer.

HTTP Basic Access Authentication scheme provides an authentication mechanism to access resources on a Web server by means of username and password. This scheme is not considered secure as the username and password are transmitted through the network in plain text. Nonetheless, the combination of HTTPS and HTTP Basic authentication in many cases is enough for securing resources on a Web server as everything being sent through the wire is encrypted.

From the IIP's standpoint, information providers are applications that relay information from devices that the patients use to information channels in the IIP. Similarly, information consumers are applications that consume information in the IIP either through the publish/subscribe mechanism or the on-demand pull approach. As depicted in Figure 3.3, information providers own information channels, and thus access control to information channels should be provided by the IIP to avoid unauthorised disclosure of information, and information providers should be given full control of which information consumers can access their information channels. Identity-based access control is proposed to be used with an access control matrix that has to be maintained by the IIP. Only the owner of an information channel can publish information to it (i.e. write access to an information channel), and the access control matrix is utilised only for authorising information consumers to access/subscribe to information channels (i.e. read access to information channels). Table 3.1 shows an example of the proposed access control matrix maintained by the IIP. Security and privacy aspects of the IIP are presented in paper V.

	channel_1	**channel_2**	...	**channel_n**
username_1	YES	YES		NO
username_2	NO	YES		YES
⋮				
username_m	NO	NO		YES

Table 3.1: Access control matrix for read access to information channels

The rows in Table 3.1 represent capabilities of information consumers, while the columns represent access control lists of information channels.

In addition to the main functionalities as illustrated in Figure 3.4, the IIP provides resources for managing access control as shown in Figure 3.5.

The additional functionalities of the IIP that are related to access control with their REST resources for information providers are described as follows.

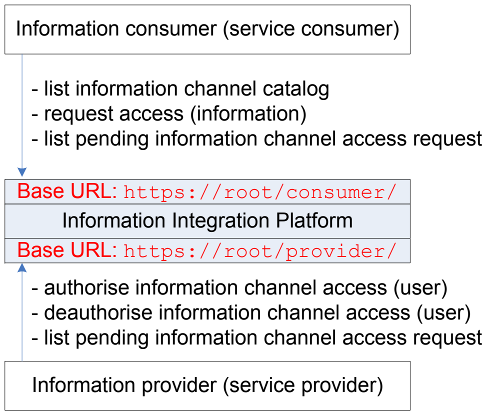


Figure 3.5: Additional functionalities of the IIP for access control

1. **Authorising information channel access**. The IIP provides a resource for information providers to grant an information consumer access to an information channel that they own. This resource accepts HTTP POST request with Content-Type `application/x-www-form-urlencoded` to URI `https://root/provider/authorization/{infoId}/`, where *infoId* is the identifier of the information channel of which an information consumer is to be granted access to. One predefined parameter is mandatory to be included in the request body, namely *username*, which refers to the information consumer user that is to be given access to the information channel. A 200 OK response will be returned by the IIP with an XML representation of the information channel in its body.

2. **Deauthorising information channel access**. Information providers can deauthorise a previously authorised information consumer to access an information channel they own via HTTP PUT request with Content-Type `application/x-www-form-urlencoded` to URI `https://root/provider/authorization/{infoId}/`, where *infoId* is the identifier of the information channel of which an information consumer's access right is to be removed. One predefined parameter is required to be included in the request body, namely *username*, which refers to the information consumer user. A 200 OK response will be returned by the IIP with an XML representation of the information channel in its body.

3. **Listing pending information channel access request**. The IIP provides a resource for information providers to list pending access requests to information channels they own via HTTP GET to URI `https://root/provider/authorization/`. An XML representation of a list of *infoId*s (i.e. the re-

quested information channels to be accessed) and information consumers' usernames (i.e. the requestors) will be returned by the IIP.

IIP's additional functionalities which relate to access control alongside their REST resources for information consumers are described as follows.

1. **Listing information channel catalog**. A resource for information consumers to list all available information channels in the IIP (i.e. information channel catalog service) is provided by the IIP at URI `https://root/consumer/catalog/`. It accepts HTTP GET request, and will return an XML representation of a list of available information channels that are anonymised (i.e. the owners' identities are not revealed).

2. **Requesting information channel access**. The IIP provides a resource for information consumers to request read-only access to an information channel via HTTP POST with Content-Type `application/x-www-form-urlencoded` to URI `https://root/consumer/authorization/{infoId}/`, where *infoId* is the unique identifier of the information channel. No parameter is needed for this resource. The IIP will return an XML representation of the access request in the response body.

3. **Listing pending information channel access request**. Information consumers can list their pending (i.e. not yet authorised) information channel access requests via HTTP GET to URI `https://root/consumer/authorization/`. This resource will return an XML representation of all pending access requests of the requestor (i.e. the information consumer) in the HTTP response body.

3.2.3 High Availability and Scalability

Although the IIP offers information distribution convergence between various different devices and services (as illustrated in Figure 3.2(b)), such centralised service broker is architecture-wise a single point of failure since both information providers and information consumers interact with and rely on it. High availability becomes a crucial factor for smooth deployment of the IIP, ensuring services receive information from devices in a timely manner and continue to work properly (i.e. reliable). Redundancy is the key to high availability, where service components are duplicated in different nodes, so that if one service component fails, other similar component will take over its tasks. Normally, high availability can be achieved in either master/slave or master/master mode. In master/slave mode, a server instance (i.e. the master) is in charge of serving client requests while another server instance (i.e. the slave) is running idle. When the master instance fails, a monitoring entity (i.e. the manager) will handover the master's tasks to the slave instance. On the flip side, all server instances are treated as masters in master/master mode, providing services to client requests in parallel. High availability and scalability of the IIP are presented in paper V.

The IIP is proposed to utilise master/master mode for high availability with an additional load balancer as proxy for handling client requests, both from information providers and information consumers. They will see the IIP as a single service broker entity, but its components are redundantly distributed among different nodes. This approach allows scalability aspect to be incorporated as well, enabling new nodes to be added when the present serving nodes are reaching their peak (i.e. fully loaded) in handling incoming requests. The IIP should be flexible enough to scale horizontally by the addition of new commodity servers when the number of participating information providers and information consumers grows.

Taking into account high availability and scalability, a 3-layer system architecture is proposed to be adopted by the IIP for deployment as shown in Figure 3.6.

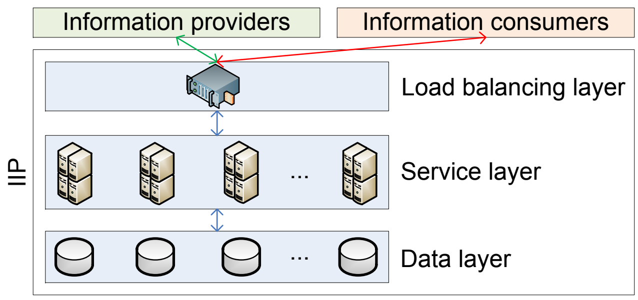


Figure 3.6: 3-layer system architecture for IIP deployment

The load balancing layer acts as a proxy service where both information providers and information consumers send requests to. The requests are then forwarded to one of the application servers in the service layer that hosts the main logic of the IIP (i.e. the IIP application) based on the load balancing criteria maintained by the load balancer. The IIP's functionalities (shown in Figure 3.4 and Figure 3.5) are realised in the service layer and are exposed through RESTful Web service interfaces. All information that needs to be stored is persisted in the data layer. By adopting this 3-layer architecture, the IIP's components can be made redundant, and can be scaled out according to deployment needs.

3.2.4 Prototype Implementation

The prototype of the IIP has been implemented both as a standalone server (i.e. all three layers in Figure 3.6 reside in one machine with no high availability and scalability features) and as a cluster of servers for high availability and scalability. The main application utilises Java Enterprise Edition (JEE) 6 in the service layer (as shown in Figure 3.6), deployed on Glassfish[6] 3.1.2.2 open source application server. Open source MySQL Community Server[7] 5.5 is used for the standalone prototype and open source MySQL Cluster[8] 7.3.2 is used for the clustered prototype to store data in the data layer. Java Persistence API (JPA) is utilised for mapping relational tables in the database to entity objects in the application through Java Database Connectivity (JDBC), and EclipseLink[9] 2.3.2 (JPA 2.0) is used as the JPA provider. HTTPS is employed for encrypting all message exchanges, and HTTP Basic authentication is used for authenticating user access to the exposed Web resources in the IIP. Figure 3.7 shows the prototype implementation architecture of the IIP.

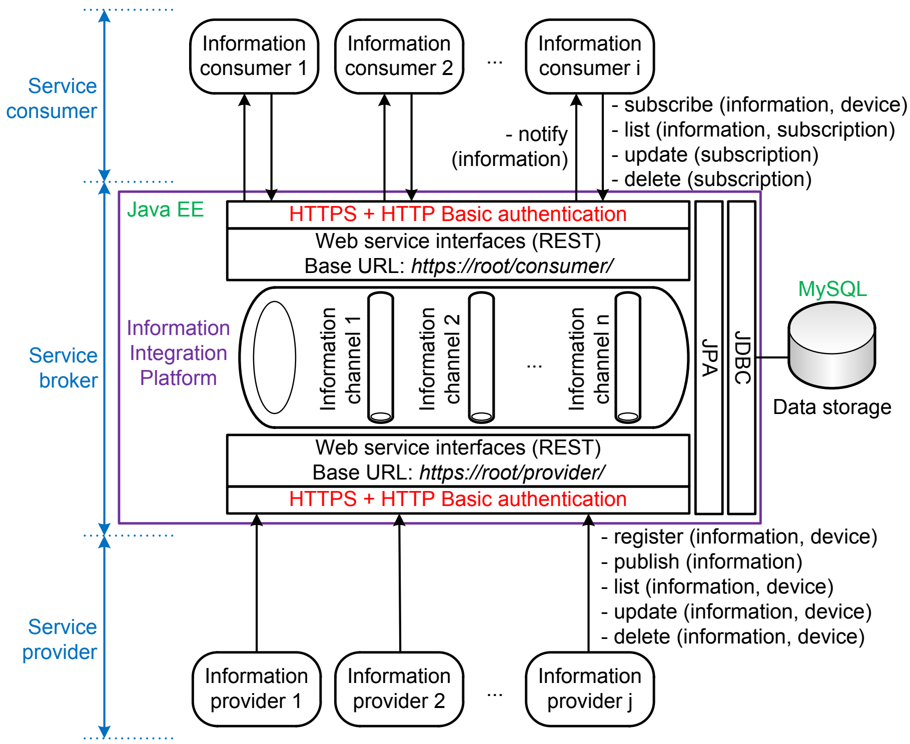


Figure 3.7: The prototype implementation architecture of the IIP

Information channels can be registered (created), listed (read), updated, and

[6]https://glassfish.java.net/
[7]http://dev.mysql.com/downloads/mysql/
[8]http://dev.mysql.com/downloads/cluster/
[9]http://www.eclipse.org/eclipselink/

deleted dynamically (i.e. on the fly) through the exposed REST interfaces without having to recompile and redeploy the IIP application. The access control matrix proposed earlier is simplified in the prototype to access control lists only (i.e. the columns in Table 3.1), and each information channel that belongs to an information provider maintains its own list (also termed as allowed user list). When an information provider authorises an information consumer to access one of its information channels, that particular information consumer's username is added to the information channel's allowed user list. Figure 3.8 depicts an example of information channel with its allowed user list represented in XML format.

```
<informationChannel>
    <infoId>info:782527135</infoId>
    <name>GPS</name>
    <description>GPS information channel</description>
    <deviceId>dev:125503922</deviceId>
    <creationDate>2013-05-05T21:37:12</creationDate>
    <lastupdateDate>2013-05-05T21:37:12</lastupdateDate>
    <ownerUsername>provider1</ownerUsername>
    <parameters>
      <latitude />
      <longitude />
    </parameters>
    <allowedUsers>
      <username>provider1</username>
      <username>consumer2</username>
      <username>consumer1</username>
    </allowedUsers>
</informationChannel>
```

Allowed user list

Figure 3.8: Example of an information channel represented in XML format

HTTP Basic authentication's credentials are used by client applications (both information providers and information consumers) to authenticate themselves when communicating with the IIP, and are directly used for the access control to information channels. Users in the IIP are categorised into three groups, namely *admin*, *provider*, and *consumer*. *Admin* users are mainly responsible for adding, removing, and assigning groups to users of the IIP. Users in the *provider* group (i.e. information providers) are by default added to the *consumer* group since information providers should be able to consume their own information (i.e. acting as information consumers). When an information provider registers (creates) a new information channel, its own username is automatically added to the information channel's allowed user list as the first information consumer that is allowed to access the information in the information channel. Users in the *consumer* group that are not included in the *provider* group are strictly information consuming-only users who cannot register (create) any information channel.

Application server clustering is supported by the Glassfish application server, and it can address the needs of both high availability and scalability. All application instances in a cluster, which can reside in different hosts, can be administered as a single unit, and user sessions can be automatically replicated between application instances. However, following the REST principles, services being provided by the IIP are stateless, and therefore, no client session from each request is maintained by the IIP. All server resources' states are persisted in the database. Thus, clustering at the service layer does not offer much advantage except simpler administration.

Open source Apache[10] Web server with `mod_jk` module is used for load balancing incoming requests from both information providers and information consumers, acting as a proxy. In the current prototype, the load balancing factor is set to be equal for all IIP application instances (hosted in different hosts) so that requests are forwarded equally (i.e. evenly distributed) among them. New application instances can be added in different hosts (scaled out) in the service layer, and load balanced through the proxy.

The IIP application instances in the service layer are responsible for handling requests from clients, but do not store any state or information. Instead, they communicate with the back-end database to store information. MySQL Cluster is used in the data layer for the clustered version of the IIP prototype, which employs synchronous replication mechanism to guarantee that data is written to multiple nodes. Three node types of MySQL Cluster are used in the prototype, namely *management*, *data*, and *SQL*. *Management* nodes are responsible for managing the entire cluster. *Data* nodes are mainly employed for storing and retrieving data from memory and disk. *SQL* nodes are utilised for providing application access to the cluster. The IIP application instances communicate with MySQL Cluster through the SQL nodes by using JDBC, which is also responsible for load balancing queries across the SQL nodes. The current prototype's deployment architecture for the clustered version of the IIP is shown in Figure 3.9.

The IIP's functionalities are exposed as RESTful Web services, enabling brokered machine-to-machine (M2M) communications. However, human involvement is sometimes needed, especially during the development phase of new services. To accommodate this, a Web console for the IIP has been implemented to ease the management of resources in the IIP (e.g. information channels, devices, subscriptions, access control) for both information providers and information consumers with HTML pages. This Web application uses the same REST interfaces that information providers and information consumers use, where each resource requires the

[10] http://httpd.apache.org/

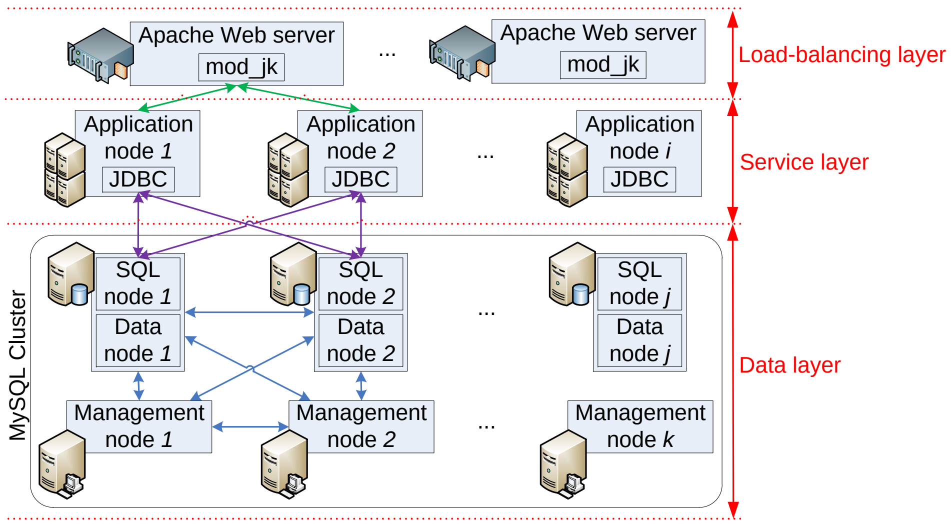


Figure 3.9: The IIP prototype deployment architecture for high availability and scalability

user to enter HTTP Basic authentication credentials. Figure 3.10 shows the main page of the Web console.

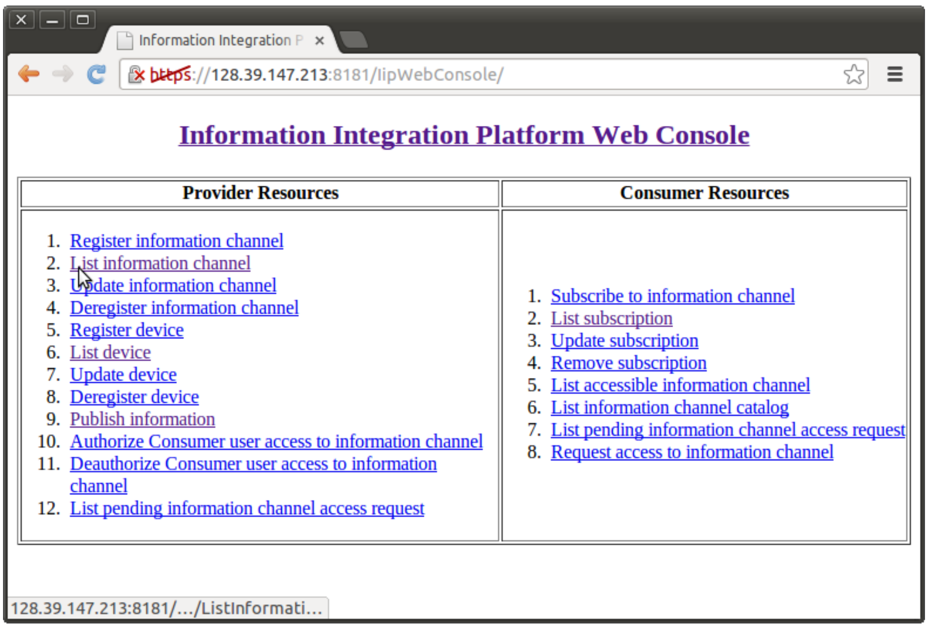


Figure 3.10: The main page of the IIP Web console

3.3 Services

Several patient-centric services in the realm of telehealth, telecare, and AAL are proposed in this thesis, utilising the previously described multi-layer architecture (as shown in Figure 3.1). The main components of the proposed services are hosted in the application layer, but they involve the lower layers as well, especially when it comes to data gathering capabilities.

3.3.1 Remote Health Monitoring Service

Remote health monitoring is one of the most important applications of telehealth as it enables healthcare personnel (e.g. doctors, nurses) as well as trusted family members and relatives to remotely monitor the health conditions of the patients. It is made possible by utilising different devices/sensors in the patients' surroundings.

The proposed remote health monitoring service consists of three main components: device gateway application, service bus, and Web portal. The device gateway application is used for gathering information from monitoring devices/sensors and sending the captured data to the service bus. The service bus acts as a service broker, mediating communications between device gateways and the Web portal in a publish/subscribe fashion.

The IIP is used as the service bus for the prototype implementation, while Android devices (tablet and smartphone) are used as the device/sensor gateways. The Web portal is an HTML page with AJAX, showing the latest measurements information from a remotely located patient, including pulse rate, blood oxygen saturation (SpO2), spirometry, and location. Since mobile devices are used at the patient's side, the patient is not restricted to do measurements from a specific location. The Web portal's user interface is shown in Figure 3.11. The remote health monitoring service is presented in paper III, IV, and V.

3.3.2 Emergency Notification Service

This service provides emergency situation notification through Short Message Service (SMS) and social media, and has been developed as two separate prototype applications (i.e. SOS-SMS and SOS-social media applications). Both applications make use of the IIP as the service bus. The ideal device interface at the patient's side would be a physical alarm button that can easily be pushed in case of emergency. For simplification, an Android application that mimics an alarm button is used in the current prototype. The Android application publishes an SOS message to an

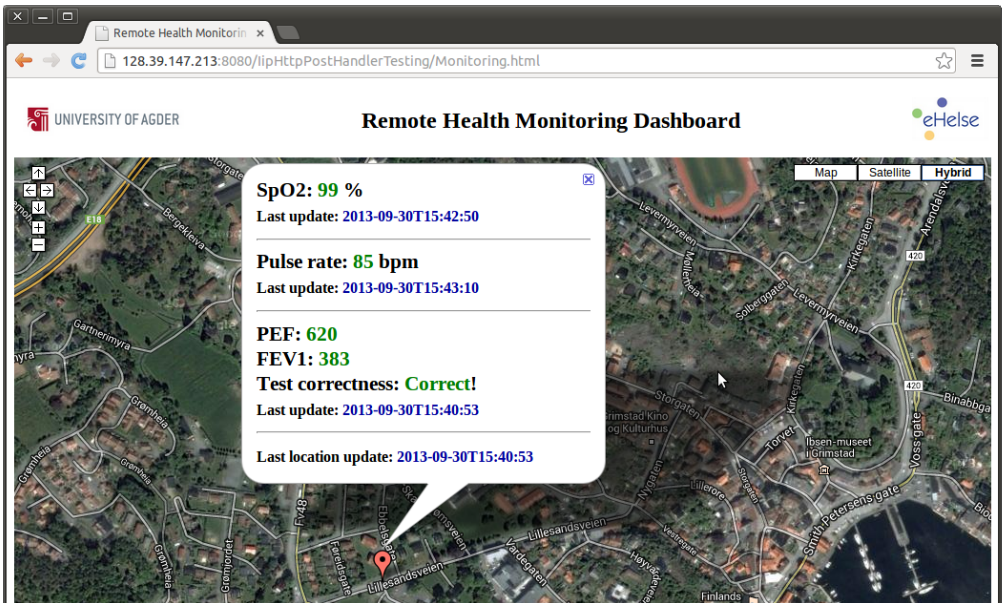


Figure 3.11: Remote health monitoring service user interface

SOS information channel in the IIP when the patient pushes the emergency button. Figure 3.12 shows the simplified emergency button.

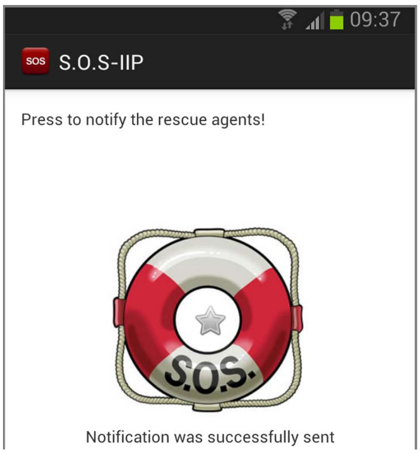
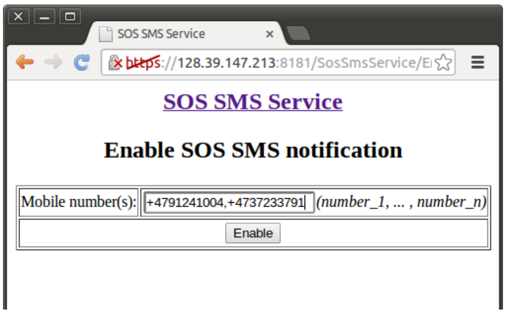


Figure 3.12: SOS button user interface

The SOS-SMS application subscribes to the SOS information channel and handles incoming notifications from the IIP whenever the patient sends SOS messages. A Web page is provided to the patient to administer which mobile numbers should be notified in case of emergency, and also to enable or disable this service. Figure 3.13(a) depicts a Web page for the patient to enable emergency notifications via SMS to a list of mobile numbers, and Figure 3.13(b) illustrates a Web page confirming that the service is enabled.

(a) Web interface for enabling specific mobile numbers to be notified



(b) Web interface confirming that SOS-SMS service is enabled

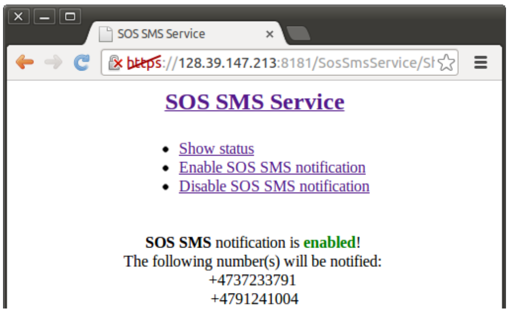
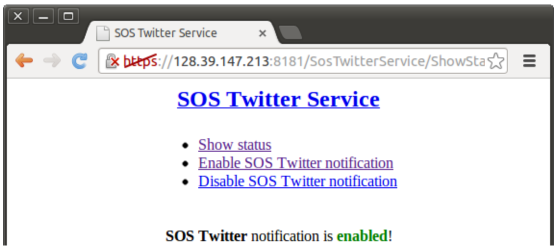


Figure 3.13: SOS-SMS service prototype application

Social media has been used extensively in the last couple of years, and it can be extended further to support emergency situations. People within the patient's social media circle are potential helpers in case of emergency (e.g. due to their close proximity to the patient). The SOS-social media prototype application has been developed, utilising a Twitter account to disseminate emergency information to the patient's Twitter followers, using the same SOS information channel used by the SOS-SMS application. The author's Twitter account is used for the current prototype. This service can be enabled or disabled by the patient through a Web interface, and Figure 3.14(a) shows a Web page confirming that the SOS-Twitter service is enabled. An example of a tweet for emergency notification is shown in Figure 3.14(b). The developed emergency notification services are presented in paper IV and V.

(a) Web interface confirming that SOS-Twitter service is enabled



(b) A tweet of an emergency notification

Figure 3.14: SOS-social media service prototype application

3.3.3 Ontology-Enhanced Home Automation Service

In a smart home environment, contextual information from various devices (e.g. home appliances, medical devices) are gathered to support AAL services via home automation. An ontology, acting as a knowledge base that describes relationships between different entities in the smart home environment is proposed and has been developed, following the Web Ontology Language (OWL) [107] standard representation. The current contexts being modelled in the ontology include activities, personal states, location, and the ambient smart home states. In addition to modelling contexts, the ontology also provides knowledge base for devices in the smart home. Figure 3.15 shows several important concepts (classes) of the proposed smart home ontology.

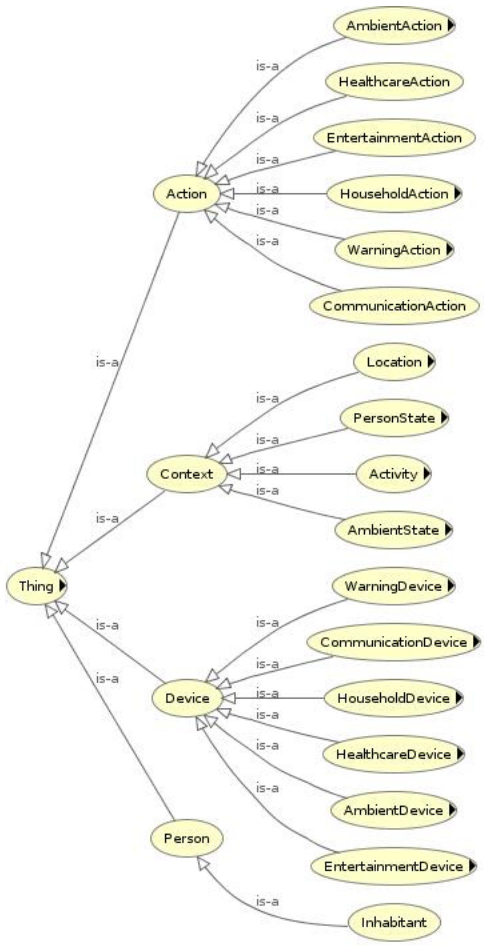


Figure 3.15: Smart home ontology

The device class has six subclasses that map one-to-one with the service enabler categories shown in Figure 3.1. The behaviours of the devices are represented by the Action class, which is also mapped one-to-one with the Device class, related by object properties. Individuals (instances) of the Action class and its subclasses represent the possible actions of the underlying devices. Context-aware applications in the application layer have to be informed about these changes so that exposed capabilities of the corresponding devices can be invoked.

Semantic Web Rule Language (SWRL) [108] is used for reasoning on the ontology to support context-aware applications and services in the smart home to make correct decisions. One application example of the SWRL rule usage is automatic window opening scenario. Let us assume that an inhabitant sets in his/her profile a comfort temperature range for the living room between 19 and 25 degrees Celcius. The home automation system detects the temperature of the living room with a thermometer and adjust the temperature state to hot whenever the thermometer's temperature value exceeds 25 degrees Celcius. A rule to simulate the knowledge base update in the ontology is as follows.

```
Room(?R) ∧ hasTemperature(?R,?T) ∧ hasTemperatureSen-
sor(?R,?S) ∧ hasTemperatureValue(?S,?V) ∧ hasMaxCom-
fortTemperature(?R,?C) ∧ swrlb:greaterThan(?V,?C) →
hasTemperatureValue(?T,?V) ∧ hasColdnessState(?T,"Hot")
```

This condition can be further utilised to control actuators, such as opening the window when the temperature state of the living room is hot. The rule for this task is as follows.

```
Room(?R) ∧ hasTemperature(?R,?T) ∧ hasCold-
nessState(?T,"Hot") ∧ Window(?W) ∧ hasRoomCompo-
nent(?R,?W) ∧ hasWindowAction(?W,?A) ∧ isClosed(?W,True)
→ open(?A,True) ∧ close(?A,False)
```

A more detailed description of the ontology-enhanced home automation service is presented in paper II.

3.4 Inside-Outside Smart Home Mobility

An important aspect of telehealth (e.g. remote health monitoring) is its pervasiveness to be used in different locations. In general, a patient's location can be categorised into two: at home and outside of home. In a smart home environment, various devices are expected to be present to assist the patient. The data gathered from these devices can be utilised to enhance the smart home's reasoning processes when combined with the data from body-worn devices. Internet connectivity in the smart home is also expected to be more reliable compared to cellular-based connectivity of personal portable device gateways (e.g. smartphones, tablets). Thus, it is better for the portable gateway used by the patient, which acts as the body-worn devices' gateway, to relay all measurement data to the smart home's gateway, for example via a WiFi link as depicted in Figure 3.16(a).

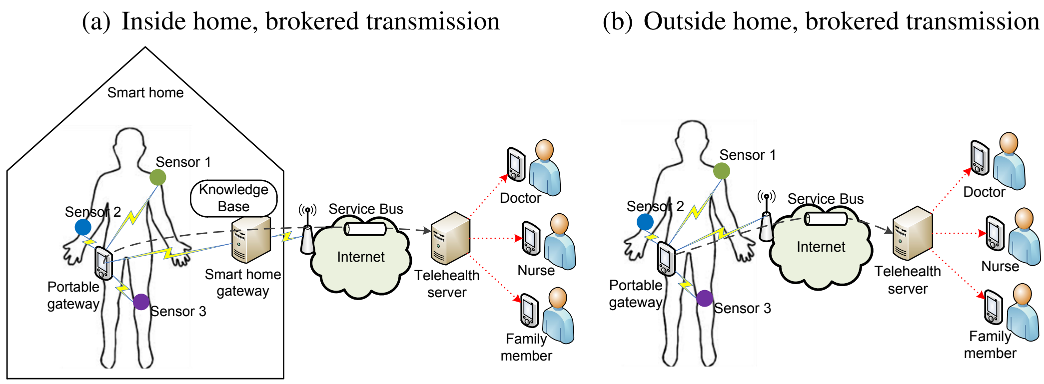


Figure 3.16: Inside and outside smart home brokered transmissions

When the patient leaves the smart home, bringing his/her portable body-worn sensors and a portable gateway, telehealth services are still expected to work reliably with limited data captured only from the portable sensors. These data should be transmitted to the back-end server due to the limited capabilities of the portable gateway for heavy processing, as well as to keep the remote healthcare personnel informed. The portable gateway application should be able to detect the patient's location (i.e. detect whether the patient is inside or outside the smart home), and transmits all captured data directly to the back-end server via a service bus (if used) if it detects the patient is outside the smart home, as shown in Figure 3.16(b). Since the location information is only required to alter the connectivity selection of the portable gateway, only the reachability information of the portable gateway to the smart home gateway is needed (i.e. the patient is considered to be at home when the portable gateway detects its reachability to the smart home gateway). This can be achieved by the portable gateway, for example, by saving the WiFi's SSID at home which provides connectivity to the smart home gateway. Location and connectivity are context data that the portable gateway application needs to capture, store, and decide upon. To better manage these contextual information, a context model can be utilised to formally represent different contextual situations. Figure 3.17 shows an example of ontology for the gateway applications.

Redundant Internet connections is necessary to maximise the always-online probability of the gateways and to maintain the real-time information dissemination to the back-end telehealth server. This may not be a big issue for the smart home gateway as both fixed and wireless Internet connectivity can easily be provisioned. However, it is challenging for the portable gateway as the fixed connectivity op-

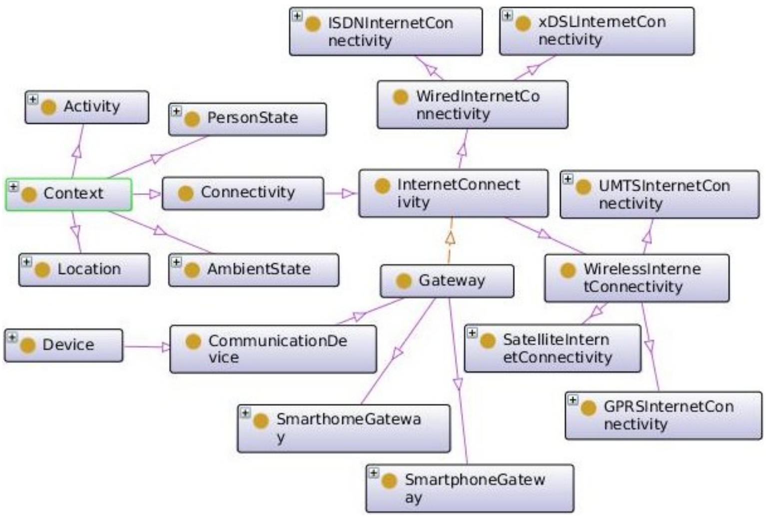


Figure 3.17: Device gateway ontology-based context model

tion does not apply. Table 3.2 shows possible combinations of Internet connectivity redundancy for both smart home and portable gateways.

Priority	Gateway	Primary	Alternative/Backup
1	Smarthome	xDSL	UMTS/WCDMA, GPRS/EDGE, WiFi, ISDN, Satellite
2	Smartphone	UMTS/ WCDMA	GPRS/EDGE, Satellite, GSM-SMS

Table 3.2: Possible connectivity redundancy combinations

Although Internet connectivity redundancy has been realised, telehealth service developers should not assume 100% up time for the gateways, and thus should incorporate offline storage in both gateway applications so that when no Internet connectivity is detected (e.g. utilising the ontology), all contextual information can be saved locally. To further reduce the Internet bandwidth usage, a simple logic can be conducted by the gateways, such as comparing two consecutive similar measurements within a specific time window, and sends the new measurement only if certain conditions are met. Energy conservation, however, should be taken into account for the portable gateway as heavy reasoning processes may impact the battery lifetime negatively. 2-level reasoning processes is proposed to be adopted, where a light reasoning is carried out by the gateways, and more sophisticated reasoning is conducted by the back-end telehealth server, as shown in Figure 3.18.

When the patient is inside the smart home, the level-1 reasoning process is delegated to the smart home gateway. This makes the portable gateway only act as a

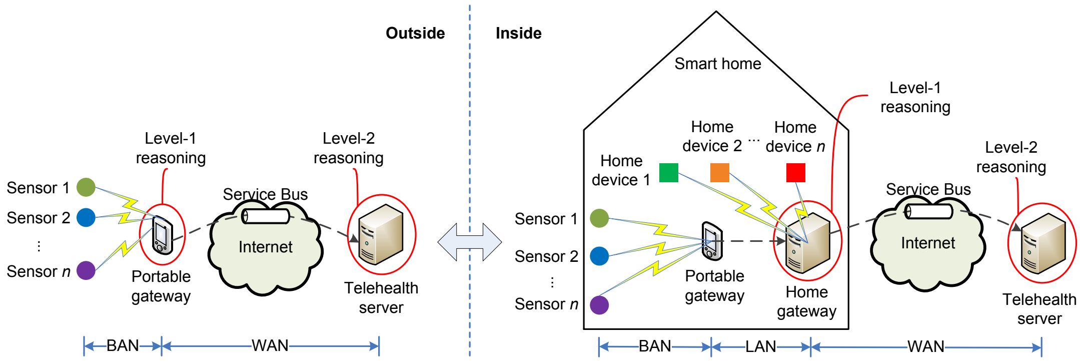


Figure 3.18: 2-level reasoning processes: outside (left) and inside (right)

relay of measurements from body-worn devices to minimise its own energy consumption.

3.5 Chapter Summary

A multi-layer guiding integration architecture was proposed and described in this chapter. It follows the SOA paradigm by exposing everyday objects' capabilities, both data gathering (sensing) and action capabilities, as modular and reusable service components (i.e. service enablers) that can be used in different applications for a wide range of services. The architecture enables the exposed capabilities of devices to be utilised by applications in a direct point-to-point manner or through a service broker (e.g. a service bus). Although the point-to-point approach is simpler to be used, it requires devices to send newly gathered information to all interested applications, which is an issue for battery-powered wireless devices.

A centralised service bus, called the IIP, was developed to solve this issue, incorporating the publish/subscribe messaging pattern. It follows the event-driven SOA concept and utilises RESTful Web services to enable out-of-the-box usage unlike many ESB technologies that require internal programmatic work to make them run. Information is shared through different information channels, and applications can manage their information channels (e.g. register, list, update, delete) via the exposed REST interfaces by using their credentials. Identity-based access control is used by the IIP, enabling applications to grant other applications read access to their information channels. However, applications can only publish new information (i.e. write access) to information channels they own.

To support high availability and scalability, clustering technology was adopted by the IIP. Remote health monitoring and emergency notification prototype services were presented in this chapter as proof-of-concepts of patient-centric health and care services following the guiding architecture, also affirming that the developed middleware functioned as intended. To support the applications in reasoning, ontology-based context modelling was proposed to be utilised for maintaining knowledge base. Portable personal devices (e.g. smartphones, tablets) were recommended to be used as the gateways for wireless medical sensors to support mobility of the users. In addition, redundant Internet connectivity was recommended in both portable and smart home gateways to maintain the uptime of the services that involve cloud-based server-side processing. 2-level reasoning processes was proposed in this chapter for the sake of energy conservation of the mobile devices.

Chapter 4

Discussion

4.1 Evaluation of Research Questions

This research work focused on the technology aspects of delivering healthcare services to patients and inhabitants living independently at their homes while still being taken care of remotely by healthcare personnel. Research questions raised in sub-chapter 1.3 are answered as follows.

1. *How can different devices be integrated to support eHealth services and to avoid vertical "silos" in a smart home environment?*

   Integration of various devices in a smart home environment should adopt the SOA paradigm. For integration purposes, the physical form of devices is less important compared to their exposed capabilities, such as to capture information or to perform certain actions. These capabilities should be exposed as reusable service components, also known as service enablers, to avoid tight coupling between devices and services. The proposed guiding integration architecture lets service designers and developers choose whether to use the reusable service enablers per se or to combine them with other applications. Service enablers can be utilised in a direct point-to-point manner or through a service bus. Either way, the traditional vertical "silo" integration approach has fundamentally been dismissed as reusable service enablers take over. The centralised service bus approach provides simpler interaction management between applications, especially as the number of services grows, and reduces the number of messages that has to be sent from devices to services significantly. Nonetheless, direct point-to-point integration is still left as an option for specific scenarios.

2. *How can eHealth services be developed and deployed rapidly to support*

changing needs of the inhabitants and the patients in a smart home environment?

A multi-layer guiding integration architecture is proposed in this thesis. Various applications hosted in the application layer (the top layer in the integration architecture) that serve different purposes can be developed, modified, and deployed rapidly following the Web 2.0 way by adopting SOA principles. These applications can be deployed either inside the smart home or in the cloud depending on the services' requirements. They can combine the exposed service enablers with external third party applications for novel services with rich features to support the inhabitants and the patients.

3. *How should eHealth services be designed to support the inhabitants' and the patients' mobility inside and outside their homes?*

   eHealth services tailored for inhabitants and patients living at their homes should be designed to be able to detect whether they are inside their homes or not. This is an important feature especially when these services make use of home appliances' capabilities which can no longer be utilised when the inhabitants and the patients are outside of their homes. This context-awareness concept enables eHealth services to only utilise available capabilities of devices surrounding the inhabitants and the patients without sacrificing the entire functionalities of the services.

4. *How should information gathered from various devices be modelled in order to provide a consistent view across different domains?*

   A unifying information model that integrates vendor-specific models is needed in order to enable a consistent information view across disparate business domains. This can be achieved when exposing the capabilities of the devices by following a specific schema to maintain consistency. Ontologies can be used for context and general information modelling as well. This work does not formally specify the information model of the captured data by the devices. Nevertheless, the proposed and developed service broker (i.e. the IIP) manages and models the gathered information in different information channels that can contain more than one parameter. All services that utilise the IIP will receive information in structured and consistent manners.

5. *How can eHealth services accommodate an increasing number of users while minimising downtime?*

Unless the number of users are fixed and known in advance, eHealth services should be designed to accommodate additional new users after their initial deployments. Thus, scalability is a crucial aspect that should be taken into account throughout the entire end-to-end chain of eHealth services. Serving nodes for eHealth services have limitations in terms of processing and storage capacities, so it is very important to design software that enables additional resources to be deployed when needed (i.e. horizontal scaling). At the same time, eHealth services should be available whenever needed by the users and should be resilient to failures in order to maintain their reliability. Redundant serving nodes with clustering technology, as utilised in the IIP prototype, can be used to achieve high availability of eHealth services while providing the possibility for horizontal scaling to support scalability.

6. *How can security and privacy aspects of eHealth services be maintained without compromising usability and performance?*

   Security and privacy are two essential aspects of eHealth services that should be taken into account seriously end-to-end, as vital signs and other health-related information are very sensitive by nature. The security layer in the proposed integration architecture lies vertically across all other layers so that security measures can be implemented in the different horizontal layers according to the security level requirements. The performance of the entire system is affected negatively the more security and privacy measures are used, as verified in several experiments in this work. In addition, multi-level security and privacy approaches may require additional actions or procedures that should be performed by the inhabitants and the patients, which result in higher complexity of service usage for them. A good balance between security, privacy, and usability aspects are necessary to be taken into account during the design of eHealth services.

4.2 Performance Evaluation of the IIP

A service broker called the Information Integration Platform (IIP) has been designed and developed following a publish/subscribe message exchange pattern. It aims to break the vertical “silo” integration approach and to simplify the interaction management between services as the number of services grows. As it plays a central role in converging information dissemination between devices and Internet-based services in the proposed architecture, it is important to evaluate its performance.

Suppose there are N different applications (i.e. information consumers) that make use of an information gathered by a device (i.e. an information provider). If a point-to-point interaction is utilised, the information provider needs to maintain the end-points of these N different information consumers as well as establishing network connections to them. On the other hand, if a brokered approach is used, the information provider only needs to maintain one network connection to the broker, which is responsible for delivering the information to different information consumers. The complexity ratio for maintaining peer-connections between the two approaches is N to 1. Energy consumption ratio between the two approaches are roughly in the same order as well. This makes the brokered approach advantageous especially when the information providers are wireless sensors.

Despite the IIP's advantages, which are inherent to the centralised brokered approach, the internal processing time adds additional delay to the whole information delivery process. Several experiments were conducted to compare the total publication-and-notification times of the brokered approach through the standalone version of the IIP and the total direct notification times of the point-to-point approach without any security feature. A multi-threaded tester application was developed, acting as an information provider sending one publication per second HTTP POST request. Figure 4.1(a) shows the comparisons of the average notification times from an information provider to all subscribers, which are varied between 1 and 400.

(a) Point-to-point and through the IIP average notification times (b) Through the IIP to point-to-point average notification times ratio

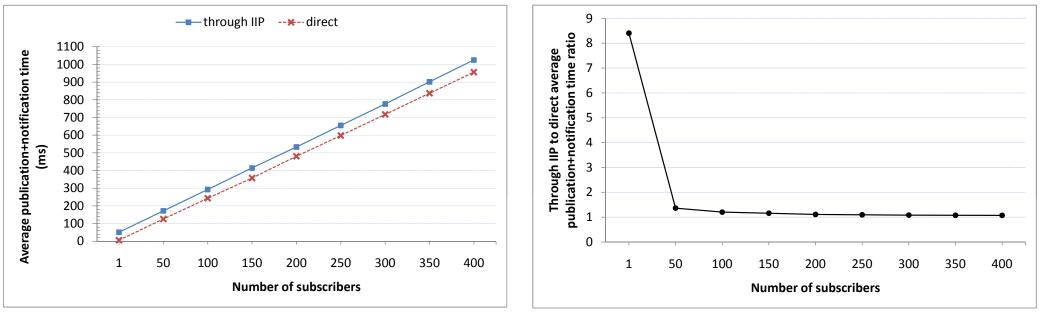
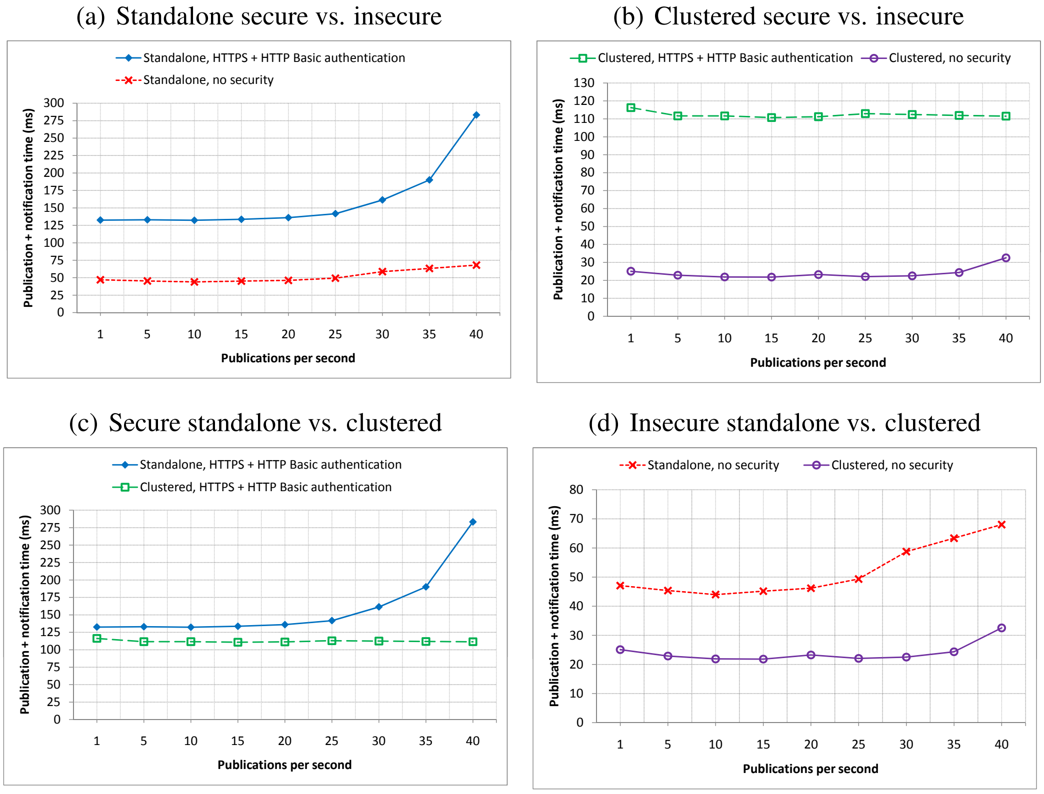


Figure 4.1: Brokered through the IIP and direct point-to-point average notification times comparisons

The processing times of both approaches tend to increase in a linear fashion with an increased number of subscribers, where the total average processing times for the brokered approach were higher compared to the direct point-to-point approach. This is mainly attributed to the internal processing in the IIP, which adds an overhead compared to the direct approach. This internal processing time, however, becomes less significant with the increasing number of subscribers, as shown in Figure 4.1(b), where the ratio between the two approaches gets near to 1 as the number of subscribers grows. These experiments are presented in paper IV.

To support high availability and scalability features, the IIP has been implemented in a clustered environment. Experiments were carried out to compare the performance of the standalone IIP prototype with the clustered version in terms of total publication-and-notification average times. In addition, comparisons between the inclusion of the security and privacy scheme and without it were conducted as well. A slightly modified version of the tester application was used as an information provider, sending HTTP POST publication messages with varying publication rates from 1 to 40 publications per second. The results are depicted in Figure 4.2.



(a) Standalone secure vs. insecure

(b) Clustered secure vs. insecure

(c) Secure standalone vs. clustered

(d) Insecure standalone vs. clustered

Figure 4.2: Publication-and-notification time comparisons with publication rate as variable

In general, the clustered realisation of the IIP performed better compared to its standalone counterpart in terms of total average publication-and-notification times as shown in Figure 4.2(c) and 4.2(d). The performance difference gap tended to widen as the publication rate increased, especially when the security mechanism was applied as depicted in Figure 4.2(c). In both implementations, the applied security measures decreased the overall performance of the IIP as shown in Figure

4.2(a) and 4.2(b).

An information channel in the IIP can consist of more than one parameter, depending on the needs of the information provider who owns it. For instance, SpO2, pulse rate, and SOS information channels may use 1 parameter each, location information channel typically has 2 parameters (i.e. latitude and longitude), while spirometry information channel can have 13 parameters for each measurement. Several experiments were conducted using similar tester application as in previous experiments, except that the publication rate was fixed at one publication per second while the number of parameters was varied instead, ranging from 1 to 700 parameters. Figure 4.3 shows the results of these experiments.

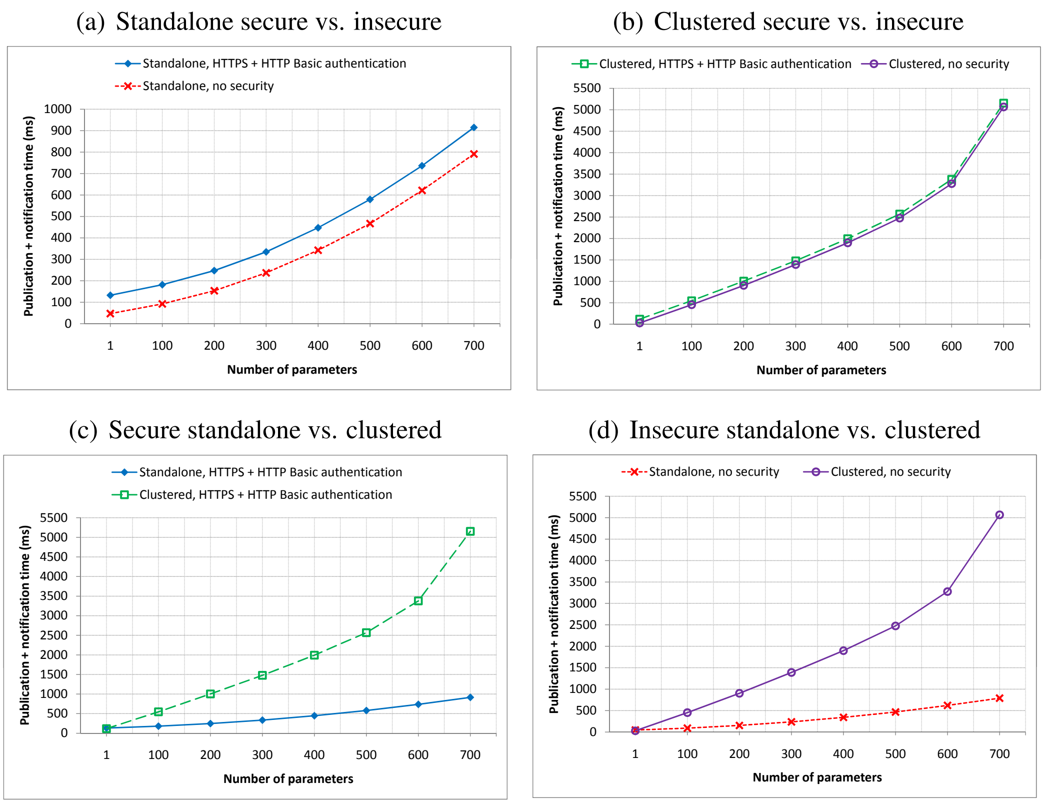


Figure 4.3: Publication and notification time comparisons with number of parameters as variable

The applied security mechanisms added extra latency in both standalone and clustered versions of the IIP with the increasing number of parameters being used as depicted in Figure 4.3(a) and 4.3(b). The clustered implementation of the IIP performed better than the standalone version when the number of parameters being used were small (i.e. close to 1), but its performance degraded significantly as the number of parameters grew, as shown in Figure 4.3(c) and 4.3(d). The synchronisation process between data nodes plays a major role in the performance degradation of the clustered version of the IIP as the number of parameters grows. These experiments are presented in paper V.

4.3 Deployment Location of the IIP

The IIP is at the heart of the proposed integration solution in this work, although the option for a point-to-point approach is made available as well in the proposed guiding integration architecture, as shown in Figure 3.1. The IIP itself is a logical entity that can be deployed in different places. Figure 4.4 illustrates two possible deployment location alternatives for the IIP as a service broker.

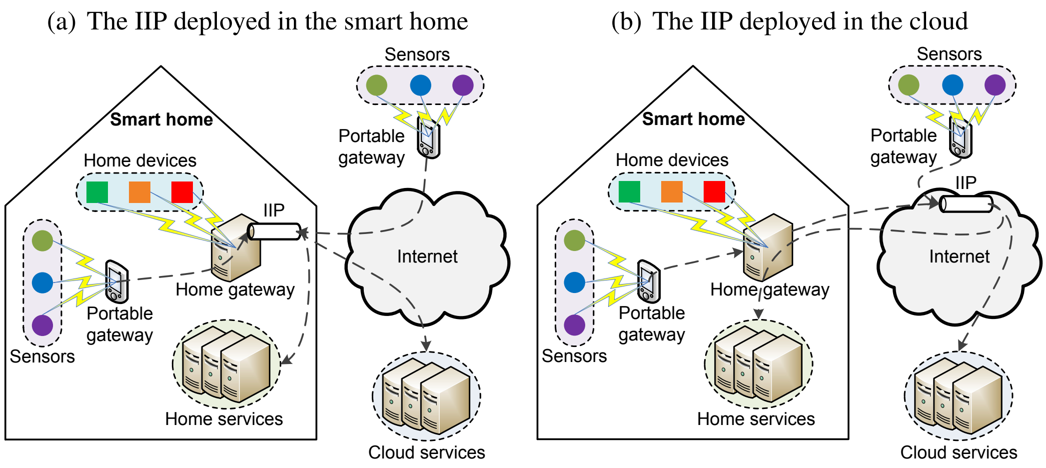


Figure 4.4: Deployment location alternatives of the IIP

The first alternative is to deploy the IIP inside the smart home, for example in the home gateway as shown in Figure 4.4(a). When the patient is at home, all sensors that he/she wears as well as home appliances that he/she interacts with send information to the IIP in the home gateway, which will forward the information to various services either deployed at home (i.e. home services) or on the Internet (i.e. cloud services). This alternative is relatively secure as the IIP is mainly used within the smart home, but it becomes more difficult to accommodate scenarios where the patient goes out of the home. The second alternative, as depicted in Figure 4.4(b), is to deploy the IIP in the cloud. In this set-up, the IIP is publicly accessible by any entity connected to the Internet, and thus, is easier to accommodate integration of various information sources globally. However, Internet connectivity is essential in this deployment scheme, and it can be considered as a drawback from the smart home's standpoint as Internet connectivity is still required even when only the home services are being utilised. To fully support inside and outside of home scenarios, the home gateway should be accessible from the Internet in both alternatives. Thus, a strong firewall rules should be employed in the home gateway to secure the resources in the smart home.

The University of Agder (UiA) has set-up a replica of the Norwegian Health Network (NHN) for test purposes in a project called *e-Helse-testsenter*. A secure network with multi-level firewall rules separates various involved actors' workspaces in different Virtual Local Area Networks (VLANs) that are interconnected with one another, as depicted in Figure 4.5.

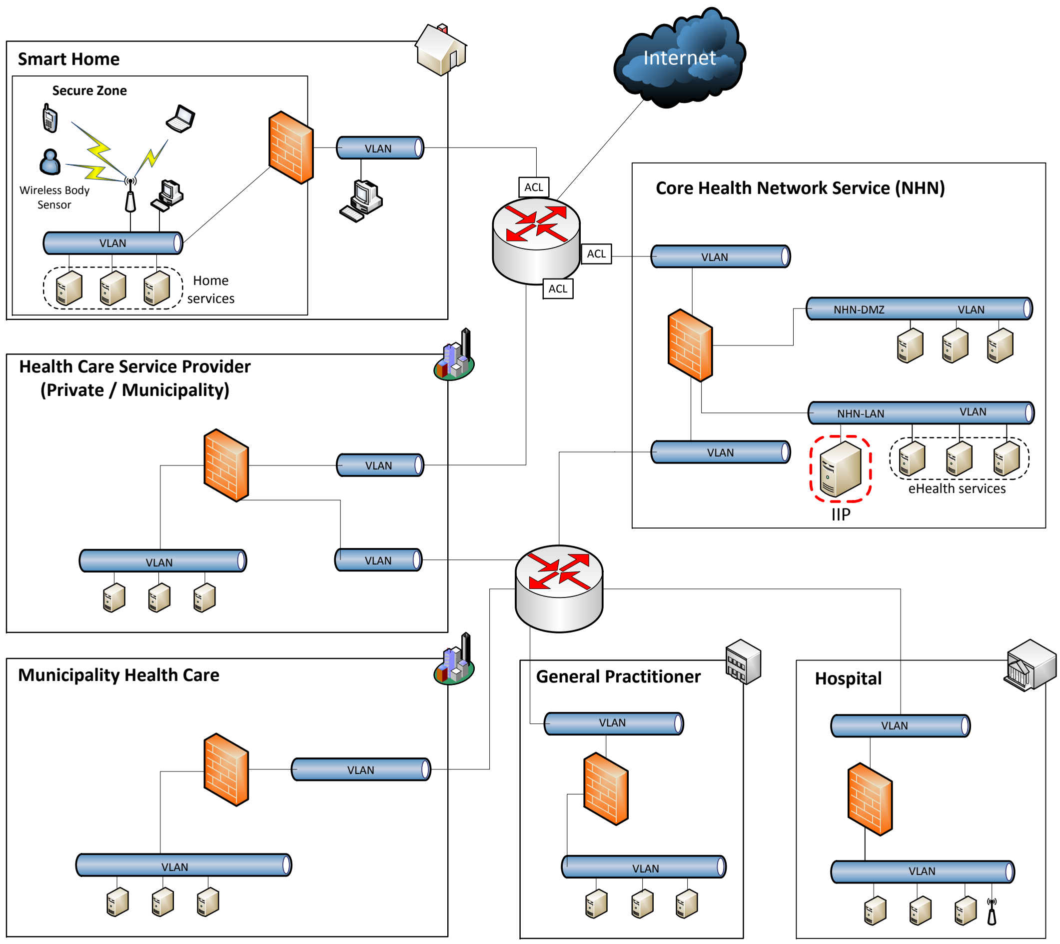


Figure 4.5: *e-Helse-testsenter* network overview

The IIP is deployed in the core NHN service network alongside other eHealth services. These services are reachable from the secure smart home network, and can be reached from the Internet through a Virtual Private Network (VPN) tunnel. The latter requires the patient to use a dedicated Subscriber Identity Module (SIM) card inserted in his/her mobile gateway device. The main advantage of this approach is the high security of message exchanges since a dedicated closed network is utilised instead of relying purely on the Internet. However, it becomes quite difficult for third party service providers to host their applications outside the NHN, if not impossible. Multiple layers of security measures possess a risk in decreasing service performance as well.

The IIP deployment within the *e-Helse-testsenter* project is a stepping stone towards its deployment in the actual NHN as part of an ongoing European project *United4Health*[1] in collaboration with several healthcare institutions and industry partners. In this project, trials with real patients are planned to be conducted where the IIP will act as the back-end information broker from medical devices to a specific healthcare application.

[1] http://united4health.eu/

Chapter 5

Summary and Future Directions

5.1 Summary

An integration architecture based on SOA principles has been proposed and described in this thesis to support the provision of health and care services, such as telehealth, telecare, and AAL, in a smart home environment to inhabitants and patients in a flexible and timely manner. It enables them to live more independently at their homes as long as possible with minimum physical intervention from healthcare personnel while still being taken care of remotely. By adopting the proposed architecture, the capabilities of different devices can be reused by different applications/services, and can be composed or mashed-up as new services that can be developed and deployed rapidly.

The capabilities of the various devices, which are exposed as service enablers, can be utilised directly by applications in a point-to-point fashion or through a service broker. To reduce power consumption, the device should avoid sending multiple messages to different services by offloading the information dispatching task to an external broker, adopting the mobile cloud computing approach.

A service broker called the IIP has been designed and developed following a publish/subscribe message exchange pattern. It aims to break the vertical “silo” integration approach and to simplify the interaction management between services as the number of services grows. RESTful Web services are utilised for exposing the platform’s functionalities, interfacing both information providers and consumers. HTTPS and HTTP Basic authentication scheme are used as security measures. To support high availability and scalability, clustering technology is incorporated in the IIP. Conducted experiments on the IIP showed that the total average processing times were higher when the IIP was used compared to the direct point-to-point approach, which was mainly caused by the internal processing in the IIP. This internal processing time became less significant as the number of subscribers was increased. The clustered version of the IIP performed better in general compared to its standalone counterpart when the publication rate was increased. However, the clustered IIP suffered more compared to the standalone IIP when the number of parameters being used was increased.

Several service prototypes, including remote health monitoring and emergency alarming services, have been developed to demonstrate the feasibility of the proposed integration architecture to be realised in a real-life setting. An ontology-based context modelling was proposed to be used for maintaining knowledge base to support reasoning processes in the applications. This would enable the deployed applications to be context-aware of the inhabitants' and the patients' situations, and to react accordingly in assisting their daily activities.

5.2 Future Directions

This dissertation addressed the adoption of SOA principles to better support the design, development and provision of healthcare services in a smart home environment. A guiding integration architecture has been proposed, a service broker called the IIP has been designed and developed, and several services have been realised utilising off-the-shelf wireless medical devices as data sources. Since the research work covers quite a wide range of areas in the realm of integration, many aspects are left out to be addressed further in the continuation of this work. Several important future work directions are summarised as follows.

- Integration of more off-the-shelf healthcare consumer electronics as well as well-known home automation standards, such as the KNX, following the proposed guiding integration architecture. The current service prototypes in this research work incorporate a very limited number of devices.

- Realisation of a common service composition platform as a service, either from scratch or by combining existing solutions, to support easy composite service creation. In this work, service composition is left out entirely to the service designers and developers.

- Incorporation of more features to the currently developed IIP. Most importantly to support action commands from services to actuators which is missing in the current implementation. Although theoretically this can be simply achieved by swapping the role of information provider and information consumer, a more detailed design especially related to the access control aspect is necessary to be carried out. In addition, support of newer IoT protocols such as CoAP and MQTT are beneficial to be included as well.

- Development of more sophisticated intelligent services that make use of various data sources for AAL, telehealth, and telecare services following the proposed architecture would be beneficial to further find out how the architecture can be improved. This can include the adoption of more complex reasoning processes adopted from the artificial intelligence domain.

- Inclusion of stronger security and privacy approaches. Although security and privacy measures are taken into consideration in this work, they may not be enough for specific eHealth services. Multiple authentication factors, for example, can be incorporated and enforced in certain services.

REFERENCES

[1] UNFPA, "Ageing in the Twenty-First Century: A Celebration and A Challenge," *[Online]*, Available: http://www.unfpa.org/webdav/site/global/shared/documents/publications/2012/Ageing-Report_full.pdf [Accessed 15th November 2013].

[2] M. Keen, A. Acharya, S. Bishop, A. Hopkins, S. Milinski, C. Nott, R. Robinson, J. Adams, and P. Verschueren, *Patterns: Implementing an SOA Using an Enterprise Service Bus*. IBM Corp., July 2004.

[3] C. Pautasso, O. Zimmermann, and F. Leymann, "Restful Web Services vs. "Big"' Web Services: Making the Right Architectural Decision," in *Proceedings of the 17th International Conference on World Wide Web*, ACM, 2008, pp. 805–814.

[4] A. T. Kearney, "The Mobile Economy 2013,", 2013.

[5] Legatum Institute, "The Legatum Prosperity Index 2013," *[Online]*, Available: http://media.prosperity.com/2013/pdf/publications/PI2013Brochure_WEB.pdf [Accessed 14th November 2013].

[6] W. Lutz, W. Sanderson, and S. Scherbov, "The coming acceleration of global population ageing," *Nature*, vol. 451, no. 7179, pp. 716–719, Nature Publishing Group, 2008.

[7] Y. Guillemette and W. Robson, *No Elixir of Youth: Immigration Cannot Keep Canada Young*. CD Howe Institute, 2009.

[8] E. Mørk, "Seniorer i Norge 2010," *Statistisk sentralbyrå*, February 2011.

[9] S. Sataline and S. Wang, "Medical Schools Cant Keep Up," *Wall Street Journal [Online]*, Available: http://online.wsj.com/article/SB10001424052702304506904575180331528424238.html [Accessed 10th May 2012], 2012.

[10] World Health Organization, *A Universal Truth: No Health Without a Workforce*, November 2013.

[11] Paraprofessional Healthcare Institute, "Occupational Projections for Direct-Care Workers 2010–2020," *[Online]*, Available: http://phinational.org/sites/phinational.org/files/phi_

factsheet1update_singles_2.pdf [Accessed 16th November 2013], February 2013.

[12] E. Fife and F. Pereira, "Digital home health and mHealth: Prospects and challenges for adoption in the US," in *2011 50th FITCE Congress (FITCE)*, IEEE, 2011, pp. 1–11.

[13] M.-P. Gagnon, G. Paré, H. Pollender, J. Duplantie, J. Côté, J.-P. Fortin, R. Labadie, E. Duplàa, M.-C. Thifault, F. Courcy, C. A. McGinn, B. A. Ly, A. Trépanier, and F.-B. Malo, "Supporting work practices through telehealth: impact on nurses in peripheral regions," *BMC Health Services Research*, vol. 11, no. 1, pp. 27, BioMed Central Ltd, February 2011.

[14] S. Koch, "Home telehealth–Current state and future trends," *International journal of medical informatics*, vol. 75, no. 8, pp. 565–576, Elsevier, 2006.

[15] M. M. Maheu, P. Whitten, and A. Allen, *E-health, Telehealth, and Telemedicine: A Guide to Startup and Success*. Jossey-Bass, 2002.

[16] D. W. Nickelson, "Telehealth and the Evolving Health Care System: Strategic Opportunities for Professional Psychology.," *Professional Psychology: Research and Practice*, vol. 29, no. 6, pp. 527, American Psychological Association, 1998.

[17] P. A. Jennett, L. A. Hall, D. Hailey, A. Ohinmaa, C. Anderson, R. Thomas, B. Young, D. Lorenzetti, and R. E. Scott, "The socio-economic impact of telehealth: a systematic review," *Journal of telemedicine and telecare*, vol. 9, no. 6, pp. 311–320, SAGE Publications, 2003.

[18] Teknologirådet, "Future Aging and New Technology," *Teknologirådet*, 2009.

[19] M. Prensky, "Digital Natives, Digital Immigrants part 1," *On the Horizon*, vol. 9, no. 5, pp. 1–6, MCB UP Ltd, October 2001.

[20] Robert M. Anderson and Martha M. Funnell, "Patient empowerment: Myths and misconceptions," *Patient Education and Counseling*, vol. 79, no. 3, pp. 277–282, June 2010.

[21] N. A. Abdullah and N. Zakaria, "Sociability aspects in e-health community: A review," in *2010 International Symposium in Information Technology (ITSim)*, June 2010, pp. 972–976.

[22] E. Thygesen, M. M. F. Fensli, R. Skaar, H. I. Sævareid, Y. Li, and R. Fensli, "User requirements for a Personalized Electronic Community for Elderly People with Risk of Marginalization," in *9th Scandinavian Conference on Health Informatics (SHI)*, August 2011, pp. 50–54.

[23] Y. B. D. Trinugroho, F. Reichert, and R. W. Fensli, "A SOA-Based eHealth Service Platform in Smart Home Environment," in *2011 13th IEEE International Conference on e-Health Networking Applications and Services (Healthcom)*, 2011, pp. 201–204.

[24] K. Ashton, "That Internet of Things Thing," *RFiD Journal*, vol. 22, pp. 97–114, 2009.

[25] S. Sarma, D. L. Brock, and K. Ashton, "The Networked Physical World," *Auto-ID Center White Paper MIT-AUTOID-WH-001*, October 2000.

[26] D. Lake, A. Rayes, and M. Morrow, "The Internet of Things," *The Internet Protocol Journal*, vol. 15, no. 3, pp. 10–19, September 2012.

[27] F. Mattern and C. Floerkemeier, "From the Internet of Computers to the Internet of Things," *From Active Data Management to Event-Based Systems and More*, pp. 242–259, Springer, 2010.

[28] N. Gershenfeld, R. Krikorian, and D. Cohen, "The Internet of things.," *Scientific American*, vol. 291, no. 4, pp. 76, 2004.

[29] M. Brenner and M. Unmehopa, "The Silo Syndrome and its Solution," *The Open Mobile Alliance: Delivering Service Enablers for Next-Generation Applications*, pp. 7–20, Wiley Online Library, April 2008.

[30] U. Batra and S. Mukharjee, "Enterprise Application Integration (Middleware): Integrating stovepipe applications of varied enterprises in distributed middleware with Service Oriented Architecture," in *2011 3rd International Conference on Electronics Computer Technology (ICECT)*, 2011, pp. 226–230.

[31] W. A. Ruh, F. X. Maginnis, and W. J. Brown, *Enterprise Application Integration: A Wiley Tech Brief.* John Wiley & Sons, 2002.

[32] N. L. Snee and K. A. McCormick, "The Case for Integrating Public Health Informatics Networks," *Engineering in Medicine and Biology Magazine, IEEE*, vol. 23, no. 1, pp. 81–88, 2004.

[33] J. Barlow, R. Curry, T. Chrysanthaki, J. Hendy, and N. Taher, "Developing the capacity of the remote care industry to supply Britains future needs," *Health and Care Infrastructure Research and Innovation Centre*, November 2012.

[34] D. Kwo, "Systems architecture for integrated care," *International Journal of Integrated Care*, vol. 12, no. Suppl1, Igitur Publishing and Archiving Services, 2012.

[35] M. Swindells, "Population Health Management," *ITNOW*, pp. 8–11, June 2012.

[36] S. C. Peirce, A. R. Hardisty, A. D. Preece, and G. Elwyn, "Designing and implementing telemonitoring for early detection of deterioration in chronic disease: Defining the requirements," *Health Informatics Journal*, vol. 17, no. 3, pp. 173–190, SAGE Publications, 2011.

[37] M. Rigby, "Integrating Health and Social Care Informatics to Enable Holistic Health Care," in *9th International Conference on Wearable Micro and Nano Technologies for Personalized Health*, 2012, pp. 41–51.

[38] E. Thomas, "Service-Oriented Architecture: Concepts, Technology, and Design," *Prentice Hall*, 2005.

[39] M. P. Papazoglou and W.-J. Van Den Heuvel, "Service oriented architectures: approaches, technologies and research issues," *The VLDB journal*, vol. 16, no. 3, pp. 389–415, Springer, 2007.

[40] M. N. Huhns and M. P. Singh, "Service-Oriented Computing: Key Concepts and Principles," *IEEE Internet Computing*, vol. 9, no. 1, pp. 75–81, 2005.

[41] N. Josuttis, *SOA in Practice*. O'Reilly, first edition, 2007.

[42] D. Krafzig, K. Banke, and D. Slama, *Enterprise SOA: Service-Oriented Architecture Best Practices*. Prentice Hall Professional, 2005.

[43] D. K. Barry, *Web Services and Service-Oriented Architecture: The Savvy Manager's Guide*. Morgan Kaufmann Pub, 2003.

[44] B. Littlewood and L. Strigini, "Software Reliability and Dependability: a Roadmap," in *Proceedings of the Conference on The Future of Software Engineering*, ACM, 2000, pp. 175–188.

[45] A. Avižienis, J.-C. Laprie, and B. Randell, *Fundamental Concepts of Dependability*. Research Report No. 1145, LAAS-CNRS, 2001.

[46] W. Stallings and L. Brown, *Computer Security: Principles and Practice*. Prentice-Hall, 2008.

[47] C. P. Friedman and J. C. Wyatt, *Evaluation Methods in Biomedical Informatics*, Health Informatics Series. Springer, 2nd edition, 2006.

[48] R. Costa, D. Carneiro, P. Novais, L. Lima, J. Machado, A. Marques, and J. Neves, "Ambient Assisted Living," in *3rd Symposium of Ubiquitous Computing and Ambient Intelligence 2008*, Springer, 2009, pp. 86–94.

[49] T. Kleinberger, M. Becker, E. Ras, A. Holzinger, and P. Müller, "Ambient Intelligence in Assisted Living: Enable Elderly People to Handle Future Interfaces,", in *Universal Access in Human-Computer Interaction. Ambient Interaction*, Constantine Stephanidis, Ed., vol. 4555 of *Lecture Notes in Computer Science*, pp. 103–112, Springer Berlin Heidelberg, 2007.

[50] A. N. Belbachir, M. Drobics, and W. Marschitz, "Ambient Assisted Living for ageing well - an overview," *e & i Elektrotechnik und Informationstechnik*, vol. 127, no. 7-8, pp. 200–205, Springer-Verlag, 2010.

[51] V. Fuchsberger, "Ambient Assisted Living: Elderly People's Needs and How to Face Them," in *Proceedings of the 1st ACM International Workshop on Semantic Ambient Media Experiences*, ACM, 2008, SAME '08, pp. 21–24.

[52] H. Sun, V. De Florio, N. Gui, and C. Blondia, "Promises and Challenges of Ambient Assisted Living Systems," in *Sixth International Conference on Information Technology: New Generations (ITNG '09)*, 2009, pp. 1201–1207.

[53] G. Van Den Broek, F. Cavallo, and C. Wehrmann, Eds., *AALIANCE Ambient Assisted Living Roadmap*, vol. 6. IOS Press, 2010.

[54] M. Weiser, "Some Computer Science Issues in Ubiquitous Computing," *Communications of the ACM*, vol. 36, no. 7, pp. 75–84, ACM, 1993.

[55] S. Poslad, *Ubiquitous Computing: Smart Devices, Environments and Interactions*. John Wiley & Sons Ltd., 2009.

[56] R. Want, B. N. Schilit, N. I. Adams, R. Gold, K. Petersen, D. Goldberg, J. R. Ellis, and M. Weiser, "An Overview of the PARCTAB Ubiquitous Computing Experiment," *IEEE Personal Communications*, vol. 2, no. 6, pp. 28–43, IEEE, 1995.

[57] S. Kareem and I. S. Bajwa, "A Virtual Telehealth Framework: Applications and Technical Considerations," in *2011 7th International Conference on Emerging Technologies (ICET)*, September 2011, pp. 1–6.

[58] J. Barlow, D. Singh, S. Bayer, and R. Curry, "A systematic review of the benefits of home telecare for frail elderly people and those with long-term conditions," *Journal of Telemedicine and Telecare*, vol. 13, no. 4, pp. 172–179, 2007.

[59] "Implementing telecare: Strategic analysis and guidelines for policy makers, commissioners and providers," *Audit Commission*, July 2004.

[60] J. Polisena, K. Tran, K. Cimon, B. Hutton, S. McGill, K. Palmer, and R. E. Scott, "Home telehealth for chronic obstructive pulmonary disease: a systematic review and meta-analysis," *Journal of Telemedicine and Telecare*, vol. 16, no. 3, pp. 120–127, SAGE Publications, 2010.

[61] G. E. Smith, A. M. Lunde, J. C. Hathaway, and K. S. Vickers, "Telehealth Home Monitoring of Solitary Persons With Mild Dementia," *American Journal of Alzheimer's Disease and Other Dementias*, vol. 22, no. 1, pp. 20–26, SAGE Publications, 2007.

[62] J. L. DelliFraine and K. H. Dansky, "Home-based telehealth: a review and meta-analysis," *Journal of Telemedicine and Telecare*, vol. 14, no. 2, pp. 62–66, SAGE Publications, 2008.

[63] M. Walsh and J. R. Coleman, "Trials and Tribulations: A Small Pilot Telehealth Home Care Program for Medicare Patients," *Geriatric Nursing*, vol. 26, no. 6, pp. 343–346, 2005.

[64] S. Nourizadeh, C. Deroussent, Y.-Q. Song, and J.-P. Thomesse, "A Distributed Elderly Healthcare System," in *International Workshop on Mobilizing Health Information to Support Healthcare-related Knowledge Work (MobiHealthInf)*, January 2009.

[65] J. Kim, H.-S. Choi, H. Wang, N. Agoulmine, M. J. Deerv, and J. W.-K. Hong, "POSTECH's U-Health Smart Home for Elderly Monitoring and Support,"

in *2010 IEEE International Symposium on A World of Wireless Mobile and Multimedia Networks (WoWMoM)*, 2010, pp. 1–6.

[66] I. Pau, F. Seoane, K. Lindecrantz, M. A. Valero, and J. Carracedo, "Home e-health system integration in the Smart Home through a common media server," in *31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2009, pp. 6171–6174.

[67] N. Noury, C. Villemazet, A. Fleury, P. Barralon, P. Rumeau, N. Vuillerme, and R. Baghai, "Ambient Multi-Perceptive System with Electronic Mails for a Residential Health Monitoring System," in *28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, 2006, pp. 3612–3615.

[68] N. Noury, "AILISA : experimental platforms to evaluate remote care and assistive technologies in gerontology," in *Proceedings of 7th International Workshop on Enterprise networking and Computing in Healthcare Industry*, 2005, pp. 67–72.

[69] C. Chen and C. Pomalaza-Raez, "Design and Evaluation of a Wireless Body Sensor System for Smart Home Health Monitoring," in *IEEE Global Telecommunications Conference (GLOBECOM)*, 2009, pp. 1–6.

[70] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, and D. F. Ferguson, *Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging and More*. Prentice Hall PTR, 2005.

[71] P. Fremantle, S. Weerawarana, and R. Khalaf, "Enterprise Services," *Commun. ACM*, vol. 45, no. 10, pp. 77–82, ACM, Oct. 2002.

[72] A. Michlmayr, F. Rosenberg, C. Platzer, M. Treiber, and S. Dustdar, "Towards Recovering the Broken SOA Triangle: A Software Engineering Perspective," in *2nd International Workshop on Service Oriented Software Engineering*, ACM, 2007, IW-SOSWE '07, pp. 22–28.

[73] D. S. Linthicum, *Next Generation Application Integration: From Simple Information to Web Services*. Addison-Wesley Longman Publishing Co., Inc., 2003.

[74] B. M. Michelson, "Event-Driven Architecture Overview," *Patricia Seybold Group*, February 2006.

[75] J. McGovern, O. Sims, A. Jain, and M. Little, "Event-Driven Architecture,", in *Enterprise Service Oriented Architectures*, pp. 317–355, Springer Netherlands, 2006.

[76] H. Taylor, A. Yochem, L. Phillips, and F. Martinez, *Event-Driven Architecture: How SOA Enables the Real-time Enterprise*. Pearson Education, 2009.

[77] R. S. Aguilar-Savén, "Business process modelling: Review and framework," *International Journal of Production Economics*, vol. 90, no. 2, pp. 129–149, 2004.

[78] B. Scholz-Reiter and E. Stickel, *Business Process Modelling*. Springer Berlin Heidelberg, 1996.

[79] Thomas Chesney, "Business Process Modelling,", in *Competitive Information in Small Businesses*, pp. 47–67, Springer Netherlands, 2003.

[80] D. A. Chappell, *Enterprise Service Bus*. O'Reilly Media, Inc., 2009.

[81] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, *Web Services: Concepts, Architectures and Applications*. Springer Publishing Company, Incorporated, 1st edition, 2010.

[82] L. Richardson and S. Ruby, *RESTful Web Services*. O'Reilly, May 2007.

[83] R. T. Fielding, *Architectural Styles and the Design of Network-based Software Architectures*, PhD thesis, University of California, 2000.

[84] S. Vinoski, "Web Services Interaction Models. Part I: Current practice," *IEEE Internet Computing*, vol. 6, no. 3, pp. 89–91, 2002.

[85] R. T. Fielding and R. N. Taylor, "Principled Design of the Modern Web Architecture," *ACM Trans. Internet Technol.*, vol. 2, no. 2, pp. 115–150, ACM, 2002.

[86] S. Vinoski, "Serendipitous Reuse," *IEEE Internet Computing*, vol. 12, no. 1, pp. 84–87, 2008.

[87] Niroshinie Fernando, Seng W. Loke, and Wenny Rahayu, "Mobile cloud computing: A survey," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 84–106, 2013.

[88] M. Satyanarayanan, “Fundamental Challenges in Mobile Computing,” in *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, ACM, 1996, pp. 1–7.

[89] N. Vallina-Rodriguez and J. Crowcroft, “Energy Management Techniques in Modern Mobile Handsets,” *IEEE Communications Surveys Tutorials*, vol. 15, no. 1, pp. 179–198, 2013.

[90] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, “The Case for VM-Based Cloudlets in Mobile Computing,” *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[91] K. Kumar and Y.-H. Lu, “Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?,” *Computer*, vol. 43, no. 4, pp. 51–56, 2010.

[92] A. Khan, M. Othman, S. Madani, and S. Khan, “A Survey of Mobile Cloud Computing Application Models,” *IEEE Communications Surveys Tutorials*, vol. PP, no. 99, pp. 1–21, 2013.

[93] D. Huang, “Mobile Cloud Computing,” *IEEE COMSOC Multimedia Communications Technical Committee (MMTC) E-Letter*, vol. 6, no. 10, pp. 27–31, 2011.

[94] A. K. Dey, “Understanding and Using Context,” *Personal Ubiquitous Comput.*, vol. 5, no. 1, pp. 4–7, Springer-Verlag, 2001.

[95] B. N. Schilit and M. M. Theimer, “Disseminating Active Map Information to Mobile Hosts,” *IEEE Network*, vol. 8, no. 5, pp. 22–32, 1994.

[96] Cristiana Bolchini, Carlo A. Curino, Elisa Quintarelli, Fabio A. Schreiber, and Letizia Tanca, “A Data-oriented Survey of Context Models,” *SIGMOD Rec.*, vol. 36, no. 4, pp. 19–26, ACM, 2007.

[97] S. Najar, O. Saidani, M. Kirsch-Pinheiro, C. Souveyet, and S. Nurcan, “Semantic Representation of Context Models: A Framework for Analyzing and Understanding,” in *Proceedings of the 1st Workshop on Context, Information and Ontologies*, ACM, 2009, pp. 1–10.

[98] N. Houssos, A. Alonistioti, and L. Merakos, “Towards efficient support of context-awareness in mobile systems,” in *14th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2003, pp. 834–838.

[99] T. Strang and C. Linnhoff-Popien, "A Context Modeling Survey," in *Workshop on Advanced Context Modelling, Reasoning and Management*, 2004, pp. 1–8.

[100] Y. B. D. Trinugroho, K. Rasta, T. H. Nguyen, M. Gerdes, R. Fensli, and F. Reichert, "A Location-Independent Remote Health Monitoring System Utilising Enterprise Service Bus," *IADIS International Journal on WWW/Internet*, vol. 10, no. 2, pp. 88–106, 2012.

[101] D. Jordan, J. Evdemon, A. Alves, A. Arkin, S. Askary, C. Barreto, B. Bloch, F. Curbera, M. Ford, Y. Goland, A. Guízar, N. Kartha, C. K. Liu, R. Khalaf, D. König, M. Marin, V. Mehta, S. Thatte, D. van der Rijn, P. Yendluri, and A. Yiu, "Web Services Business Process Execution Language Version 2.0," *OASIS Standard*, April 2007.

[102] J. Pansiot, D. Stoyanov, D. McIlwraith, B. P. L. Lo, and G. Z. Yang, "Ambient and Wearable Sensor Fusion for Activity Recognition in Healthcare Monitoring Systems,", in *4th International Workshop on Wearable and Implantable Body Sensor Networks (BSN)*, vol. 13 of *IFMBE Proceedings*, pp. 208–212, Springer Berlin Heidelberg, 2007.

[103] W.-Y. Chung, S. Bhardwaj, A. Purwar, D.-S. Lee, and R. Myllylae, "A Fusion Health Monitoring Using ECG and Accelerometer sensors for Elderly Persons at Home," in *29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2007, pp. 3818–3821.

[104] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The Many Faces of Publish/Subscribe," *ACM Comput. Surv.*, vol. 35, no. 2, pp. 114–131, ACM, Jun. 2003.

[105] D. Trossen and D. Pavel, "Building a Ubiquitous Platform for Remote Sensing Using Smartphones," in *The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous)*, IEEE, 2005, pp. 485–489.

[106] A. Nadalin, C. Kaler, R. Monzillo, and P. Hallam-Baker, "Web Services Security: SOAP Message Security 1.0 (WS-Security 2004)," *Oasis Standard*, vol. 200401, March 2004.

[107] D. L. McGuinness and F. Van Harmelen, "OWL Web Ontology Language Overview," *World Wide Web Consortium (W3C) Recommendation*, February 2004.

[108] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML," *W3C Member Submission*, May 2004.

Part II

Appendix A

List of Publications

The author of this dissertation has published eight peer-reviewed scientific articles during the course of the PhD programme. The author is the main author in all articles. Five articles (paper I–V) are included in this dissertation to give the reader a quick overview of the author's work in a chronological order of publication.

Papers included in the dissertation

Paper I Y. B. D. Trinugroho, F. Reichert, and R. Fensli, "eHealth Smart Home Environment Service Platform: Enabling Remote Monitoring and Service Composition through Social Media," in *Proc. of 5th International Conference on Health Informatics (HEALTHINF)*, Vilamoura, Algarve, Portugal, 1-4 February 2012, pp. 434-438.

Paper II Y. B. D. Trinugroho, F. Reichert, and R. Fensli, "An Ontology-Enhanced SOA-Based Home Integration Platform for the Well-Being of Inhabitants," in *Proc. of IADIS International Conference e-Health 2012*, Lisbon, Portugal, 17-19 July 2012, pp. 159-164.

Paper III Y. B. D. Trinugroho, K. Rasta, T. H. Nguyen, M. Gerdes, R. Fensli, and F. Reichert, "A Location-Independent Remote Health Monitoring System Utilising Enterprise Service Bus," in *IADIS International Journal on WWW/Internet*, vol. 10, no. 2, December 2012, pp. 88-106.

This is an invited article, extended from paper VIII.

Paper IV Y. B. D. Trinugroho, M. Gerdes, M. M. M. Amjad, F. Reichert, and R. Fensli, "A REST-Based Publish/Subscribe Platform to Support Things-to-Services Communications," in *Proc. of 19th Asia-Pacific Conference on Communications (APCC)*, Bali, Indonesia, 29-31 August 2013, pp. 327-332.

Paper V Y. B. D. Trinugroho, "Information Integration Platform for Patient-Centric Healthcare Services: Design, Prototype, and Dependability Aspects," in *Future Internet*, vol. 6, no. 1, March 2014, pp. 126-154.

Papers not included in the dissertation

Paper VI Y. B. D. Trinugroho, F. Reichert, and R. W. Fensli, "A SOA-Based eHealth Service Platform in Smart Home Environment," in *Proc. of 13th IEEE International Conference on e-Health Networking, Applications and Services (HEALTHCOM)*, Columbia, MO, USA, 13-15 June 2011, pp. 201-204.

Paper VII Y. B. D. Trinugroho, R. Fensli, and F. Reichert, "Design Recommendations for a Reliable Body-Worn Patient Monitoring and Alarming Service," in *Proc. of 7th International Conference on Body Area Networks (BODYNETS)*, Oslo, Norway, 24-26 September 2012, pp. 135-138.

Paper VIII Y. B. D. Trinugroho, K. Rasta, T. H. Nguyen, R. Fensli, and F. Reichert, "A Real-Time Web-Based Health Monitoring System Based on Enterprise Service Bus," in *Proc. of IADIS International Conference WWW/Internet 2012*, Madrid, Spain, 18-21 October 2012, pp. 165-172.

This article received best paper award.

Appendix B

Paper I

Title:	eHealth Smart Home Environment Service Platform: Enabling Remote Monitoring and Service Composition through Social Media
Authors:	**Yohanes Baptista Dafferianto Trinugroho**, Frank Reichert, and Rune Fensli
Affiliation:	University of Agder, Faculty of Engineering and Science, Jon Lilletuns vei 9, 4879 Grimstad, Norway
Published in:	*Proceedings of 5th International Conference on Health Informatics (HEALTHINF)*, Vilamoura, Algarve, Portugal, 1-4 February 2012.

eHealth Smart Home Environment Service Platform: Enabling Remote Monitoring and Service Composition through Social Media

Yohanes Baptista Dafferianto Trinugroho, Frank Reichert, and Rune Fensli

ICT Department, Faculty of Engineering and Science, University of Agder, Grimstad, Norway

Abstract — **Demographic changes with the growth of elderly populations implies a need of developing efficient technology tools in order for the patients to stay in their own home for as long time as possible feeling safe and secured, and where the medical follow up can be achieved by remote home monitoring equipment. The lack of actual international standards has the consequence of proprietary solutions being used by vendors, making interoperability difficult. On the other hand, social media has become an integral part of modern society and has changed the way people interact with one another. It can potentially be extended to support well-being at home as well. In this paper we argue that social media have the potential to play an important role in remote home monitoring and remote service composition for provisioning health-care-related services in the future. An overview of a proposed platform based on Service-Oriented Architecture (SOA) paradigm for interoperability between different devices is also presented.**

Keywords—eHealth, Social media, Remote home monitoring, Interoperability, Service platform, SOA.

I. Introduction

Demographic changes in population with increased number of elderly people is a huge challenge for future health care services. The new generation elderly people are supposed to give new requirements to those services compared to the elderly generation today. In Norway, a public report used Mick Jagger as an icon representing the new generation elderly [1]. This generation is supposed to bring new challenging requirements to the future health care services mainly because they opposed the existing authorities in their younger days and are still a generation with an active lifestyle.

In order for this new generation elderly to be able to continue an independent living with high degree of self-care and self-management, new technology developments will compensate for their needs in daily living activities. This generation is known as the digital immigrants, compared to younger generations as digital natives, described by Prensky [2]. However, during their active working life, they have been trained and got used to new information and communications technology (ICT) solutions such as the Internet, email, Facebook, and smartphones. In their elderly days, it is reasonable to expect them to require updated technology solutions both for ambient assisted living purposes and for their electronic collaboration with healthcare service providers.

Their needs of new technologies for an active and independent lifestyle will give a future private market for technology solutions. However, in many countries the social welfare services will support the elderly person with actual need of assistive technologies. This paradox will have tremendous consequences to interoperable solutions for future eHealth solutions, in order for the elderly to have home-care installations being able to communicate with public healthcare services.

Today, there is a lack of common standards for Tele-home-care technology; those challenges are focused in the European project HITCH [3], still having a long way ahead trying to have acceptance for the use of international standards. Alternatively, the use of open standards and open source solutions can contribute to rapid development of interoperability in the home healthcare solutions. In this paper, we will highlight how open standards can be developed as a fundamental platform for future development. With the rapid growth of social media users around the globe [4], social media becomes a promising tool to be integrated into Tele-home-care services.

II. MOTIVATION

eHealth is defined as the use of ICT to access health and lifestyle information, to give support and improved health care services [5]. Tele-home-care is a specific instance of telehealth that focuses on providing remote healthcare services to patients in their homes. Healthcare personnel can use remote communication mechanisms either in consulting and cooperating with their colleagues or in advising and guiding their patients at home. Patients at home may need to interact with measurement devices which in turn will generate clinical information that can be analysed by healthcare personnel remotely. Several examples of measurements that can be acquired include heart rate, blood pressure, oxygen saturation, and weight. In addition, monitoring of activities of daily living (ADL) can be provided as well by making use of various different devices installed within the home environment [6]. These devices combined by the computer system being deployed transform the home environment to smart home.

Besides the elderly population growth and the decreasing number of healthcare workers, other important driving force for adopting Tele-home-care is cost [7]. Tele-home-care services are expected to reduce the cost of healthcare services in general, enabling a personalised treatment based on long-term personal records with final goal of improving quality of medical treatments. By employing Tele-home-care solutions, patients can reduce hospitalisation periods, reduce energy and carbon emissions generated by transportation means, as well as decrease the number of necessary hospital facilities. However, new devices are needed to be installed within the home environment to support the provisioning of Tele-home-care services. In addition, Internet connectivity and other backup communications channels (e.g. telephone network) are needed to be provisioned as well. These additional requirements may need a huge amount of initial set-up cost if everything is owned and managed by the inhabitants at home. To tackle this issue, different business models can be used by healthcare service providers, such as bundling the devices and services in a monthly subscription manner, so that the patients at home do not need to buy the devices. From this standpoint, different healthcare service providers can provide different types of healthcare services to the patients at home.

Various different devices available in a home environment, including medical devices, may not be utilised to their full potential without combining their capabilities with one another. For example, a smart carpet detects a falling patient. After five minutes if no movement is detected by motion sensors in that particular room, then the patient's private doctor, nurses, and relatives should be informed by means of text messaging and email. In addition, the main door of the house will be unlocked to enable first aid personnel to come in. To deploy this scenario, capabilities from several different devices should be combined (i.e. smart carpet for fall detection, motion sensors for movement detection, clock for timing, mobile phone for sending text messages, personal computer for sending emails, door actuator for locking and unlocking a door). In order to achieve this, capabilities from different devices should be exposed by means of standardised application programming interfaces (APIs) which form basic services in a home environment. These services will then act as building blocks for composite services [8].

Social interactions in Tele-home-care should not be limited to interactions between healthcare service providers and patients. Involvement of relatives and colleagues is crucial as well for encouragement of better lifestyle at home [9]. However, direct physical interactions may not always be possible due to location and timing constraints. Collaboration tools are needed to fulfil this requirement, and as the Internet plays a crucial role as a communications backbone in a smart home environment, social media is expected to contribute as a collaboration platform for virtual meetings [10]. Figure B.1 shows an overview of social media's role in Tele-home-care.

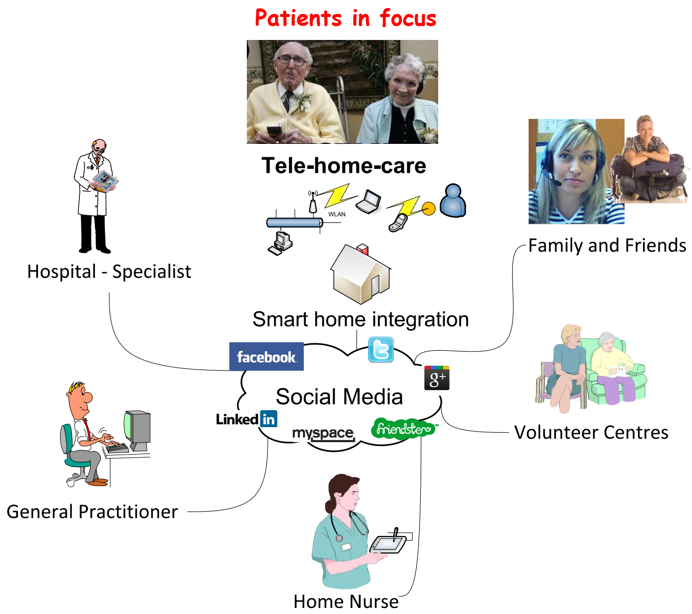


Figure B.1: Social Media and Tele-home-care

III. Social Media Services in Tele-Home-Care

Social media services can play different roles in Tele-home-care scenarios. Table B.1 shows a classification of possible Internet-based social media services for home health care [11].

Table B.1 classifies Internet-based social media services into three main categories: cognitive support systems, secured communities, and web-based services. All these three categories can be used in Tele-home-care systems, although the secured communities category may be used more often than the others. What may be missing in this classification is the usage of social media services to remotely compose services in the smart home environment. Taking the previous example of patient fall detection service that involves capabilities from different devices, what if the waiting period is to be changed from five minutes to ten minutes? Or what if the personnel being alarmed should be changed? What if a new device should be involved in the sequence such as turn on every single light in the house when the accident happened at night?

A home automation system should provide the possibility to be remotely administered. This can be accomplished by making use of the Internet, by providing a

Internet-based social media services	Cognitive support systems	Memo planner system
		Clock/calendar functions
		Time-dependent reminders
		Remote control systems
	Secured communities	Home healthcare information exchange
		Family and friends social care and information exchange
		Virtual meeting places for social activities
		Voluntary social services
	Web-based services	Internet communities
		Internet information search
		Entertainment
		Music & films/videos
		News and channels

Table B.1: Classification of Internet-based social media services for home health care

web-based home management system to remotely compose services from available devices installed in the corresponding home. This web-based tool can then also be used to remotely monitor the patient's conditions. This will enable healthcare personnel (e.g. a doctor) to remotely monitor patient as well as compose services from existing devices at home to fit best with the patient's situation and condition. However, this tool should be accessible not only by healthcare workers, but also relatives and colleagues of the patient. This will turn the web-based tool to a social media, where different users can monitor the patient's condition, communicate with one another, as well as review any change in deployed services at home. A web-based role-based access control (RBAC) is needed to be implemented in the system [12] so that not everyone can alter services being provisioned at home.

However, if the deployed web-based remote home monitoring and management system is a standalone dedicated application, colleagues and relatives of the patient may not access it frequently. In this case, it is beneficial if the web-based tool is integrated with available social media services such as Facebook. Instead of deploying the tool as a standalone system, a dedicated Facebook application can be developed for this purpose. When integrating the use of unsecured social media in remote home care services, this implies security and juridical issues not clearly defined [13].

In order to incorporate the remote access, a service platform for integrating capabilities of various devices at home is needed to be deployed in the smart home environment. In the next section we will discuss about such platform.

IV. Home eHealth Service Platform Solution

Healthcare service providers can collocate different devices in a smart home environment to provide specific health-related services. These devices' capabilities, combined with one another, can provide various composite services, which can lead to better personalised services to the patient at home. However, different vendors being chosen for these devices may raise incompatibility issues between them. To avoid building services from scratch each time a new service is needed, service-oriented architecture (SOA) paradigm is promising to be adopted for a home eHealth service platform, as it promotes interoperability, modularity, and reusability of service components.

To further reduce the involvement of the patient in accomplishing a service's aim, a context-aware home automation system is also needed. A context-aware home automation system should be able to gather the patient's information and the surrounding environment, then adjust the environment setting to suit the patient's needs and preferences. Such context-aware home integration platform was proposed in authors' previous work [14], shown in Figure B.2.

The proposed platform exposes each device's capabilities as reusable services, termed as service enablers, that can be used to compose more complex services.

In order for a home automation system to be able to adapt the surrounding environment, a knowledge base should exist within the home environment, storing knowledge about different entities' situation, including the patient's. Reasoning can then take place on top of the knowledge base. Pure automated reasoning may not be desirable for Tele-home-care services as the system may infer false condition of the patient. On the other hand, healthcare specialist may know better what is best for the patient. Thus, a static rule-based reasoning mechanism may fit well, where the rules can be updated remotely by qualified personnel.

The proposed platform in Figure B.2 is quite generic in the sense that it may accommodate different technologies. Since the application for remote home monitoring and management system is web-based, standardised Web Services technology will suit best for implementation purpose. This means that the capabilities of each device are exposed as Web Services end-points. Then web-based client application (e.g. dedicated Facebook application) can be developed for remote monitoring and management. Figure B.3 shows the high-level overview of the proposed system.

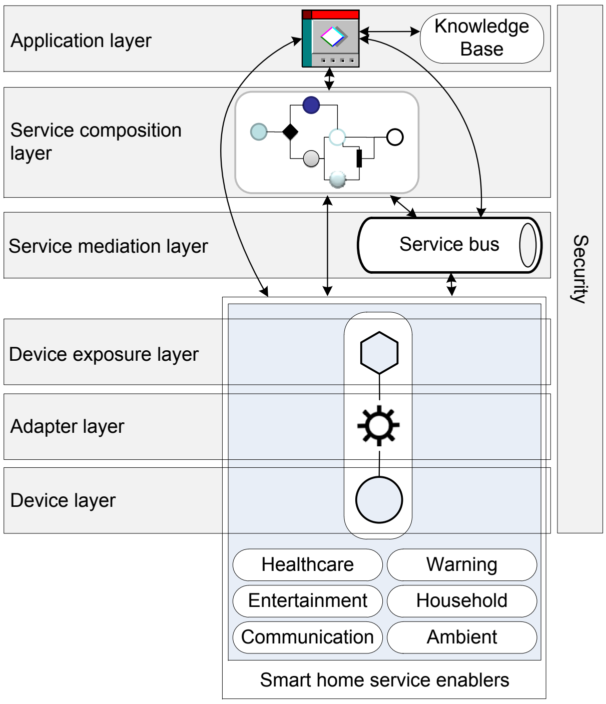


Figure B.2: Home eHealth Service Platform

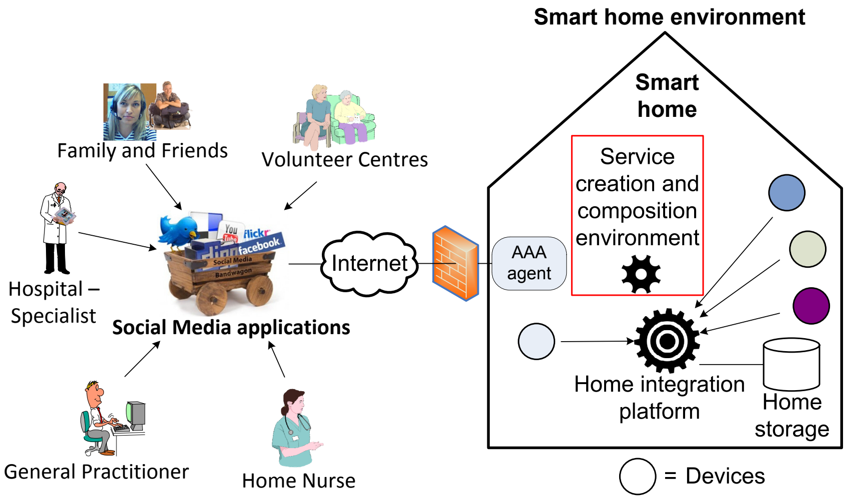


Figure B.3: Remote home monitoring and management through social media applications

By deploying this idea, social media applications can act as a web-based remote home monitoring and management tool, with three main functions:

1. Remote monitoring of patient's condition as well as the surrounding environment.

2. Remote service composition for new services to meet patient's needs. This

action should be conducted by healthcare specialists.

3. Remote rule creation for automated reasoning on the knowledge base. This action should also be conducted by healthcare personnel.

Authentication, authorisation, and accounting (AAA) agent is required to be present in the smart home environment especially for security measure. Only authorised external applications (e.g. Facebook application) and authorised users may access the service creation environment within the smart home environment. And since all data are transmitted through the Internet, Hyper Text Transfer Protocol Secure (HTTPS) protocol is suggested to be used for secure transmission.

A crucial issue needed to be taken into account during the development of the web-based social media application is the user-friendliness of the graphical user interface (GUI). Icons or puzzles being used to represent the capabilities of devices should be clear enough for regular users. Message flow of a service should also be easily visible, and should be easily modified.

V. Conclusions

Social media's user base growth is intriguing, as a direct consequence of fast penetration of Internet usage in modern society. This trend can be seen as a positive reality as it enables remote collaboration between individuals, breaking time and space barriers. In this paper we argue that social media applications can be further pushed to provide remote home monitoring and management services, including remote service composition, for patients living at home. Security measure is suggested to include current best-practice approaches using AAA agent at home and HTTPS transmission protocol.

References

[1] Teknologirådet, "Future Aging and New Technology," Teknologirådet, 2009.

[2] M. Prensky, "Digital natives, digital immigrants Part 1," On the Horizon, vol. 9, no. 5, pp. 1-6, 2001.

[3] HITCH. "Healthcare Interoperability Testing and Conformance Harmonisation", [Online]. Available: `http://www.hitch-project.eu/` [Accessed 10 Oct 2011].

[4] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan, "Group Formation in Large Social Networks: Membership, Growth, and Evolution," in *Proc. of*

12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 2006, pp. 44-54.

[5] European-Commission, "What is eHealth," [Online]. Available: `http://ec.europa.eu/information_society/activities/health/whatis_ehealth/index_en.htm` [Accessed 10 Oct 2011].

[6] X. H. B. Le, M. Di Mascolo, A. Gouin, and N. Noury, "Health Smart Home for elders – A tool for automatic recognition of activities of daily living," in *Proc. of 30th IEEE EMBS Annual International Conference (EMBC),* Vancouver, British Columbia, Canada, August 2008, pp. 3316-3319.

[7] C. Menkens and W. Kurschl, "VoIP Based Telehomecare Application Kiosk," in *Proc. of 7th International Conference on Information Technology: New Generations (ITNG),* Las Vegas, NV, USA, April 2010, pp. 783-790.

[8] Y. B. D. Trinugroho, F. Reichert, and R. W. Fensli, "A SOA-Based eHealth Service Platform in Smart Home Environment," in *Proc. of 13th IEEE International Conference on e-Health Networking, Applications and Services (HEALTHCOM),* Columbia, MO, USA, June 2011, pp. 201-204.

[9] N. A. Abdullah and N. Zakaria, "Sociability aspects in e-health community: A Review," in *Proc. of International Symposium in Information Technology (ITSim),* Kuala Lumpur, Malaysia, June 2010, pp. 972-976.

[10] E. Thygesen, M. M. F. Fensli, R. Skaar, H. I. Svareid, Y. Li, and R. Fensli, "User requirements for a Personalized Electronic Community for Elderly People with Risk of Marginalization," in *Proc. of 9th Scandinavian Conference on Health Informatics (SHI),* Oslo, Norway, August 2011, pp. 50-54.

[11] B. Dale, J. G. Dale, M. M. F. Fensli, and R. W. Fensli, "Omsorg og teknologi: i dag og i morgen," Utdanning til omsorg i fortid ntid og fremtid, pp. 180-197, 2010.

[12] X. Chungen, Y. Han, and L. Fengyu, "The implementation of role-based access control on the Web," in *Proc. of International Conferences on Info-tech and Info-net (ICII),* Beijing, China, November 2001, pp. 251-255.

[13] J. B. Williams and J. H. Weber-Jahnke, "Social Networks for Health Care: Addressing Regulatory Gaps with Privacy-by-Design," in *Proc. of 8th Annual International Conference on Privacy, Security and Trust (PST),* Ottawa, ON, Canada, August 2010, pp. 134-143.

[14] Y. B. D. Trinugroho, F. Reichert and R. Fensli, "An Ontology-Enhanced SOA-Based Home Integration Platform for the Well-Being of Inhabitants," in *Proc. of 2nd International Conference on Pervasive Computing, Signal Processing and Applications (PCSPA),* Gjøvik, Norway, September 2011.

Appendix C

Paper II

Title:	An Ontology-Enhanced SOA-Based Home Integration Platform for the Well-Being of Inhabitants
Authors:	**Yohanes Baptista Dafferianto Trinugroho**, Frank Reichert, and Rune Fensli
Affiliation:	University of Agder, Faculty of Engineering and Science, Jon Lilletuns vei 9, 4879 Grimstad, Norway
Published in:	*Proceedings of IADIS International Conference e-Health 2012*, Lisbon, Portugal, 17-19 July 2012.

An Ontology-Enhanced SOA-Based Home Integration Platform for the Well-Being of Inhabitants

Yohanes Baptista Dafferianto Trinugroho, Frank Reichert and Rune Fensli
Faculty of Engineering and Science, University of Agder – Jon Lilletuns vei 9, 4879 Grimstad, Norway

Abstract — **Smart homes are expected to provide better services to inhabitants, supporting independence especially to elderly people in living their lives. Home automation system plays an important role in supporting the well-being of inhabitants, reducing necessary human interventions in achieving different tasks. Sensors, actuators, and various devices are required to be installed in smart homes to provision context-aware services. This paper presents a home integration platform architecture based on Service-Oriented Architecture paradigm that can be used to integrate the functionalities of different devices. A proposed ontology to enhance the automated reasoning process is also presented, and several example scenarios related to safety and security at home utilising the Semantic Web Rule Language rules are described as well.**

Keywords—eHealth, smart home, SOA, Web Services, integration platform, ontology

I. INTRODUCTION

The role of Information and Communications Technology (ICT) is becoming more crucial in home environments, especially for supporting elderly inhabitants. Home automation system should be able to control various devices installed at home, turning the aforementioned homes to so-called smart homes. Smart homes are also envisaged to enable healthcare providers to provide remote patient monitoring and care, which can potentially reduce the required time for patients to stay at the healthcare premises. From this standpoint, smart homes will provide more independence to elderly people in living their lives with minimum interventions.

Integrating various different devices (including sensors and actuators) in a smart home can be challenging, especially when different vendors are involved in the process. Redundant and overlapping functionalities between different devices may exist with very limited reusability in different services being provided to the inhabitants. To solve this issue, a service-oriented approach is proposed to promote reusability and interoperability, as its loosely coupled nature allows integration of legacy and existing systems in granular way that can easily accommodate changing needs. The adoption of Service-Oriented Architecture (SOA) [1] paradigm in a smart home will provide a unified way of combining and using data from various devices, providing a standard way to develop and compose services.

As automated decision making is of great importance in a smart home environment, a knowledge base containing real-time information and context of different entities in the smart home should be present. This knowledge base has to be updated whenever any condition of the entities is changed, then reasoned upon by the home automation system to make correct decisions. Ontology is an approach for storing and managing knowledge base that can be used in a smart home environment. It can also be used to model and store contexts.

This paper, which is based on an ongoing research project, proposes a SOA-based home integration platform architecture that is enhanced by incorporating an ontology for modelling contexts of different entities in a smart home environment. Some scenarios related to safety and security of inhabitants are also discussed with corresponding rules for automated reasoning.

II. Towards Service-Oriented Paradigm

SOA has been widely adopted until recently, especially using Web Services [1] technology. Web Services technology promotes interoperability between various software applications running on disparate platforms by employing open standards and protocols. In addition, it also enables the reuse of services and components which further increases the speed of service creation. In general there are three different entities in SOA: service provider, service consumer, and service registry. Service provider provides services and publishes their interfaces as well as access information to the service registry. Service consumer locates entries in the service registry and binds to the service provider in order to invoke services.

SOA is envisaged to give a significant impact when applied to eHealth services in smart home context for the well being of inhabitants as it is well suited to tackle interoperability issues by separating implementation logic and interface of a service. Furthermore, since the SOA concept promotes reusability of existing services, new and tailor-made healthcare services can be provisioned in a timely manner.

As eHealth services becoming more pervasive, medical sensors alongside other devices are required to be installed in the smart home to deliver eHealth-related services. These devices may not be owned by the inhabitants, but rather provided by healthcare service providers (e.g. hospitals) with a wider scope of business processes. A device placed by a healthcare service provider in a smart home generally serves a specific purpose. This, however, limits the full potential of the corresponding device as the data produced or captured by the device can potentially be utilised by other services in the smart home. A common interaction platform between devices is necessary to be present in the smart home to tackle this issue.

Point-to-point communication between service provider and service consumer is mainly used in traditional Web Services approach. This may work well when a small number of devices are present. With an increasing number of new devices being installed in the smart home, a centralised management system is beneficial to be deployed. This can be achieved by introducing a logical smart home service bus that supports event-driven SOA. A technology called Enterprise Service Bus (ESB) [2] existed within the SOA domain that can be used for this purpose. An ESB provides the implementation backbone for an SOA, acting as a hub between service provider and service consumer. It provides a loosely coupled, event-driven SOA with a highly distributed universe of named routing destinations across a multi-protocol message bus. Applications in the ESB are abstractly decoupled from each other, and connected together through the bus as logical endpoints that are exposed as event-driven services. In general an ESB has four major functions: message routing, message transformation, protocol mediation, and event handling.

III. Context-Awareness in Smart Home Environment

Context is any information that characterise an entity's situation [3], where an entity can be a person, place, or any object that plays a role in any type of interaction. Although the definition of context within computer science communities rooted in user location, context encompasses many other things of interest including physical surrounding environments (e.g. lighting, noise level, temperature), network connectivity, communication costs, communication bandwidth, and social situation. Context-aware computing is about gathering user information and their environment, where such information is used for adjusting environment settings to suit user needs and preferences. Context-aware applications and services are most suitable to be applied in spaces where humans spend the majority of their time, including homes. Context-awareness is a prerequisite for adaptivity [4], and when applied in a smart home environment, it can potentially release the inhabitants from doing different tasks and support their well being at home.

Acquiring context is a starting point for any context-aware system. Context acquisition is the process where the real situation in the world is captured, the significant features are assessed, and an abstract representation is created, which is then provided to components in the system for further use. In a smart home environment, contextual information is mainly gathered by various sensors. In order to support home automation system, the context-aware system architecture should enable integration of different sensors with other devices. Thus, a home integration platform that is context-aware is required.

IV. Home Integration Platform Architecture

Integration of different devices and services in smart home environment is necessary in order to enable collaboration among them. By utilising the SOA principles, services beyond the basic utilisation of each device or service can be provisioned. To achieve this, a logical home integration platform is needed, acting as a logical central hub for all communications among devices and services. This platform should provide service creation and composition capabilities as well as event-driven message handling functionalities. Additionally, the platform should also follow standard-based interoperability best practices by utilising open standards to further avoid vendor lock-in. Figure C.1 shows the proposed architecture of the home integration platform, which is extended from the authors' previous work in [5].

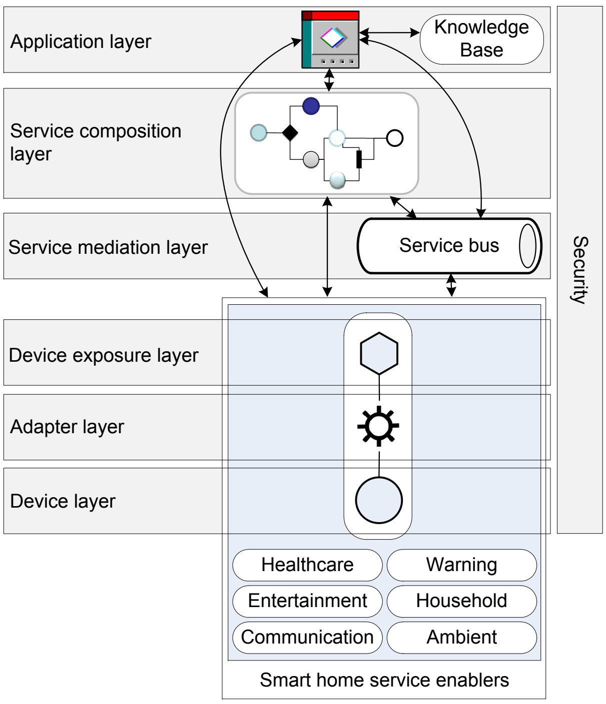


Figure C.1: Home integration platform architecture

Application developers are given the freedom to either use service enablers per se, combined the application with composite services, or rely on the mediation layer's specific functionalities such as the mediation flow for sequencing the invocation of services. By deploying this architecture, it is envisaged that integration of various devices and services will be easier in the smart home, and new services can be created faster to meet the inhabitants' needs.

Initial prototype of the architecture in Figure C.1 has been implemented within remote health monitoring application domain. Only three different types of information have currently been integrated: location, SpO2, and pulse rate. SpO2 and pulse rate information are gathered from Nonin Onyx II 9560 device with Bluegiga AP3201 as its gateway. Location information is gathered from an Android phone application deployed on HTC Desire. Mule open source ESB and Telenor Objects' Shepherd Platform are used as the service bus to provide event-driven, publish/subscribe messaging pattern. Web services interfaces (SOAP and REST) are employed for data input interfaces to the service bus. A web-based application based on Ajax and Java servlet technologies has been developed to visualise the remote health monitoring dashboard. The preliminary prototype implementation architecture is shown in Figure C.2.

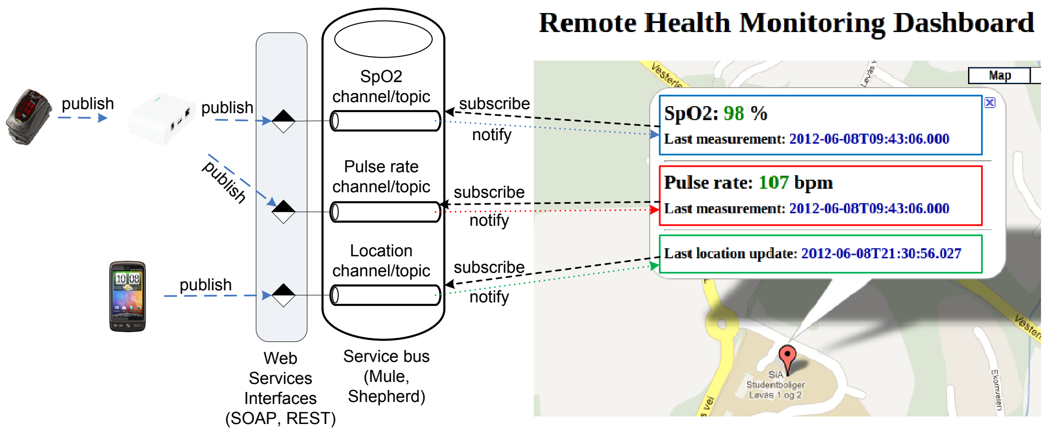


Figure C.2: Remote health monitoring prototype

V. ONTOLOGY-BASED AUTOMATED REASONING

An ontology has been developed within the Application layer in Figure C.1, acting as a knowledge base to describe relationships between different entities in the smart home environment. The ontology itself is developed following the Web Ontology Language (OWL) [6] standard representation. Semantic Web Rule Language (SWRL) [7] is used for reasoning on the ontology, where the rules are of the form of an implication between an antecedent and a consequent. The ontology, combined with SWRL rules, is aimed to support context-aware applications and services in the smart home to make correct decisions.

V.1 Smart Home Ontology

The well-being of inhabitants in a smart home setting is dependent on both the inhabitants' personal and surrounding ambient conditions. The developed ontology should cover both person-centric and ambient smart home context modelling. The current contexts being modelled in the ontology include activities of the inhabitants, the personal state of the inhabitants which covers both physical and mental states, location of the inhabitants, and the surrounding ambient smart home states. In addition to modelling contexts, the ontology also provides knowledge base for devices in the smart home. Figure C.3 shows several important classes (concepts) of the proposed smart home ontology.

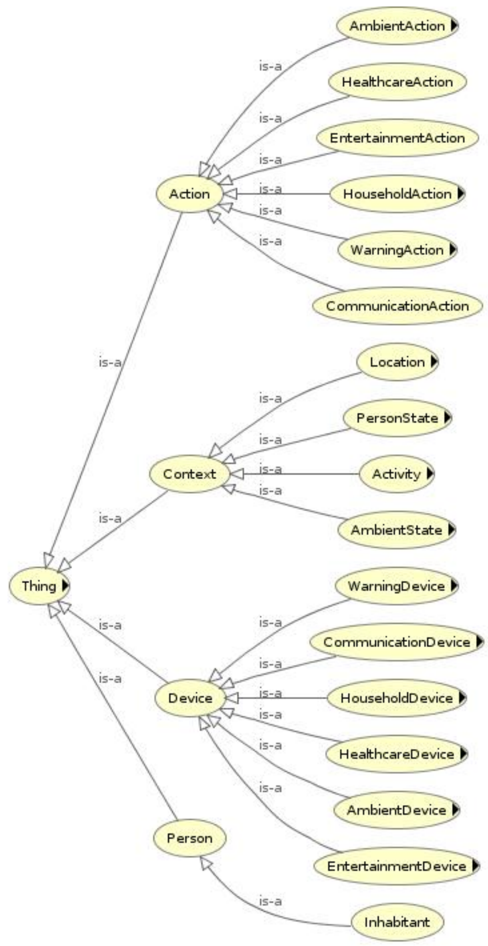


Figure C.3: Smart home ontology

The Device class has six subclasses that map one-to-one with the service enablers in the home integration platform architecture (Figure C.1). This particular class enables the underlying devices to take part in the reasoning process. The behaviour of the devices is represented by the Action class, which is also mapped one-to-one with the Device class, related by different object properties. Individuals (instances) of the Action class (and its subclasses) represent the possible actions of the underlying devices. The changing values of datatype properties of the Action class (and its subclasses) resulting from the reasoning process indicates that specific actions should be performed on the physical devices. The context-aware application has to be informed about these changes so that service functionalities of the corresponding devices can be invoked. After a successful invocation, the changed value of a datatype property should be reverted. From the SWRL rules' perspective, the value changes of datatype properties of the Action class' individuals act as consequents of the rules, where the antecedent is mainly played by the contextual information provided by various sensors. Several SWRL rules are provided in the scenario examples. Figure C.4 shows the general context-aware application architecture. The context-aware application subscribes to all sensor-related information to the service bus, and listens to all notifications coming from the service bus whenever new sensor-related information arrives. It then updates the ontology and runs the SWRL rules. If an action should be carried out, the context-aware application invokes the corresponding service in the action space category. The services within the action space category can be actuators as well as external services such as social media APIs.

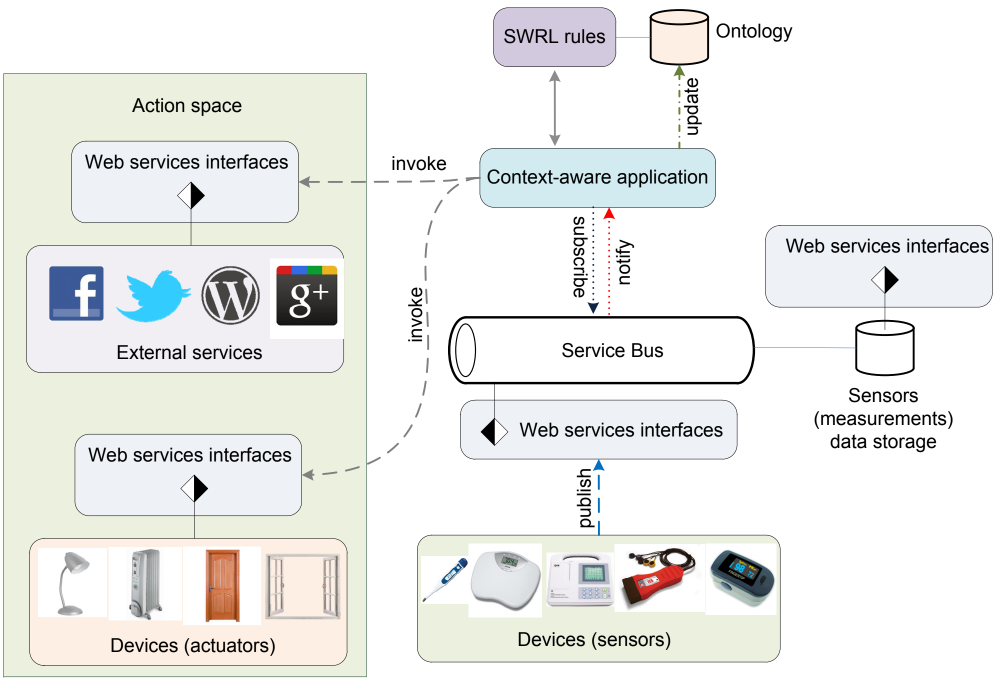


Figure C.4: Context-aware application architecture for home automation system

V.2 Example Scenarios

Safety and security of inhabitants in smart home are of great importance. Home automation, which is an intrinsic part of smart home, can strengthen safety and security aspects of smart home. Several simple scenarios have been devised relating to those aspects, described as follows.

- *Automatic Door Locking*
  An inhabitant leaves his house, gets on his car, and drives away. However, he forgot to lock the house's main entrance door. The home automation system detects this situation and makes a correct decision (i.e. locking the main entrance door). A rule to simulate this scenario is as follows.

  ```
  Person(?P) ∧ MainEntrance(?E) ∧ SmartHome(?S) ∧ Door(?D)
  ∧ hasRoom(?S,?E) ∧ hasRoomComponent(?E,?D) ∧ is-
  LocatedAt(?P,?L) ∧ isTransportation(?L,True) ∧ is-
  Locked(?D,False) ∧ hasDoorAction(?D,?A) → lock(?A,True)
  ∧ unlock(?A,False)
  ```

- *Automatic Window Opening*
  An inhabitant sets in his profile that a comfort temperature for a living room is between 19 and 25 degrees Celcius. A living room is considered to be hot when the temperature is above 25 degrees Celcius. The home automation system detects the temperature of the living room with a thermometer and change the temperature state to hot whenever the thermometer's temperature value exceeds 25 degrees Celcius. A rule to simulate the knowledge base update in the ontology of the changing temperature state is as follows.

  ```
  Room(?R) ∧ hasTemperature(?R,?T) ∧ hasTemperatureSen-
  sor(?R,?S) ∧ hasTemperatureValue(?S,?V) ∧ hasMaxCom-
  fortTemperature(?R,?C) ∧ swrlb:greaterThan(?V,?C) →
  hasTemperatureValue(?T,?V) ∧ hasColdnessState(?T,"Hot")
  ```

  This condition can be further used to control actuators, such as opening the window when the temperature state of the living room is hot. The rule for this task is as follows.

  ```
  Room(?R) ∧ hasTemperature(?R,?T) ∧ hasCold-
  nessState(?T,"Hot") ∧ Window(?W) ∧ hasRoomCompo-
  nent(?R,?W) ∧ hasWindowAction(?W,?A) ∧ isClosed(?W,True)
  → open(?A,True) ∧ close(?A,False)
  ```

- *Automatic Electric Stove Turning Off*
  An inhabitant sets the standard maximum time for taking a shower in the bathroom in his profile to 60 minutes. Given that each room in the smart home

(including the bathroom) is equipped with several infrared sensors for detecting the inhabitant's presence, the home automation system can detect the most current location of the inhabitant as well as the duration of his presence in that particular room. The inhabitant cooked food using an electric stove, then left the kitchen for taking a shower. 60 minutes has passed and he has not left the bathroom yet. The home automation system detects this anomaly, turns off the electric stove, then activates an alarming system in the bathroom for a notification to the inhabitant that he has been taking a shower more than the maximum normal duration and the electric stove has been turned off. A rule to simulate this scenario is as follows.

```
Person(?P) ∧ BathRoom(?B) ∧ SmartHome(?S) ∧ has-
Room(?S,?B) ∧ isLocatedAt(?P,?B) ∧ hasTimeOfStay-
InAPlaceMinutes(?P,?T) ∧ hasMaxNormalDurationMin-
utes(?B,?X) ∧ swrlb:greaterThan(?T,?X) ∧ Electric-
Stove(?V) ∧ isOff(?V,False) ∧ hasElectricStoveAc-
tion(?V,?O) ∧ ElectricAlarm(?W) ∧ hasDevice(?B,?W)
∧ hasAlarmAction(?W,?A) → turnOff(?O,True) ∧
turnOn(?O,False) ∧ setVolumeLevel(?A,"Low") ∧ setWarn-
ingMessage(?A,"Kitchen stove turned off!") ∧ start-
Warning(?A,True) ∧ stopWarning(?A,False) ∧ hasAlert-
State(?P,"Yellow")
```

As previously mentioned, the antecedent of the rules (subsequent to the implication symbol) indicates actions needed to be taken by the home automation system application. Thus, the application should be notified of these changes after a reasoning process takes place. The presented rules have been tested with Jess rule engine [8].

VI. Conclusions and Future Work

Interoperability, reusability, and modularity are some of the positive traits of SOA that are expected to give positive impacts when applied to home automation systems. In this paper a SOA-based home integration platform to support context-aware services was proposed. An ontology which describes the relationships between different entities in the smart home was also presented, and several reasoning scenarios related to safety and security of inhabitants using the SWRL rules were described. The deployment of the proposed home integration platform, combined with the ontology, is foreseen to enable the creation and deployment of sophisticated composite services for well-being of inhabitants in the smart home.

Implementation of context-aware applications utilising the developed ontology and various devices is planned to be carried out in the advancement of this work.

REFERENCES

[1] D. Barry, "Web Services and Service-Oriented Architecture: The Savvy Manager's Guide," Morgan Kaufmann Pub, 2003.

[2] D. Chappell, "Enterprise Service Bus," O'Reilly Media, Inc., 2004.

[3] A. K. Dey, "Understanding and Using Context," *Personal and Ubiquitous Computing*, vol. 5, no. 1, pp. 4-7, February 2001.

[4] B. Schilit, N. Adams, and R. Want, "Context-Aware Computing Applications," in *Proc. of 1st Workshop on Mobile Computing Systems and Applications (WMCSA),* Santa Cruz, CA, USA, December 1994, pp. 85-90.

[5] Y. B. D. Trinugroho, F. Reichert, and R. W. Fensli, "A SOA-Based eHealth Service Platform in Smart Home Environment," in *Proc. of 13th IEEE International Conference on e-Health Networking, Applications and Services (HEALTHCOM),* Columbia, MO, USA, June 2011, pp. 201-204.

[6] D. L. McGuinness and F. Van Harmelen, "OWL Web Ontology Language Overview," W3C Recommendation, February 2004.

[7] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML," W3C Member Submission, May 2004.

[8] E. Friedman-Hill, "Jess, The Rule Engine for the Java Platform," Sandia National Laboratories, 2003.

Appendix D

Paper III

Title:	A Location-Independent Remote Health Monitoring System Utilising Enterprise Service Bus
Authors:	**Yohanes Baptista Dafferianto Trinugroho**, Kamyar Rasta, Trinh Hoang Nguyen, Martin Gerdes, Rune Fensli, and Frank Reichert
Affiliation:	University of Agder, Faculty of Engineering and Science, Jon Lilletuns vei 9, 4879 Grimstad, Norway
Published in:	*IADIS International Journal on WWW/Internet*, vol. 10, no. 2, December 2012.

A Location-Independent Remote Health Monitoring System Utilising Enterprise Service Bus

Yohanes Baptista Dafferianto Trinugroho, Kamyar Rasta, Trinh Hoang Nguyen, Martin Gerdes, Rune Fensli and Frank Reichert
Faculty of Engineering and Science, University of Agder, Jon Lilletuns vei 9, 4879 Grimstad, Norway

Abstract — **Telehealth has been widely used in recent years to provide healthcare services at a distance. It supports eliminating space barriers and improves access to healthcare services, as well as alleviates cost burden of healthcare services by moving non-urgent treatments from healthcare premises to patients' homes. This is currently feasible to achieve by employing portable on-body wireless sensors for reading vital signs within Wireless Body Area Networks, and Internet technologies. This information is then forwarded to a back-end server, and responsible personnel (doctors, nurses, family members etc.) are notified in a real-time manner. However, patients may leave their homes for some walks and their vital information is still needed to be monitored remotely by their care providers. Thus, indoor-outdoor scenarios should also be considered by the end-to-end system. This paper presents an architecture for a location-independent remote health monitoring system by utilising Enterprise Service Bus for centralised data integration from different data sources. A prototype, which has been implemented as a proof-of-concept, is also presented.**

Keywords—Health monitoring, Enterprise Service Bus, SOA, publish/subscribe, real-time, web, context-awareness.

I. Introduction

Advancements in medical science have achieved unforeseen improvements in healthcare quality. These quality improvements, coupled with a decreasing birthrate, have contributed to an increasing average age of population in many developed countries. Statistics Norway, for example, reported that around 625,000 people older than 67 years old lived in Norway in 2010, and the figure is expected to have doubled by 2060 [1]. This situation, however, is not balanced by an increasing number of healthcare workforce, which widens the gap between demand and supply within the healthcare sector [2]. On the other hand, healthcare cost has been increasing steadily each year. In the United States, the total healthcare expenditure was around $2.6 trillion in 2010 alone, and the amount is expected to double within five years [3]. Early detection and preventive care are potential solutions to lower the cost pressure of healthcare services. Telehealth provides capabilities to assist health maintenance and emergency detection by utilising Information and Communications Technology (ICT), eliminating distance barrier between healthcare providers and patients [4]. In addition, telehealth supports remote alarming services as well which enables healthcare personnel to be notified whenever emergency situation occurred to the distantly-located patient.

Remote health monitoring is one of the most important applications of telehealth as it provides measurements and reports of patients' activities and health conditions to distantly-located healthcare providers. It enables healthcare personnel (e.g. doctors, nurses) as well as family members and relatives to remotely monitor elderly people and patients with special needs in their daily activities. This will provide the patients with more independence in living their lives within their own, well known facilities, with minimum physical intervention from others, and saving additional costs for moving into special care facilities. Remote health monitoring is made possible by utilising different devices/sensors in the patient's surroundings, usually deployed in a smarthome environment. Most of these devices are worn by the patient, especially the ones which measure vital information. However, assuming a patient to always stay at home is not very realistic, especially for those with good mobility condition. On the other hand, promoting elderly people to have a short walk for a couple of hours a day may have a positive impact on their health. Thus, the health monitoring system should also enable patient monitoring in outdoor environments. Advancements in wireless near field communications technologies have enabled devices and sensors around the patient's body to communicate without wires, forming so-called Wireless Body Area Networks (WBAN). Although the measured vital signs information may not be as thorough as in a smarthome environment, the outdoor health monitoring system is at least able to inform responsible personnel about the patient's latest health condition. The pervasiveness of Internet connectivity in recent years has enabled such information to be transmitted to a centralised monitoring server in a real-time manner.

There are some related previous initiatives within the area of remote health monitoring which are interesting to consider. Authors in [5] proposed a product line generic architecture with an example application of real-time health monitoring using a wireless sensor connected to a central station by means of a smart phone, employing a Service-Oriented Architecture (SOA) paradigm. However, options for message flow control from different data sources (e.g. sensors) and indoor-outdoor scenario are not highlighted. A SOA framework for WBAN-based patient monitoring was proposed in [6], where sensors are coordinated by a node which is responsible to retransmit the signals to a remote central monitoring unit. The proposed middleware makes use of web services, but how the messages being handled within the back-end monitoring server to the client side terminal is not described. Authors in [7] described a remote health monitoring system by employing ZigBee protocol to transfer all data output from medical devices to a GPRS gateway, which then forwards them to a healthcare centre for further analysis. How this data is being reported to healthcare personnel is not presented in detail as the main focus is on the usage of ZigBee protocol for gathering the data from sensors. A smartphone-based healthcare system providing real-time continuous monitoring of health conditions was proposed in [8]. Several sensors are included in the implementation, but how healthcare personnel are being notified is not clearly described.

This paper presents an architecture for a real-time web-based health monitoring system which uses Enterprise Service Bus as messaging backbone in the health monitoring server to support an event-driven publish/subscribe messaging model. A prototype implementation is also presented.

The rest of this paper is organised as follows. Technological concepts that are used in this paper are described in section II. Section III describes the design of our proposed remote health monitoring system. A prototype based on the system design which encompasses smartphone and server applications as well as ontology based modelling are described in section IV. Section V discusses some important aspects of a remote health monitoring system and improvements that can be added to the current implemented prototype. Conclusions are drawn in section VI alongside some future work directions that are planned to be conducted.

II. TECHNOLOGICAL CONCEPTS

II.1 Towards Service-Oriented Paradigm

SOA has become a major trend in the last couple of years as a way to support business processes of organisations, especially by using web service technology [9]. Web service technology promotes interoperability between various software applications running on disparate platforms by employing open standards and protocols. In addition, it also supports reuse of services and components which further increases the speed of service creation.

As SOA has been proved well suited for tackling interoperability issues by separating implementation logic from the interface of a service, SOA is envisaged to have significant impact when applied to eHealth services in smarthome context and beyond. Furthermore, since the SOA concept promotes reusability of existing services, new and tailor-made healthcare services can be provisioned in a timely manner. In a smarthome environment, sensors and actuators play vital roles for monitoring and controlling home conditions, tailored to meet the inhabitants' needs. With eHealth services becoming more pervasive, medical sensors are required to be installed and used in the smarthome to deliver eHealth services, including remote health monitoring. Most of these devices may not be owned by the residents of the home, but rather provided by healthcare service providers (e.g. hospitals) with a wider scope of business processes. A device placed by a healthcare service provider in a smarthome usually serves a specific purpose (i.e. a service to the user). This, however, limits the full potential of the corresponding device as the data produced or captured by the device can potentially be used by other services in the smarthome as well. A common data collection mechanism is required, and further, a common integration platform between devices is necessary to be present in the smarthome, which leads to the notion of Data as a Service (DaaS). A technology called Enterprise Service Bus (ESB) [10] exists within the SOA domain that can be used for this purpose. ESB provides the implementation backbone for SOA, acting as a hub between service providers and service consumers.

II.2 Enterprise Service Bus as Messaging Backbone

SOA promotes interoperability between different software applications running on disparate systems by employing open standards and protocols, separating implementation logic and interface of a service. Point-to-point communication between service providers and service consumers is mainly used in a traditional web service approach. This may work well when a small number of devices are present. With an increasing number of new devices being used, a centralised management system is beneficial to be deployed. This can be achieved by introducing a logical service bus that supports event-driven SOA. Combining the SOA paradigm with event-driven processing lays the foundation for an emerging technology that amalgamates various conventional distributed computing, middleware, Business Process Modelling (BPM) and Enterprise Application Integration (EAI) technologies. It offers a unified backbone on top of which enterprise services can be advertised, composed, planned, executed, monitored, and decommissioned [11]. This messaging implementation backbone is referred to as ESB. It acts as a loosely coupled, event-driven SOA with a highly distributed universe of named routing destinations across a multi-protocol message bus. Unlike a traditional web service point-to-point approach which could not avoid tight coupling between service consumers and service providers of the messages being exchanged, ESB provides a centralised approach for integration tasks, avoiding direct contacts between communicating services. Applications are abstractly decoupled from each other, and connected together through ESB as logical endpoints that are exposed as event-driven services. In general, ESB has four major functions: message routing, message transformation, protocol mediation, and event handling [10]. ESB supports various message exchange patterns, and one of the most important is publish/subscribe model, where applications can subscribe to a specific topic and notified whenever a data provider for that particular topic publishes a message for that topic.

Within healthcare arena, ESB is mainly used to integrate different health information systems from various healthcare providers and vendors. In this paper we propose ESB to be utilised as messaging backbone for remote health monitoring, acting as a single messaging hub for integration of all information sent from the smartphone at the patient side, and deployed in the monitoring server. The event-driven nature of ESB is taken advantage of in our system to support real-time message dissemination by incorporating the publish/subscribe message exchange pattern between data providers (i.e. WBAN sensors) and client application (i.e. remote health monitoring dashboard).

II.3 Context-Awareness

Context is defined as any information that characterises an entity's situation [12], where an entity can be a person or any object that plays a role in any type of interaction. Although the definition of context within computer science communities rooted in user location, context encompasses many other conditions of interest about physical surrounding environments (e.g. lighting, noise level, temperature), network connectivity, communication costs, communication bandwidth, and social situation. Context-aware systems aim at automatically personalising user's environment depending on the user's context, and hence, minimising user interaction with the system and the invoked services [13]. Context-awareness is a prerequisite for adaptivity [14], and when applied in a smarthome environment, it can potentially release the inhabitants from doing mundane tasks and support their wellbeing at home. However, smarthome inhabitants do not always stay at home, and thus, context-aware applications and services should also be deployed in devices which the inhabitants bring with them most of the time. Smartphone is a potential target device for such applications.

Acquiring context is the starting point for any context-aware system. Context acquisition is the process where the real conditions surrounding the entity of interest is captured, the significant features are assessed, and an abstract representation is created, which is then provided to components in the system for further use. In order to support remote health monitoring and other eHealth services, a context-aware system architecture should enable integration of a variety of sensors for gathering context data.

III. System Design

III.1 Home Integration Platform

In order to enable patients to live in their homes as long as possible while still being taken care of remotely by healthcare providers, integration of different devices and services in a smarthome environment is necessary to enable collaboration among them. By utilising SOA principles, value added services beyond basic utilisation of each device or service can be provisioned. To achieve this, a logical home integration platform is needed in the smarthome environment, acting as a logical central hub for all communications among devices and services. This platform should provide service creation, composition, and event-driven message handling functionalities. Additionally, the platform should also follow standard-based interoperability best practices by utilising open standards. Figure D.1 shows the proposed architecture of the home integration platform, which is part of the authors' previous work in [15].

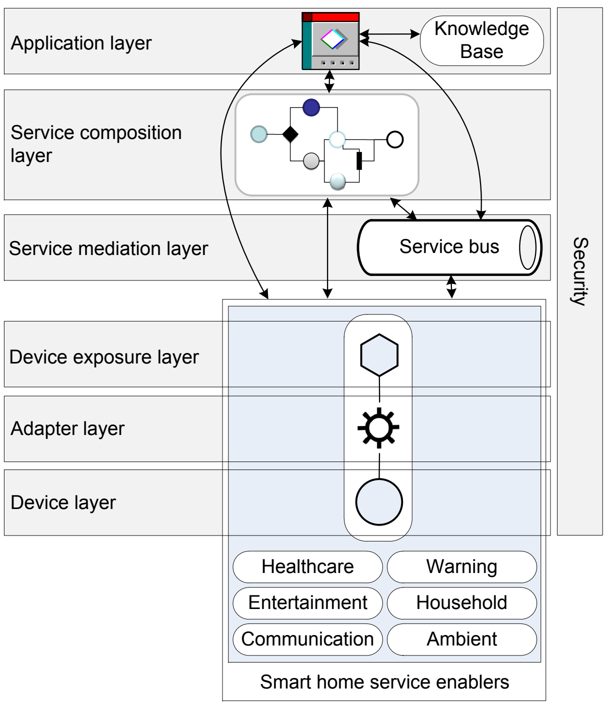


Figure D.1: Home integration platform architecture

The architecture of the platform follows a multi-layer approach, consisting of seven main layers described as follows.

1. **Device layer**. This bottom-most layer consists of various devices in the smarthome which are used for providing data to the platform. These devices can be controlled by the three upper-most layers of the platform. Devices in this layer are not necessarily physical devices, but any data source which may provide usable information to smarthome applications.

2. **Adapter layer**. This layer is responsible for translating a service operation call from the adjacent upper layer to a native API call of a device. For incoming data from the lower layer, this layer is responsible for transforming incoming data format to the suitable format of the upper layers, including call-backs to applications. The adapters can be implemented in the device itself if it has the capability to be programmed, otherwise a separate box with listeners to incoming messages from the device should be used. If a mediation layer is used, such as ESB, specific message translation functionalities of the mediation layer's components can be used as adapters.

3. **Device exposure layer**. This layer is responsible for exposing the functionalities of devices from the bottom-most layer in a standard way to promote interoperability between service providers and service consumers. This layer turns the underlying details of functionalities and data from the devices into services, acting as an interface to service consumers. In traditional web service approach, this layer's role is mainly played by a description language called Web Services Description Language (WSDL) [16]. If a mediation layer is used, such as ESB, devices' functionalities exposures can be performed by the mediation layer.

4. **Service mediation layer**. This layer is responsible for mediating communications between service providers and service consumers in the SOA environment, playing the role of a service broker. Publish/subscribe messaging pattern is one of the most widely used communication methods played by this layer's role, where service consumers subscribe to specific information to the service broker, and get notifications of new information whenever the corresponding service providers publish new information to the service broker. This layer should have its own data storage for persisting all incoming information from the devices to enable later information retrieval from service consumers. ESB technology is an example that suits well in this layer.

Implementation of this layer can either be deployed in the smarthome or in the cloud. This opens up opportunities for trusted third party service brokers to participate as well.

5. **Service composition layer**. This layer provides the capability to combine and link existing services, either atomic or composite services, and create new value-added services. Service composition can be seen in a part-of sense where a larger part encapsulates services and exposes itself as a service, or in a sequencing sense where invocation order of existing services is defined. Web Services Business Process Execution Language (WS-BPEL) [17] is a commonly used language for web service composition for part-of composition, and many ESB implementations support message flow for sequential invocation of services. This layer can be implemented either in the smarthome or in the cloud.

6. **Application layer**. This upper-most layer acts as a host to applications created to provide services to the inhabitants in the smarthome, encapsulating different logics. The main intelligence in the smarthome is envisaged to be residing in this layer by making use of underlying basic and/or composite services. Applications in this layer can be deployed within the smarthome environment as well as in third party service providers' data centres or in the cloud. Since the applications use published APIs from underlying layers, security for the message exchange between these applications (i.e. service consumers) and the home integration platform (i.e. service provider) is of high importance to be taken into consideration.

7. **Security layer**. This vertical layer is responsible for handling security issues across different horizontal layers. Different security approaches may be applied to different layers depending on the required level and type of security by each layer's implementation. Since devices' functionalities and access to devices' information/measurements are published in terms of standardised APIs (especially using web service interfaces), messaging and transport layer security between service providers and service consumers will be the main focus of concern. On top of it, different user roles may be granted access to only a subset of all resources.

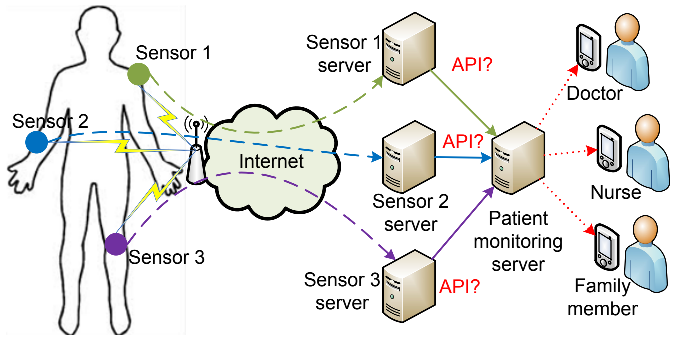
By deploying this architecture, devices' functions and data are exposed as services that enable new services such as remote health monitoring to be produced faster to meet the inhabitants' demands.

III.2 Indoor-Outdoor Scenario

A real-time health monitoring system requires measurements of vital signs from sensors worn by the patient to be sent to a back-end monitoring server for notification and further analysis. Each sensor can send the measurements directly to the back-end monitoring server or through a gateway, which then relays the information. A direct approach (without any Internet gateway), which is shown in Figure D.2(a), is rarely adopted as each sensor would need its own Internet connectivity. This is not feasible especially in outdoor scenarios where Internet connectivity is mainly provided through cellular links, which then requires each sensor to have its own SIM card as well as physical connectivity component and corresponding stack with relatively high energy consumption. One alternative solution to tackle this issue is relaying the measurements to a personal WBAN gateway with its own SIM card, which the patient carries around everywhere he/she goes. As smartphones' processing capabilities have increased quite drastically within the last couple of years nearing to the performance of current entry-level personal computers, coupled with ever decreasing price tags, it can be expected that smartphones will become the standard communications devices in the near future. The next generation of elderly people is expected to use smartphones in their daily lives as well. From this standpoint, treating smartphones as WBAN gateways for remote health monitoring purposes is reasonable especially for outdoor scenarios, as shown in Figure D.2(b). The smartphone carried by the patient will aggregate data from different sensors within the WBAN (e.g. via Bluetooth) and transmits them to the back-end servers.

Figure D.2(b) is becoming a common practice within the consumer healthcare sector where healthcare-related vendors sell their devices/sensors to consumers bundled with smartphone applications (e.g. iOS applications). These applications usually act as visualisation and storage tools for the end user only. Some vendors also provide storage servers with web-based front-ends, where the servers receive the measurements from their smartphone applications. However, not all vendors provide this functionality, and only a small fraction of those who provide it also expose the measurement data by means of APIs (e.g. web service interfaces). The lack of APIs for gathering measurement data, either at the WBAN gateway or at the sensor's server, is considered to be a major drawback in remote health monitoring in particular, especially considering the vast number of device/sensor vendors in the market. Each vendor creates its own island of user measurement data which cannot be combined with other measurement data from other vendors' devices. This situation will prevent the realisation of a unified remote health monitoring system that

(a) Logical direct transmission



(b) Logical brokered transmission

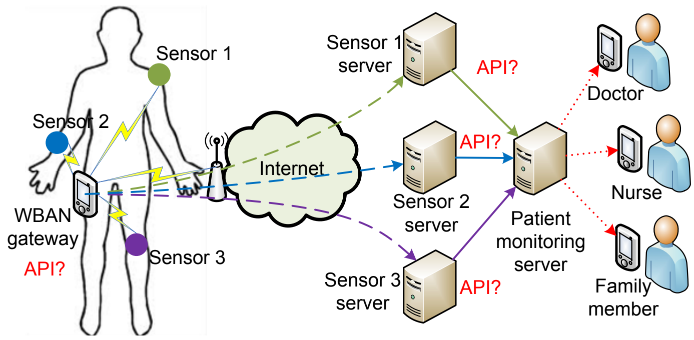


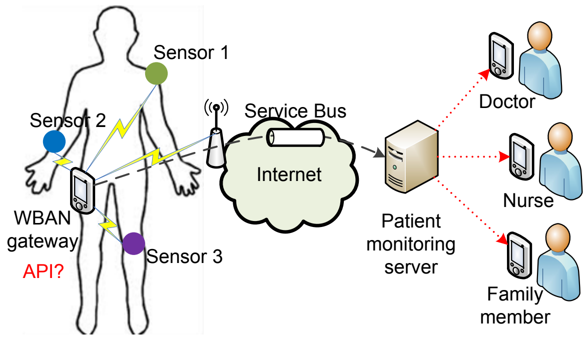
Figure D.2: Different approaches of sensor measurements transmission to back-end servers

aggregates all measurement data from various devices that the patient uses.

A better and simpler approach would be to aggregate all measurement data in the WBAN gateway (i.e. smartphone), store them locally, and forward them to the back-end patient monitoring server whenever Internet connectivity is available for the portable gateway, as shown in Figure D.3(a). All measurement data are sent to a service bus (service broker/mediator) following the high-level architecture of the home integration platform in Figure D.1, which makes the smartphone aggregator application and body-worn sensors acting as service providers, and the patient monitoring server acts as the service consumer from SOA perspective. The service bus then forwards all data to the patient monitoring server. To realise this architecture, a common, standardised, and rich API is needed to be present at the WBAN gateway side that covers a wide range of access network protocols, such as Bluetooth.

An important aspect of a remote health monitoring system is its pervasiveness to be used in different locations. Although it may be impossible to support all scenarios for remote health monitoring, in general monitoring locations can be divided into two different basic cases from the patient's perspective: at home (i.e. indoor) and outside the home (i.e. outdoor). In a smarthome environment, many sophisticated devices are expected to be present to assist the patient's daily life. Data

(a) Outdoor, brokered, aggregated transmission



(b) Indoor, brokered, aggregated transmission

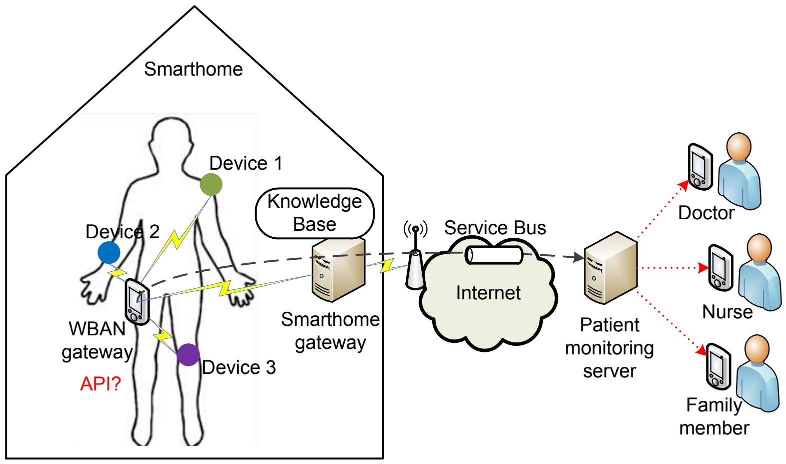


Figure D.3: Outdoor and indoor brokered transmissions

gathered from these devices can be used to enhance the smarthome's reasoning processes in assisting the patient when combined with data gathered from body-worn devices. Internet connectivity in a smarthome environment is also expected to be more reliable compared to smartphone's cellular-based connectivity. Thus, it is beneficial if the smartphone (as WBAN gateway) relays all measurement data to the smarthome's gateway, for example through a WiFi link as shown in Figure D.3(b). The smarthome gateway application then forwards all data to the service bus, which in turn sends all data to the monitoring server. The service bus can be deployed in the smarthome (e.g. co-located with the smarthome gateway application, reachable from the Internet), in the cloud, or directly at the patient monitoring server for last-mile integration.

Transmitting all gathered data from sensors to the monitoring server may not be necessary as many of the measurement values may be similar or are only slightly different from previous readings. In this case, it is useful if the smartphone and smarthome gateway (for indoor scenario) can carry out a reasoning logic every time new data from a sensor arrives. This will minimise the Internet bandwidth required between the smartphone as well as the smarthome gateway and the monitoring server. To achieve this, a context-aware application should be deployed in the smartphone and smarthome gateway. This application can be used to detect anomalies of the patient's health status and can send corresponding alarm messages to the monitoring server as well as directly contact responsible healthcare personnel. However, energy consumption needs to be taken into account as well in particular for the smartphone, where heavy reasoning processes may have negative impacts on the battery lifetime. Therefore, more sophisticated reasoning processes should be employed at the monitoring server. Figure D.4 illustrates the general idea of 2-level reasoning processes for outdoor (left) and indoor (right) scenarios. Whenever the patient enters the smarthome, the smartphone gateway application should sense the location change, turns off its local reasoning logic, and relays all measurement values to the smarthome gateway application, which will then take over the reasoning processes with its own logic.

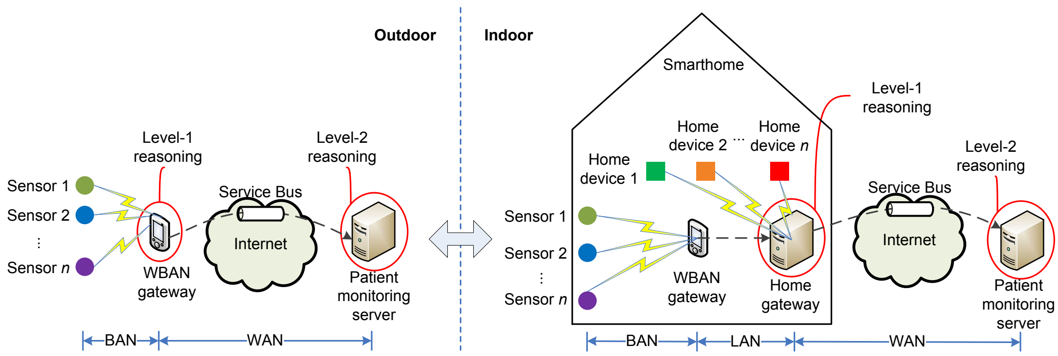


Figure D.4: 2-level reasoning processes: outdoor (left) and indoor (right)

IV. Proof-of-Concept Implementation

A simple prototype of remote health monitoring system has been implemented following the general integration platform architecture in Figure D.1 and the system design described in the previous section. The system architecture of the prototype is depicted in Figure D.5. As web service has become the de facto standard for interoperable machine-to-machine interaction, web service interfaces are provided in the system for data reception services, which enables sensors, through the smartphone gateway, to send measurement data from any location as long as Internet connectivity is available. The on-body sensors worn by the patient become data providers, with the smartphone gateway as a relay. The smartphone application performs light reasoning processes before forwarding the gathered information from the sensors such as checking whether the new data value from a particular sensor is equal to the previously received value. Web-based client application, acting as a service consumer from SOA perspective, is chosen instead of native application for a specific platform since web-based application is more pervasive in today's information technology landscape, as web browsers are generally available for all platforms. This will avoid platform-specific application development.

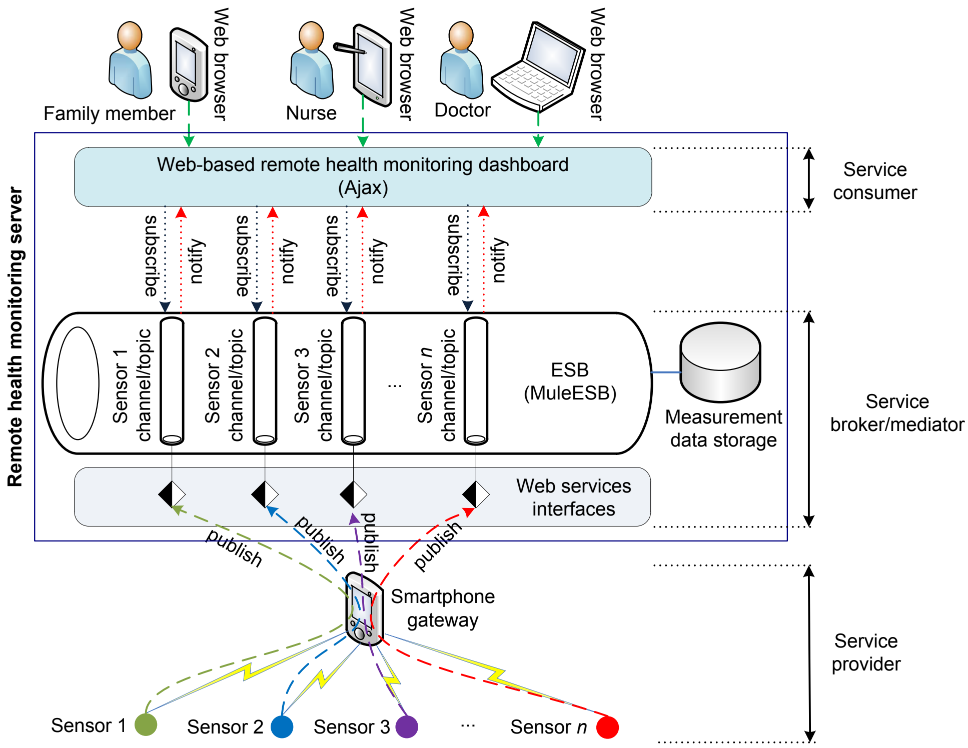


Figure D.5: System architecture of remote health monitoring prototype

IV.1 Smartphone Application

For the prototype implementation, an Android-based smartphone is used as WBAN gateway device for different sensors. The sensors provide different measurement data including pulse rate, body temperature, and blood oxygen saturation. These values are then checked by the smartphone application whether they are similar to previous values, and if they are exactly the same, those data will not be transmitted to the monitoring server. This will minimise Internet bandwidth usage, which in turn will save some cost when transmitted through a cellular network. Since the remote health monitoring system is aimed to be used in both indoor (at home) and outdoor (outside the home) environments, location information is of high importance, especially for the latter case. Location information can be useful to locate the patient when an emergency situation occurs, as well as track his/her travelling route for further analysis. The internal GPS receiver of the smartphone is utilised for this purpose by gathering the latitude and longitude information of the smartphone. The current configuration for the location listener is set to receive notifications of new location from both GPS_PROVIDER and NETWORK_PROVIDER (WiFi and 3G), which are made available through the LocationManager class, only if the patient moves 10 meters from the last-known location, and the accuracy should be less or equal than 20 meters for outdoor GPS location provider and less or equal than 60 meters for indoor WiFi network provider (the lesser the value, the more accurate the location information is). The average accuracy during several testing sessions is 6 meters for outdoor GPS location provider and 50 meters for indoor WiFi network provider. The application presents the patient with his/her location on a map as well as the measurement values.

The application makes use of ksoap2 web service client library to wrap the gathered data as SOAP messages and send them to the back-end monitoring server through web service end-points. The sensors use Bluetooth technology as transport medium to send all measurement data to the smartphone, and the smartphone application utilises Bluetooth Health Device Profile (HDP) API which is provided since Android 4.0 (API level 14). Figure D.6 shows a screenshot of the smartphone application.

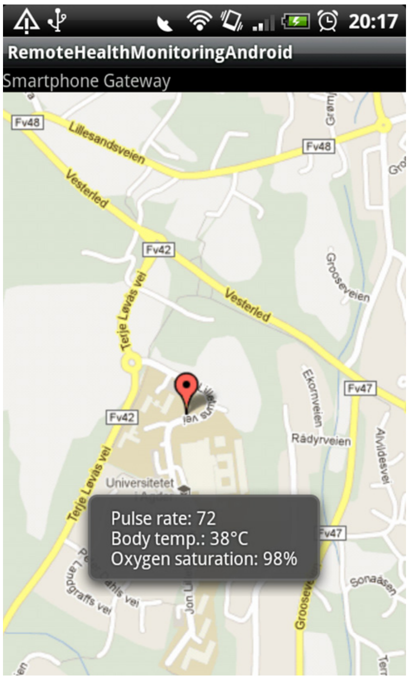


Figure D.6: Smartphone application

IV.2 Remote Health Monitoring Server Application

The server-side application consists of an ESB and a web-based client application based on Ajax. For the prototype implementation, open source MuleESB software is used due to its wide range of supported transport protocols as well as ease of development by means of a friendly integrated development environment called MuleStudio. The web-based client application provides a remote health monitoring dashboard to healthcare personnel and family members, which can be accessed pervasively through the Internet using various web browsers of choice (including those on mobile devices). This client application subscribes to different topics/channels of sensor information fed by the smartphone application at the patient side. Whenever new information reaches any of the subscribed channels/topics, the client Ajax application is notified by means of callback functions. By utilising the publish/subscribe message exchange model which is supported by the ESB, a real-time remote health monitoring dashboard can be provisioned for “anytime, anywhere” patient monitoring. In the current prototype, four channels/topics are deployed for four different data types: location, pulse rate, body temperature, and blood oxygen saturation. Figure D.7 shows an Ajax client application code snippet which subscribes to different information topics/channels.

```
...
mule.subscribe("/services/location", callbackLocation);
mule.subscribe("/services/pulserate", callbackPulseRate);
mule.subscribe("/services/bodytemp", callbackBodyTemp);
mule.subscribe("/services/oxygensaturation", callbackOxygenSaturation);
...
```

Figure D.7: Ajax subscriptions code snippet

MuleESB can be used to host different Mule applications. A Mule application is a deployment unit that encapsulates the necessary requirements which an application would need to function, such as libraries, custom code, and any environment settings accompanying the application [18]. The Mule application contains one or more Mule flows in the form of XML configuration files. A Mule flow consists of pre-packaged building blocks which are defined in specific sequences. The Mule application processes and orchestrates Mule messages based on those sequences [19]. The MuleStudio provides two ways to modify building blocks in a Mule flow, either through a graphical user interface or by writing corresponding XML tags of those building blocks directly in the XML configuration file.

A Mule message crosses from one block to the next block in the Mule configuration file while each block processes the message and takes actions according to its configuration. A Mule message consists of three parts: header, payload, and an attachment. The header contains sets of properties and is used to route the message to the destination. The payload consists of patient-specific data, such as pulse rate information, that is read from a sensor. Each message can also carry an optional header along with it.

Ajax is a set of interrelated technologies that simplify the creation of asynchronous web applications. With the aid of Ajax, web applications are able to send and receive data to and from servers while maintaining a consistent and homogenous user interface [20]. MuleESB provides the Ajax namespace and Ajax connector in order to bind the Mule flow and the web service to the Ajax channel. The Ajax endpoint is configured as an outbound operation. It creates a transport channel to send messages asynchronously to and from an Ajax server, which connects the Mule flow to an external web page. A JavaScript function attached to the web-based client application listens for incoming messages. It extracts and classifies the received data and shows it on the web page. Figure D.8(a), D.8(b), D.8(c), and D.8(d) depict mappings of MuleStudio flows to their corresponding XML configurations in MuleESB for handling the aforementioned four different information.

The server application has a local storage for persisting measurement data which is realised by a Java component. The storage itself utilises a MySQL database management system. Java components can be used for reasoning purposes in the message flow as well for future extensions. Figure D.9 depicts the remote health monitoring dashboard accessed via a web browser. The location of the patient is rendered in real-time on the browser's canvas whenever new latitude and longitude information is sent by the smartphone application. Any new information from the sensors is also updated instantaneously.

IV.3 Ontology-Based Context Modelling

In order to realise the indoor-outdoor scenario as depicted in Figure D.4, the smartphone application should be able to sense the location of the patient (i.e. detect whether the patient is at home or outside). But since the location information is only required to alter the connectivity selection of the smartphone, only reachability of the smarthome gateway is needed for this scenario (i.e. a patient is considered to be at home when the smartphone detects its reachability to the smarthome gateway). This can be achieved by the smartphone, for instance, by saving the WiFi's SSID at home which provides connectivity to the smarthome gateway, connect automatically to the access point whenever in range, and stop the cellular Internet connection. Location and connectivity are context data that the smartphone application needs to capture, store, and decide upon. To better manage these different context situations, a context model is needed to formally represent different context situations.

(a) Location information flow


```
<flow name="LocationFlow" doc:name="LocationFlow">
    <inbound-endpoint address="http://128.39.201.109:65082/services/Location" doc:name="Location Inbound"/>
    <cxf:jaxws-service port="80" serviceClass="no.uia.ehealth.ws.Location" enableMuleSoapHeaders="false" doc:name="SOAP"/>
    <component doc:name="Java">
        <singleton-object class="no.uia.ehealth.ws.Location"/>
    </component>
    <ajax:outbound-endpoint channel="/services/location" connector-ref="AjaxConnector" doc:name="Ajax"/>
    <echo-component doc:name="Echo"/>
</flow>
```

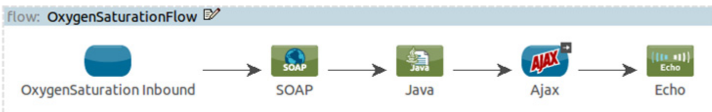
(b) Pulse rate information flow


```
<flow name="PulseRateFlow" doc:name="PulseRateFlow">
    <inbound-endpoint address="http://128.39.201.109:65082/services/PulseRate" doc:name="PulseRate Inbound"/>
    <cxf:jaxws-service port="80" serviceClass="no.uia.ehealth.ws.PulseRate" enableMuleSoapHeaders="false" doc:name="SOAP"/>
    <component doc:name="Java">
        <singleton-object class="no.uia.ehealth.ws.PulseRate"/>
    </component>
    <ajax:outbound-endpoint channel="/services/pulserate" connector-ref="AjaxConnector" doc:name="Ajax"/>
    <echo-component doc:name="Echo"/>
</flow>
```

(c) Body temperature information flow


```
<flow name="BodyTempFlow" doc:name="BodyTempFlow">
    <inbound-endpoint address="http://128.39.201.109:65082/services/BodyTemp" doc:name="BodyTemp Inbound"/>
    <cxf:jaxws-service port="80" serviceClass="no.uia.ehealth.ws.BodyTemp" enableMuleSoapHeaders="false" doc:name="SOAP"/>
    <component doc:name="Java">
        <singleton-object class="no.uia.ehealth.ws.BodyTemp"/>
    </component>
    <ajax:outbound-endpoint channel="/services/bodytemp" connector-ref="AjaxConnector" doc:name="Ajax"/>
    <echo-component doc:name="Echo"/>
</flow>
```

(d) Blood oxygen saturation information flow




```
<flow name="OxygenSaturationFlow" doc:name="OxygenSaturationFlow">
    <inbound-endpoint address="http://128.39.201.109:65082/services/OxygenSaturation" doc:name="OxygenSaturation Inbound"/>
    <cxf:jaxws-service port="80" serviceClass="no.uia.ehealth.ws.OxygenSaturation" enableMuleSoapHeaders="false" doc:name="SOAP"/>
    <component doc:name="Java">
        <singleton-object class="no.uia.ehealth.ws.OxygenSaturation"/>
    </component>
    <ajax:outbound-endpoint channel="/services/oxygensaturation" connector-ref="AjaxConnector" doc:name="Ajax"/>
    <echo-component doc:name="Echo"/>
  </flow>
```

Figure D.8: Mappings of MuleStudio flows to their corresponding XML configurations

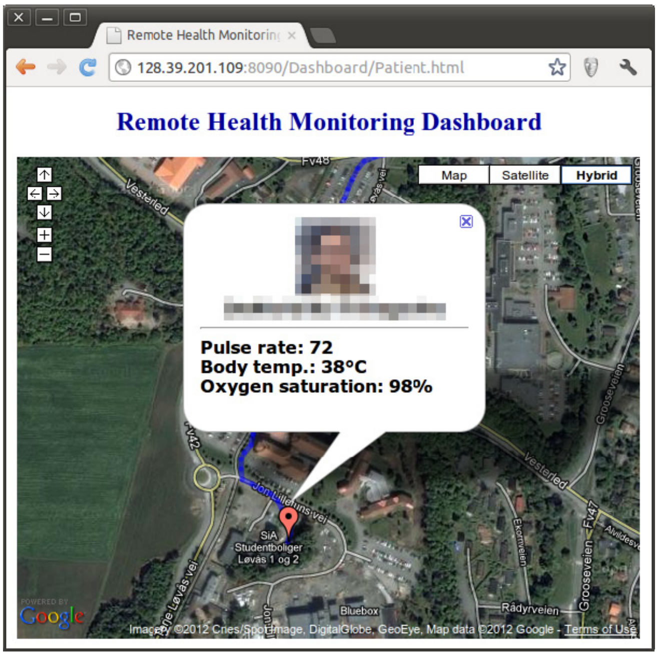


Figure D.9: Web-based remote health monitoring dashboard

Ontology-based context modelling has been used quite often in recent years due to its structured and rich description of a user's context that allows semantic reasoning. An ontology model is considered to have an advantage over other context-modelling approaches as it enables knowledge sharing in open dynamic systems, allows an efficient reasoning on context information with well-defined declarative semantics, and enables service interoperability as well as collaborative networked services in a non-ambiguous manner [21]. Thus, an ontology-based context-aware system is a good choice to be incorporated by the smartphone application to maintain the knowledge base related to the patient, including his/her location and the type of connectivity to the Internet.

To maximise the always-online probability of the remote health monitoring service, redundant Internet connections are important to be considered. This may not be a big issue for the smarthome gateway as both fixed and wireless Internet connectivity can easily be provisioned, but it is challenging for the smartphone gateway as the option for fixed connectivity does not apply, and yet redundancy in connectivity is needed to support outdoor patient monitoring. Both smarthome and smartphone gateways should have one primary Internet connectivity link and at least one alternative/backup connection, with the smarthome gateway's Internet connectivity having higher priority than the smartphone's (i.e. the smartphone gateway relays all data gathered from the WBAN to the smarthome gateway whenever it is reachable). Table D.1 shows possible combinations of Internet connectivity redundancy for both gateway types.

Priority	Gateway	Primary	Alternative/Backup
1	Smarthome	xDSL	UMTS/WCDMA, GPRS/EDGE, WiFi, ISDN, Satellite
2	Smartphone	UMTS/ WCDMA	GPRS/EDGE, Satellite, GSM-SMS

Table D.1: Possible connectivity redundancy combinations

An ontology prototype based on Web Ontology Language (OWL) [22] has been developed as shown in Figure D.10. This ontology is designed to be incorporated by the smartphone application as a knowledge base of different contextual information where the smartphone application can decide upon.

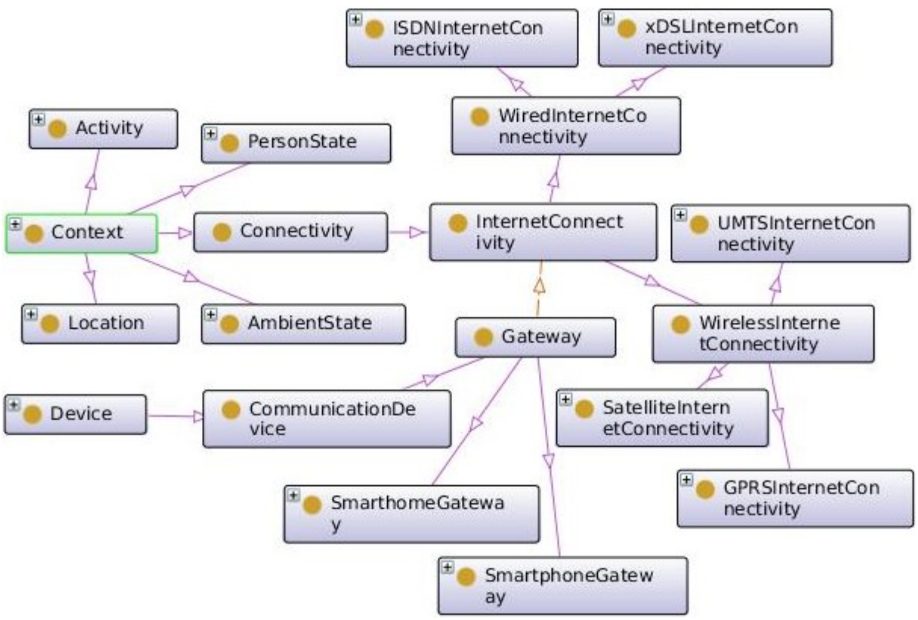


Figure D.10: Ontology-based context model

Several context parameters are depicted in Figure D.10, and connectivity is one of them. In addition to better manage contextual information, this knowledge base can also be used to conduct some reasoning processes related to connectivity issues. One example of such reasoning is to make a decision when to choose a new link in case the default link is down. The Semantic Web Rule Language (SWRL) [23] is used to show this example as follows.

```
Patient(?A) ∧ SmartHome(?B) ∧ livesIn(?A,?B) ∧ hasCom-
municationDevice(?A,?C) ∧ hasSmarthomeGateway(?B,?D) ∧
hasNoReachability(?C,?D) ∧ hasDefaultInternetConnectiv-
ity(?C,?E) ∧ isActive(?E,True) ∧ isUp(?E,False) ∧ hasS-
martphoneGatewayAction(?C,?F) → chooseNewLink(?F,True)
```

In this rule it is checked if the default Internet connectivity of the smartphone gateway is down. The reachability of the smarthome gateway from the smartphone gateway is also checked (i.e. checking whether the patient is at home or in the near vicinity), and if it is out of reach, a new (alternative) link should be chosen.

V. Discussion

Developing and deploying an always-on remote health monitoring system is not an easy task. As the Internet is used for exchanging messages between sensors and the monitoring server, availability and reliability of the Internet connectivity becomes a crucial aspect to be considered in such a service. Redundant Internet connectivity is recommended to be deployed both for the smarthome and for the smartphone. The smartphone should be location-aware such that it will switch Internet connectivity from its own to the smarthome gateway's as soon as reachability between both devices is detected and connectivity is established. Applications in both devices should also have local data storages for storing all measurement data which are useful when Internet connectivity of either device is down. These data will then be sent automatically to the monitoring server whenever Internet connectivity is up again. The drawback is that, if this happens, the remote health monitoring service will lose its real-timeliness as the most current measurements are not updated/shown. From this perspective, Internet connection redundancy for availability directly affects the service's information real-timeliness and accuracy.

The ESB plays a central role as data aggregation point, connecting service providers with service consumers. Following the system architecture in Figure D.5, the current prototype implementation of the remote health monitoring service treats the ESB as last-mile integration component. This means the ESB (service mediator/broker) is co-located with the client web application (service consumer) which is directly accessed by the users (e.g. doctors, nurses, family members of the patient). This is adequate for deploying a service such as the remote health monitoring prototype. However, this implies all additional services to be either co-located with the ESB in the same machine, or to have their own ESBs. This will become a major flexibility drawback when different application service providers are involved to provide different services in the cloud. Thus, a better approach would be to deploy the ESB in the smarthome or in the cloud, with two general categories of web service interfaces: one for service providers to provide data, and the other one for service consumers to access the data. And as only a small subset of ESB functionalities are used for healthcare-related services, a cut-down, custom-made service bus is sufficient to be implemented as a service mediator/broker. RESTful web service interfaces are also considered to be more appropriate to be used, facing both service providers and service consumers, instead of the traditional SOAP-based web service interfaces as RESTful web services have been proven to be more lightweight.

VI. Conclusions and Future Work

Remote health monitoring is an important part of telehealth which enables early detection of health anomalies and preventive distant care. A web-based remote health monitoring "anytime, anywhere" system utilising ESB is presented in this paper, enabling distantly-located healthcare personnel and family members of a patient to monitor the patient in a real-time manner with a web browser of choice. This is achieved by employing publish/subscribe message exchange pattern functionality of the ESB. Web service interfaces are used to expose inbound data entry points to the monitoring flow within the ESB. A smartphone gateway is used at the patient side as data relay for on-body WBAN sensors worn by the patient. Simple reasoning processes are carried out by the smartphone application before transmitting the aggregated data to the monitoring server through the Internet.

From the prototype implementation it can be concluded that ESB is well suited for remote health monitoring service, acting as a central hub for various information coming from different devices. By combining event-driven architecture and SOA, ESB provides client applications the possibility to subscribe to information of interest and to get notification whenever new information comes without having to send requests every certain period of time. By adding location-awareness to the smartphone application, indoor-outdoor roaming of the patient also becomes possible to achieve.

Physical separation of service mediator/broker and service consumers is planned to be carried out for future work by simplifying the ESB and adding web service interfaces facing service consumers. RESTful web services are planned to be utilised, instead of SOAP-based web services, facing both service providers and service consumers. Implementation of semantic reasoning based on ontologies in the smartphone gateway is also planned to be conducted.

REFERENCES

[1] E. Mørk, "Seniorer i Norge 2010," Statistisk sentralbyrå, February 2011.

[2] S. Sataline and S. S. Wang, "Medical Schools Can't Keep Up," [Online]. Available: `http://online.wsj.com/article/SB10001424052702304506904575180331528424238.html` [Accessed 10th May 2012].

[3] E. Fife and F. Pereira, "Digital home health and mHealth: Prospects and Challenges for Adoption in the U.S.," in *Proc. of 50th FITCE Congress,* Palermo, Italy, September 2011, pp. 1-11.

[4] M. Gagnon, G. Paré, H. Pollender, J. Duplantie, J. Côté, J. Fortin, R. Labadie, E. Duplàa, M. Thifault, F. Courcy, C. A. McGinn, B. A. Ly, A. Trépanier, and F. Malo, "Supporting Work Practices through Telehealth: Impact on Nurses in Peripheral Regions," *BMC Health Services Research*, vol. 11, no. 1, pp. 1-9, February 2011.

[5] M. A. Laguna, J. Finat, and J. A. González, "Remote Health Monitoring: A Customizable Product Line Approach," *Lecture Notes in Computer Science*, vol. 5518, pp. 727-734, 2009.

[6] M. Abousharkh and H. Mouftah, "Service Oriented Architecture-based Framework for WBAN-enabled Patient Monitoring System," in *Proc. of 2nd Kuwait Conference on e-Services and e-Systems (KCESS),* Kuwait City, Kuwait, April 2011, pp. 1-4.

[7] Y. Hongzhou and L. Lu, "Remote Health Monitoring System Using ZigBee Network and GPRS Transmission Technology," in *Proc. of 4th International Symposium on Computational Intelligence and Design (ISCID),* Hangzhou, China, October 2011, pp. 151-154.

[8] N. Nawka, A. K. Maguliri, D. Sharma, and P. Saluja, "SESGARH: A Scalable Extensible Smart-Phone based Mobile Gateway and Application for Remote Health Monitoring," in *Proc. of IEEE 5th International Conference on Internet Multimedia Systems Architecture and Application (IMSAA),* Bangalore, India, December 2011, pp. 1-6.

[9] D. Barry, "Web Services and Service-Oriented Architecture: The Savvy Manager's Guide," Morgan Kaufmann Pub, 2003.

[10] D. Chappell, "Enterprise Service Bus," O'Reilly Media, Inc., 2004.

[11] M. P. Papazoglou and W. J. Van Den Heuvel, "Service Oriented Architectures: Approaches, Technologies and Research Issues," *The VLDB Journal*, vol. 16, no. 3, pp. 389-415, March 2007.

[12] A. K. Dey, "Understanding and Using Context," *Personal and Ubiquitous Computing*, vol. 5, no. 1, pp. 4-7, February 2001.

[13] S. Najar, O. Saidani, M. Kirsch-Pinheiro, C. Souveyet, and S. Nurcan, "Semantic Representation of Context Models: A Framework for Analyzing and Understanding," in *Proc. of 1st Workshop on Context, Information and Ontologies,* Heraklion, Greece, June 2009, pp. 1-10.

[14] B. Schilit, N. Adams, and R. Want, "Context-Aware Computing Applications," in *Proc. of 1st Workshop on Mobile Computing Systems and Applications (WMCSA),* Santa Cruz, CA, USA, December 1994, pp. 85-90.

[15] Y. B. D. Trinugroho, F. Reichert, and R. W. Fensli, "A SOA-Based eHealth Service Platform in Smart Home Environment," in *Proc. of 13th IEEE International Conference on e-Health Networking, Applications and Services (HEALTHCOM),* Columbia, MO, USA, June 2011, pp. 201-204.

[16] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana, "Web Services Description Language (WSDL) 1.1," W3C, March 2001.

[17] A. Alves, A. Arkin, S. Askary, C. Barreto, B. Bloch, F. Curbera, M. Ford, Y. Goland, A. Guzar, N. Kartha, C. K. Liu, R. Khalaf, D. König, M. Marin, V. Mehta, S. Thatte, D. Van Der Rijn, P. Yendluri, and A. Yiu, "Web Services Business Process Execution Language Version 2.0," OASIS Standard, April 2007.

[18] A. Perepelytsya and K. Magnusson, "Application Format - Mule ESB 3.3 User Guide," [Online]. Available: `http://www.mulesoft.org/documentation/display/MULE3USER/Application+Format` [Accessed 24th July 2012].

[19] N. Bock and A. Dickey, "Mule Application Architecture - Mule ESB 3.3 User Guide," [Online]. Available: `http://www.mulesoft.org/documentation/display/MULE3USER/Mule+Application+Architecture` [Accessed 24th July 2012].

[20] C. Ullman and L. Dykes, "Beginning Ajax," Wrox Press, 2007.

[21] V. Suraci, S. Mignanti, and A. Aiuto, "Context-aware Semantic Service Discovery," in *Proc. of 16th IST Mobile and Wireless Communications Summit,* Budapest, Hungary, July 2007, pp. 1-5.

[22] D. L. McGuinness and F. Van Harmelen, "OWL Web Ontology Language Overview," W3C Recommendation, February 2004.

[23] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML," W3C Member Submission, May 2004.

Appendix E

Paper IV

Title:	A REST-Based Publish/Subscribe Platform to Support Things-to-Services Communications
Authors:	**Yohanes Baptista Dafferianto Trinugroho**, Martin Gerdes, Mohammad Mahdi Mahdavi Amjad, Frank Reichert, and Rune Fensli
Affiliation:	University of Agder, Faculty of Engineering and Science, Jon Lilletuns vei 9, 4879 Grimstad, Norway
Published in:	*Proceedings of 19th Asia-Pacific Conference on Communications*, Bali, Indonesia, 29-31 August 2013.

A REST-Based Publish/Subscribe Platform to Support Things-to-Services Communications

Yohanes Baptista Dafferianto Trinugroho, Martin Gerdes,
Mohammad Mahdi Mahdavi Amjad, Frank Reichert and Rune Fensli
Dept. of Information and Communication Technology, University of Agder,
Grimstad, Norway
{dafferianto.trinugroho, rune.fensli, frank.reichert}@uia.no

Abstract — **The term "Internet of Things" has been used widely in recent years, bringing up a grand idea of connecting physical objects with one another through both wired and wireless networks. Maturity of the Internet Protocol has pushed 21st century communications between devices to go all-IP, which includes everyday objects. Many new services are deployed in the "cloud", utilising the Internet infrastructure and the Web, and information from everyday objects can play an important role. This paper presents an information integration platform which enables everyday objects to communicate with Internet services based on event-driven service-oriented architecture paradigm. A proof-of-concept prototype that has been developed is also described, and two application scenarios within healthcare domain are presented as well.**

I. Introduction

The Internet of Things (IoT) vision extends the Internet into the real world, embracing everyday objects so that physical items are no longer disconnected from the virtual world [1]. Instead of relying only on people to capture and create information, everyday objects are employed to gather information from the physical world [2].

The Internet has become the universal platform for deploying services beyond time and space barriers. This situation is backed heavily by advancements of information and communications technology (ICT), and the new paradigm of the "cloud", which offers resources and services that allow an efficient deployment of new services. Although communications between everyday objects are important for many localised use cases, communications with myriad Internet-based services is the main driver for global pervasiveness of things.

In order to enable everyday objects to communicate with the current Internet infrastructure, either a full TCP/IP stack is implemented in the objects, or a gateway that can translate object-specific communication protocol to IP based communications should be present. The latter is more feasible to be adopted as everyday objects may have physical limitations to talk IP. On the bright side, personal mobile devices (e.g. smartphones, tablets) are getting more compact in size but yet more powerful in terms of processing capabilities. These devices usually support multiple access technologies (e.g. bluetooth, WiFi, UMTS, LTE), and thus, are good candidates for everyday objects' gateway to the Internet.

Hypertext Transfer Protocol (HTTP) is one of the most widely used application layer protocols to exchange contents over the Internet. It is also the most common protocol used by both traditional Web services [3] and RESTful Web services [4] for message exchange. And since the majority of Internet-based services makes use of Web service interfaces, the HTTP protocol, together with Web service approach, are a good combination to be utilised for communications in the application layer.

Many existing middleware prototypes make use of publish/subscribe messaging pattern. In the research domain, for example, there are Gryphon [5], Hermes [6], JEDI [7], Scribe [8], and SIENA [9]. Although these systems provide relatively complex topic-based, content-based, or type-based subscription schemes, they mainly provide progamming language-specific application programming interfaces (APIs), such as for Java or C++. Within the Web services arena, WS-Eventing and WS-Notification are two major competing specifications for traditional Web services to incorporate publish/subscribe message exchange paradigm. Although their architectures differ from each other, convergence between the two has been realised to a certain degree [10]. However, there is no standardisation initiative for RESTful Web services to include publish/subscribe for supporting an asynchronous communications model.

In this paper, an architecture and prototype for a publish/subscribe platform that aggregates and provides information through information channels for event-driven information dissemination from everyday objects to Internet-based services is presented. The platform itself is designed following SOA paradigm in order to be flexible enough to not only accept information from physical devices but also from any type of information provider. It aims to simplify the information integration process while still following the event-driven SOA concept, so that the platform can be deployed in the "cloud" and can be used out-of-the-box without modifying anything in the platform programmatically. This is mainly achieved by applying a strong constraint that both service providers and service consumers have to exchange messages through RESTful Web services.

II. Information Integration Platform Architecture and Design

The proposed Information Integration Platform (IIP) acts as a broker between information providers and information consumers (service providers and service consumers, respectively, from SOA standpoint) as shown in Fig. E.1.

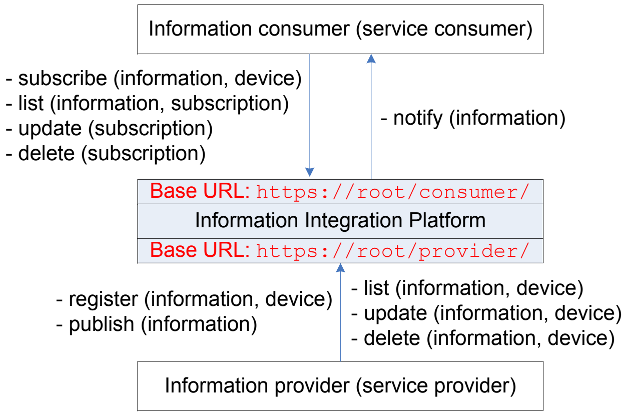


Figure E.1: Information Integration Platform (IIP) general architecture

REST interfaces are used for exposing different services of the platform to both information providers and consumers with two main Web resource base URLs, namely `https://root/provider/` and `https://root/consumer/`, respectively. The `root` part of the URL refers to the domain of the specific IIP deployment. HTTP request methods (i.e. GET, POST, PUT, DELETE) are utilised for exposing the services, where GET is used for nullipotent services, PUT and DELETE for idempotent services, and POST for non-idempotent services.

II.1 Information Provider Resources

Information is shared through different "information channels" which are managed internally by IIP. Information providers should initially register information channels for passing information around. Information consumers can then subscribe to those information channels for notifications, so that whenever any information provider publishes new information to those information channels, all subscribed information consumers will be notified by the IIP with the new information. This approach is almost similar to topic-based publish/subscribe mechanism, but more strongly typed in the sense that information consumers should know exactly which information channels to subscribe to. IIP provides the possibility to categorise different information channels by devices which generate the information.

As depicted in Fig. E.1, there are five main resources that are provided by IIP to information providers. These are *registration*, *publication*, *listing*, *updating*, and *deletion*, described as follows. For authentication, authorisation and accounting (AAA) purposes, each request to an information provider resource has to contain an identifier of the information provider.

1. *Registration:* To register an information channel, IIP provides a resource

with URL `https://root/provider/registration/information/` that accepts HTTP POST with Content-Type `application/x-www-form-urlencoded`. This resource accepts three predefined parameters related to the registered information channel, namely *deviceId*, *name*, and *description*. The *deviceId* parameter is a pointer to an existing device in IIP, and if this parameter is not specified in the HTTP request body (or specified but no such device exists in IIP), a new device will be registered automatically. The *name* parameter is used for naming the information channel, and if it is not specified in the request, then it will be set to a similar value as *infoId*, which is generated by IIP for every information channel being registered. The *description* parameter is optional. In addition to these parameters, IIP automatically generates *creationDate* and *lastupdateDate* for timestamping purposes. If other parameter names (apart from the six that were mentioned earlier) are included in the request body, then these parameters are treated by IIP as information parameters (to store actual values of the registered information channel). For example, a location information can have two information parameters, namely latitude and longitude. After successful processing, an XML representation of the registered information channel will be sent back in the body of the HTTP POST response. Fig. E.2 shows a simplified sequence diagram of the information channel registration process with the IIP.

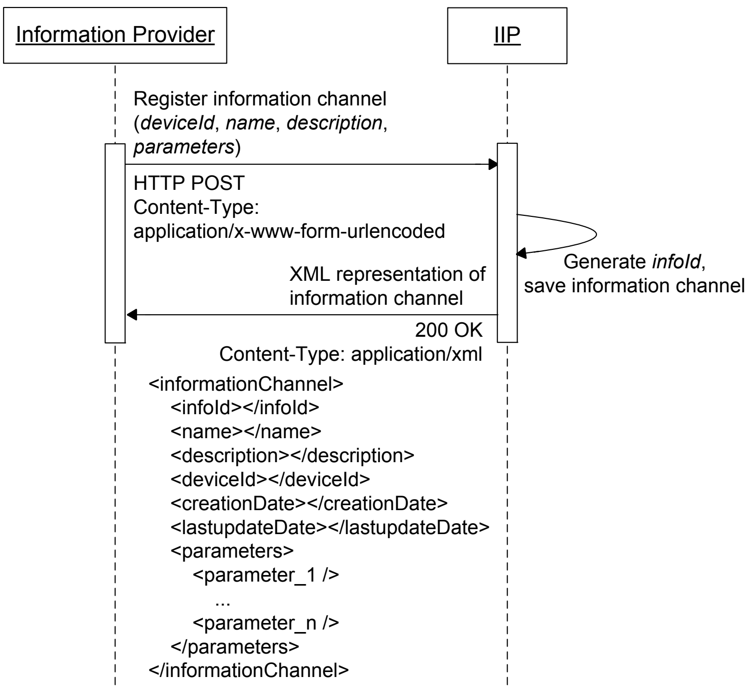


Figure E.2: Information channel registration sequence diagram

The IIP also provides a resource in order to register a device that can contain different information channels with URL `https://root/provider/registration/device/`. This resource accepts HTTP POST with Content-Type `application/x-www-form-urlencoded`, and two predefined parameters, namely *name* and *description*. If not specified, the *name* parameter will be set similar to *deviceId* which is automatically generated by IIP. The *description* parameter is optional. Similar to the information channel registration, IIP automatically generates *creationDate* and *lastupdateDate* for timestamping purposes, and other parameters provided will be omitted. An XML representation of the registered device will be returned in the body of the HTTP POST response.

2. *Listing:* The IIP provides a resource for information providers to retrieve all information channels in XML format that have been registered by using HTTP GET request to URL `https://root/provider/registration/information/`, and a resource to retrieve a specific information channel by *infoId* with URL `https://root/provider/registration/information/{infoId}/`. The same goes for devices, HTTP GET request to URL `https://root/provider/registration/device/` will return an XML representation of all registered devices, and HTTP GET request to URL `https://root/provider/registration/device/{deviceId}/` will return a specific device by *deviceId*.

3. *Updating:* Two resources are provided by the IIP for updating registered information channels and devices via URL `https://root/provider/registration/information/{infoId}/` and `https://root/provider/registration/device/{deviceId}/`, respectively. Both resources accept HTTP PUT with Content-Type `application/x-www-form-urlencoded`, and require *infoId* and *deviceId*, respectively, to be included in the URLs for updating. Accepted parameters are similar to the registration process of both information channel and device, and updating partial parameters is possible (i.e. only a subset of accepted parameters are provided in the request body). The *lastupdateDate* parameter will be updated automatically by the IIP, and an XML representation of the updated information channel or device will be returned in the response body.

4. *Deletion:* The IIP provides a resource for deletion of a specific information channel by *infoId* with URL `https://root/provider/registration/information/{infoId}/`, and another resource for deletion of a specific

device by *deviceId* with URL `https://root/provider/registration/device/{deviceId}/`. Both resources accept HTTP DELETE request, and 204 No Content will be returned as response. Deletion of an information channel will result in automatic deletion of all subscriptions to that particular information channel. Deletion of a device will result in automatic deletion of all information channels (including all subscriptions to them) attached to that particular device.

5. *Publication:* The IIP provides a resource for information providers to publish new information to registered information channels. This information is treated as an event from e.g. any everyday object that is delivered to interested information consumers. Information consumers that have subscribed to corresponding information channels are then notified by the IIP. The URL of this resource is `https://root/provider/publication/{infoId}/`, and it accepts HTTP POST with Content-Type `application/x-www-form-urlencoded`. Information parameters included in the body of the HTTP POST request can be a subset of all information parameters for that particular information channel (partial information update). The *lastupdateDate* parameter of each updated information parameter will be updated automatically by the IIP. If no information parameter is included, then this publication does not contain any new information, and thus, no subscriber will be notified. An XML representation of the published information with the included information parameters and their values will be returned in the body of the response, and the same XML representation will also be forwarded to subscribing information consumers through an HTTP POST request. Fig. E.3 shows the publication and notification sequence diagram.

II.2 Information Consumer Resources

Information consumers (in the IIP terminology) are services that make use of information from information providers through the IIP. These consumers can receive information which they are interested in either by pull mode, where they send requests to specific information channels in the IIP, or by push mode, where they subscribe to selected information channels for different information, and they will be notified by the IIP whenever new information is published by information providers to the information channels they have subscribed to.

As shown in Fig. E.1, there are five main resources provided by the IIP to information consumers. These are *subscription*, *listing*, *updating*, *deletion*, and *notification*, described as follows. Similar to the information provider resources,

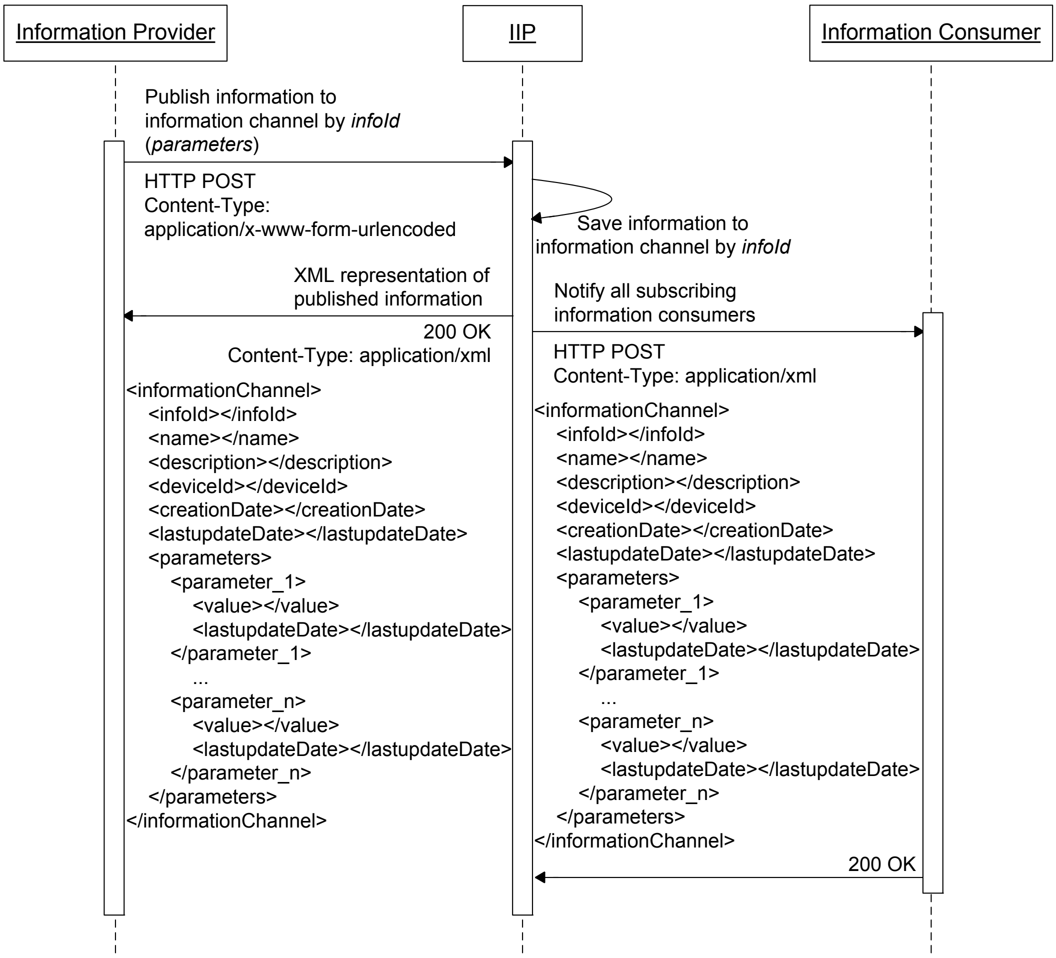


Figure E.3: Information publication and notification sequence diagram

each request to an information consumer resource has to contain an identifier of the information consumer.

1. *Subscription:* The IIP provides a resource which allows information consumers to subscribe to a specific information channel for notifications. This resource accepts HTTP POST request with Content-Type `application/x-www-form-urlencoded` at URL `https://root/consumer/subscription/`. The request body can contain four predefined parameters, namely *infoId*, *notificationUrl*, *name*, and *description*. The *infoId* parameter is mandatory as it points to which information channel the information consumer wants to subscribe to. The *notificationUrl* parameter is also mandatory as it acts as the endpoint of where notifications will be sent to. The *name* parameter is used for naming the subscription, and if it is not provided, then it will be set to a value similar to the *subscriptionId* which is

generated by the IIP for every registration of a subscription. The *description* parameter is optional. The IIP enables an information consumer to replace the *infoId* parameter with *deviceId*. If *deviceId* is used instead of *infoId*, then subscriptions to all information channels attached to the specified device will be registered by the IIP. The *infoId* and *deviceId* parameters cannot be used simultaneously in this resource. Fig. E.4 shows the sequence diagram for the subscription to an information channel.

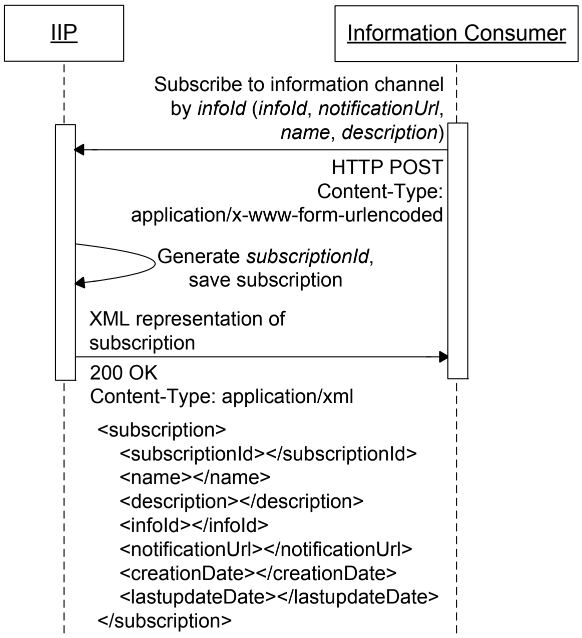


Figure E.4: Subscription to information channel sequence diagram

2. *Listing:* The IIP provides a resource for infomation consumers to retrieve the latest information in XML format from all information channels they have subscribed to previously by using HTTP GET request to URL `https://root/consumer/listing/information/`. In addition, a resource to retrieve the latest information from a specific information channel indicated by the *infoId* is also provided at URL `https://root/consumer/listing/information/{infoId}/` and can be requested by sending HTTP GET. Following this HTTP GET request, the parameters' values and their last update dates are included in the body of the response message. These values are the actual information being passed through the information channels.

   In addition, the IIP provides a resource for information consumers to list all their subscriptions in XML format by sending an HTTP GET request to URL `https://root/consumer/subscription/`. The IIP also provides a resource for listing a specific subscription by *subscriptionId* with URL `https://root/consumer/subscription/{subscriptionId}/` and a resource for listing all subscriptions to a specific information channel by *infoId* with URL `https://root/consumer/subscription/{infoId}/`.

3. *Updating:* A resource is made available by the IIP for information consumers to update a subscription to an information channel via URL `https://root/consumer/subscription/{subscriptionId}/`, which accepts HTTP PUT request with Content-Type `application/x-www-form-urlencoded`. *subscriptionId*, which indicates a specific subscription to an information channel, is mandatory to be included in the URL of this resource, and the accepted parameters are similar to the subscription process, namely *infoId*, *notificationUrl*, *name*, and *description*. Updating partial parameters is possible (i.e. only a subset of the accepted parameters are provided in the request body), and the *lastupdateDate* parameter will be updated internally by the IIP. An XML representation of the updated subscription will be returned in the response body.

4. *Deletion:* The IIP provides a resource for the deletion of a specific subscription to an information channel indicated by *subscriptionId* with URL `https://root/consumer/subscription/{subscriptionId}/`. It accepts HTTP DELETE request, and 204 No Content will be returned.

5. *Notification:* The IIP does not provide a specific resource for notification of newly published information to an information channel.

III. PROTOTYPE IMPLEMENTATION

A prototype of IIP has been implemented using Java Enterprise Edition (EE) platform, utilising Java API for RESTful Web Services (JAX-RS) and Java Persistence API (JPA). A MySQL database is mainly used for storing information, information channels, and subscriptions. HTTPS is used for encrypting all message exchanges and HTTP Basic authentication is utilised to force both information providers and consumers to authenticate themselves in every HTTP request by means of usernames (i.e. information provider and consumer identifiers) and passwords. Fig. E.5 shows the prototype implementation architecture of IIP.

Two prototype services, namely a remote health monitoring and an SOS services, have been developed as proof-of-concept to enhance the safety and quality of life of patients, by utilising the IIP to integrate information from the patients and their environment. Fig. E.6 shows the interaction of these two services

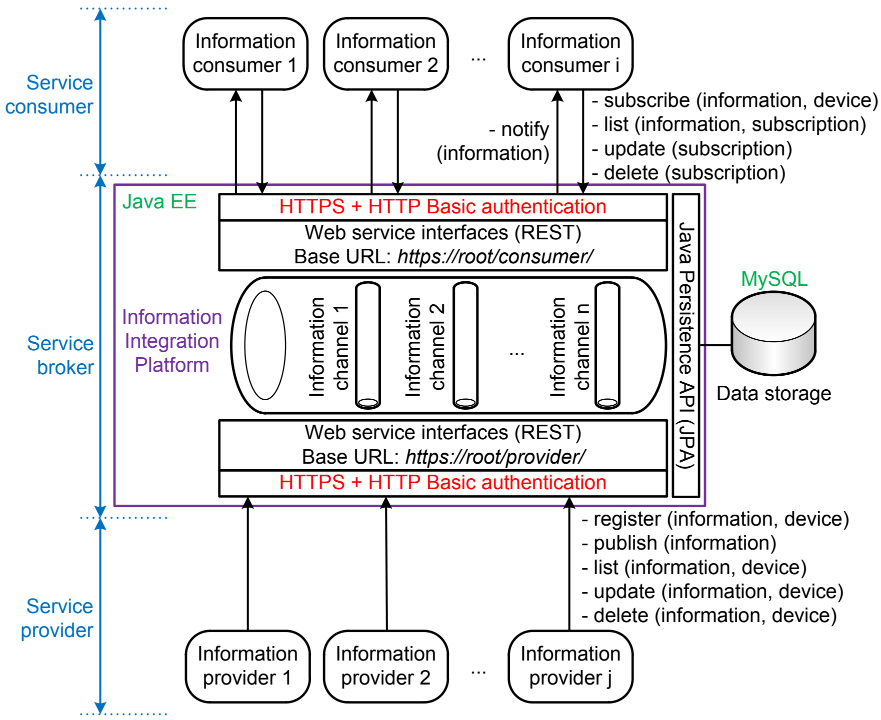


Figure E.5: Information Integration Platform (IIP) prototype implementation architecture

with the IIP.

III.1 Remote Health Monitoring Service

The implemented remote health monitoring service is a Web application that allows healthcare personnel such as doctors, nurses, as well as family members to monitor a remotely-located elderly patient through a Web browser of choice as shown in Fig. E.6. This service subscribes to three different information channels: SpO2 (blood oxygen saturation), pulse rate, and location. Nonin Onyx II pulse oximeter is used for SpO2 and pulse rate measurements, and the data is then transmitted via bluetooth to an Android smartphone. An application on the smartphone, carrying out the information provider role, then publishes both information to two different information channels in the IIP. Whenever there is new information published to these two information channels, the remote health monitoring service is notified by the IIP. Location information is also provided by the smartphone since it has GPS functionality. In this service, the pulse oximeter acts as everyday health-related object that communicates its captured information through IIP to an Internet-based service (i.e. the remote health monitoring dashboard). A similar service was previously developed by the authors on top of an Enterprise Service Bus (ESB) [11].

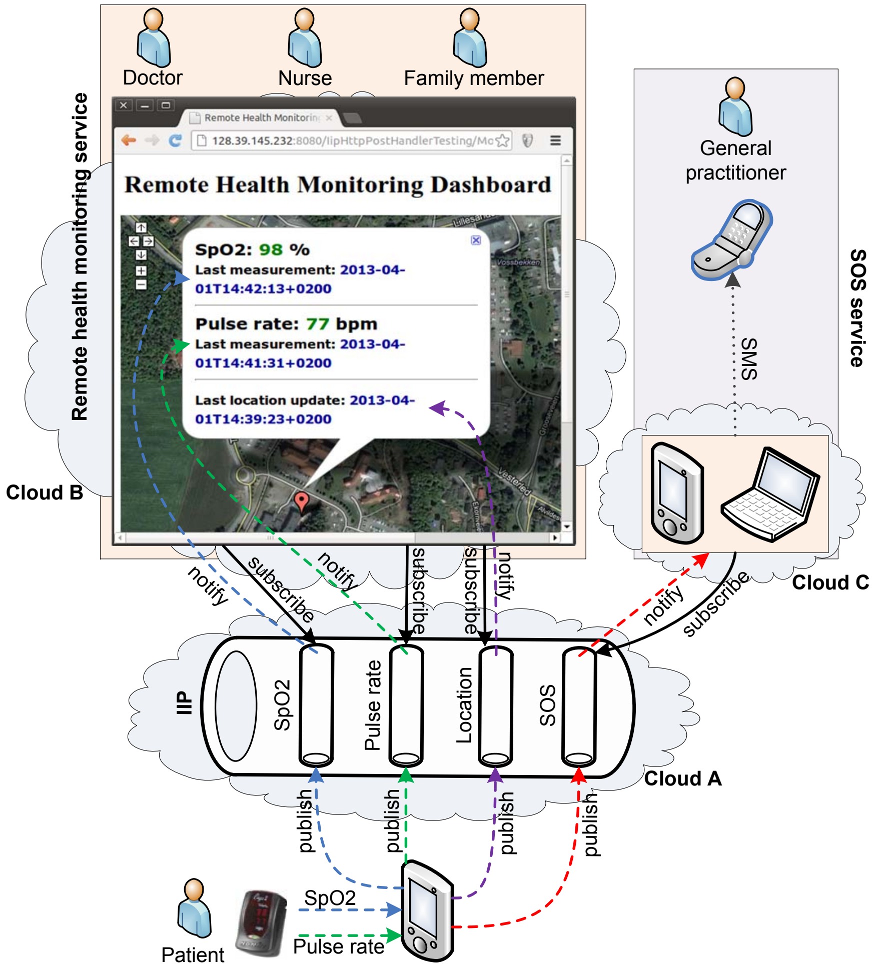


Figure E.6: Remote health monitoring and SOS services through IIP

III.2 SOS Service

The implemented SOS service prototype mainly consists of an Android application, which publishes an SOS message to an SOS information channel in the IIP, shown in Fig. E.7(a), and two applications for handling the notifications from the IIP. The latter applications are an Android application and an ASP.net Web application, and both subscribe to the SOS information channel. The Android application is a light weight Web server that handles HTTP POST notifications sent by the IIP, analyses them, and in case of emergency sends an SMS to a list of recipients. Regardless of the implementation differences, the ASP.net Web application does more or less the same thing, but for sending SMS, an external SMS gateway is used (as opposed to directly sending SMS from an Android phone). When the ASP.net application gets notified by IIP, it sends an HTTP GET request to the SMS gateway, which then sends SMS to the given list of recipients. Fig. E.7(b) shows a received

SMS notification in a mobile terminal.

(a) SOS Android application (b) Received SMS notification

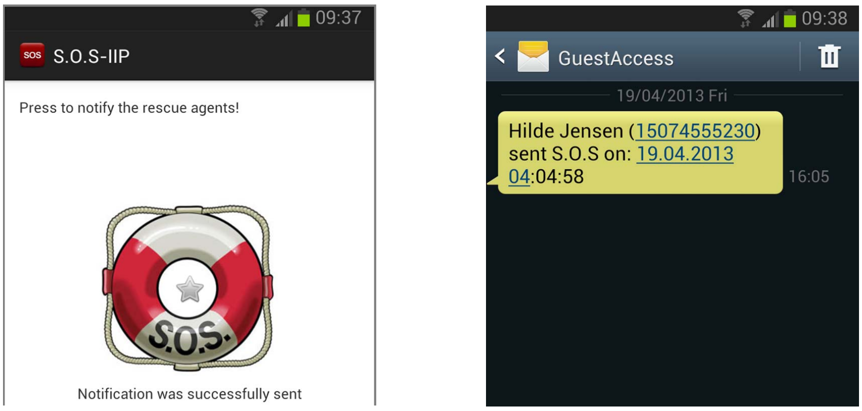


Figure E.7: SOS service screen shots

IV. Pros and Cons Evaluation

IV.1 Direct vs. Brokered Communications

IIP plays a broker role between service providers and consumers in a SOA environment. A major advantage of such a broker is the removal of tight-coupling of direct point-to-point message exchanges between information providers and consumers. Supposing there are N different information consumers that are interested in using the information gathered and provided by an information provider. If the direct approach is used, the information provider would then need to maintain the list of those N different information consumers as well as establishing network connections to them, as shown in Fig. E.8(a). This is not a good approach as the information provider is an everyday object or a gateway of everyday objects which may have limited processing power or battery lifetime. On the other hand, if the brokered approach is utilised, such as using the IIP, the information provider only needs to list and maintain one network connection to the broker, as shown in Fig. E.8(b). The complexity ratio of maintaining peer-connections for the information consumer is N to 1 between the two approaches. Similarly, the information consumers do not need to connect to many different information providers to consume different types of information, but only to the broker, when the brokered approach is employed. From this perspective, the brokered approach gives a clear advantage to both information providers and consumers.

IV.2 Pull vs. Push Communications

Both direct and brokered approaches can be realised in pull and push fashion. In

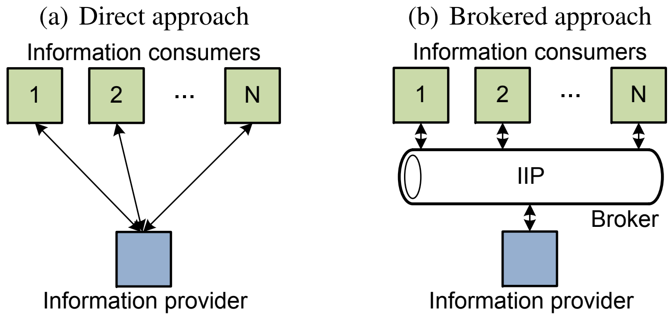


Figure E.8: Direct and brokered approaches of message exchange

general, pull mode requires an information consumer to request information from an information provider, whereas push mode lets an information provider send information to an information consumer as notifications. The main issue in pull mode is information freshness, where information consumers have to send requests for information to information providers (either directly or brokered) repeatedly to get information as real-time as possible. On the contrary, push mode enables information consumers to be notified in almost real-time whenever new information is gathered by information providers. However, if push mode is used in direct approach, information providers are burdened with notification tasks to all interested information consumers as described earlier. In this case, the IIP is advantageous since it follows the publish/subscribe messaging pattern, which is one of the most commonly used push modes for message exchange. One disadvantage of using a brokered approach, such as IIP, is that internal processing in the broker adds additional delay to the whole notification process.

An experiment with IIP was conducted to find out the significance of this internal processing time with different numbers of subscribers involved. Direct and brokered through IIP scenarios were compared with varying number of subscribers between 1 and 400. A multi-threaded tester application was developed for this experiment, acting as an information provider sending one publication per second HTTP POST request to not put too high load on the receiving application. Three machines were used following Fig. E.8(a) and E.8(b) as information provider, broker, and information consumer with similar specifications (Intel Core 2 Duo 2.4GHz, 8GB RAM running Linux Ubuntu 12.04), connected through a switch. Three measurements were conducted for each scenario, and each measurement was run for five minutes, resulting in 300 data for each measurement. Average notification times from an information provider to all subscribers were calculated, and the result from 900 measurement data for each scenario with their standard deviation is shown in Fig. E.9.

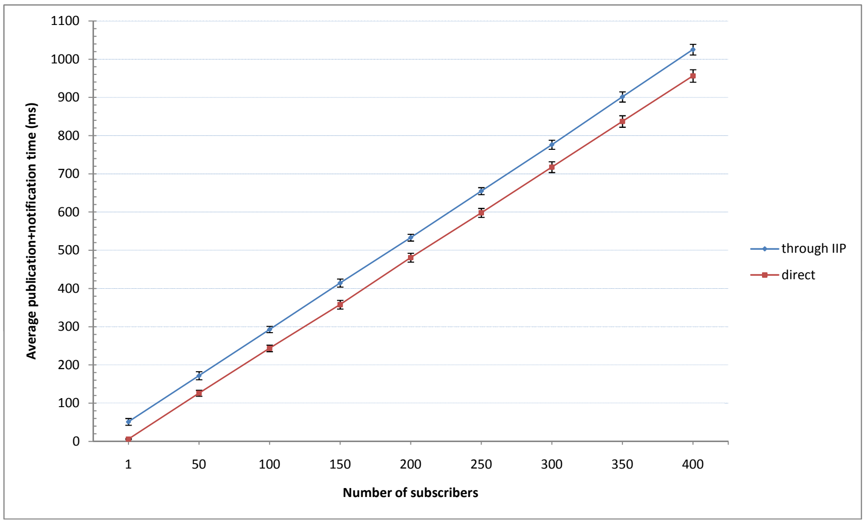


Figure E.9: Direct and through IIP average publication and notification time

It can be seen from Fig. E.9 that the processing time for both approaches tends to increase in a linear fashion with an increased number of subscribers. It can also be seen that the average processing time for brokered approach through the IIP was higher compared to the direct approach due to internal processing of the IIP. On the other hand, the internal processing time becomes less significant with the increasing number of subscribers, as it tends to be almost constant. The ratio between brokered and direct average times gets near to 1 as the number of subscribers increase, which alleviates the disadvantage of the brokered approach. This ratio is shown in Fig. E.10.

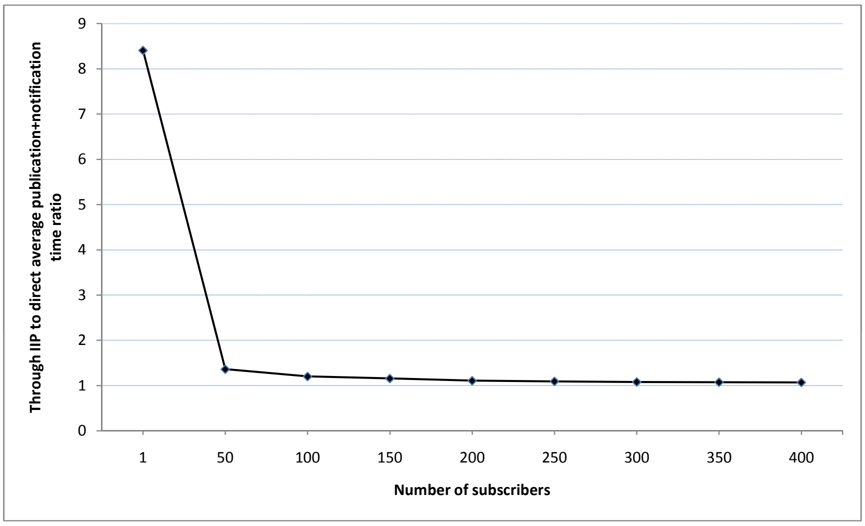


Figure E.10: Through IIP to direct average publication and notification time ratio

V. Conclusions and Future Work

A publish/subscribe information integration platform has been proposed, designed, and implemented by following a SOA paradigm which enables communications convergence between everyday objects (information providers) and Internet-based services (information consumers) in an event-driven fashion. RESTful Web service interfaces are used for exposing the platform's functionalities, interfacing both information providers and consumers, and HTTPS as well as HTTP Basic authentication are used as security measures. The platform can be used for provisioning various services. Two service prototypes within the healthcare domain (remote health monitoring and SOS services) have been developed as proof-of-concept. Message delivery in a brokered approach, such as using IIP, reduces the processing burden of information providers in particular, and also information consumers in general, compared to its direct counterpart. Conducted experiments showed that the additional processing load in the broker is negligible when many subscribers (information consumers) exist.

A more granular access control for different information providers and consumers will be carried out as the next step of research, and extension of the platform to provide actionable resources (e.g. controlling actuators) will be developed.

References

[1] F. Mattern and C. Floerkemeier, "From the Internet of Computers to the Internet of Things," *Lecture Notes in Computer Science*, vol. 6462, pp. 242-259, 2010.

[2] K. Ashton, "That Internet of Things Thing," *RFiD Journal*, vol. 22, pp. 97-114, June 2009.

[3] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, and D. F. Ferguson, "Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging and More," Prentice Hall PTR, 2005.

[4] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," Ph.D. dissertation, University of California, 2000.

[5] R. Strom, G. Banavar, T. Chandra, M. Kaplan, K. Miller, B. Mukherjee, D. Sturman, and M. Ward, "Gryphon: An Information Flow Based Approach to Message Brokering," in *Proc. of International Symposium on Software Reliability Engineering,* 1998.

[6] P. R. Pietzuch and J. M. Bacon, "Hermes: A Distributed Event-Based Middleware Architecture," in *Proc. of 22nd International Conference on Distributed Computing Systems Workshops (ICDCSW),* Vienna, Austria, July 2002, pp. 611-618.

[7] G. Cugola, E. Di Nitto, and A. Fuggetta, "The JEDI Event-Based Infrastructure and Its Application to the Development of the OPSS WFMS," *IEEE Transactions on Software Engineering*, vol. 27, no. 9, pp. 827-850, September 2001.

[8] A. Rowstron, A.-M. Kermarrec, M. Castro, and P. Druschel, "SCRIBE: The Design of a Large-Scale Event Notification Infrastructure," *Lecture Notes in Computer Science*, vol. 2233, pp. 30-43, 2001.

[9] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf, "Achieving Scalability and Expressiveness in an Internet-Scale Event Notification Service," in *Proc. of 19th Annual ACM Symposium on Principles of Distributed Computing (PODC),* Portland, OR, USA, July 2000, pp. 219-227.

[10] Y. Huang and D. Gannon, "A Comparative Study of Web Services-based Event Notification Specifications," in *Proc. of International Conference on Parallel Processing Workshops (ICPPW),* Columbus, OH, USA, August 2006, pp. 7-14.

[11] Y. B. D. Trinugroho, K. Rasta, T. H. Nguyen, M. Gerdes, R. Fensli, and F. Reichert, "A Location-Independent Remote Health Monitoring System utilising Enterprise Service Bus," *IADIS International Journal on WWW/Internet*, vol. 10, no. 2, pp. 88-106, December 2012.

Appendix F

Paper V

Title:	Information Integration Platform for Patient-Centric Healthcare Services: Design, Prototype, and Dependability Aspects
Authors:	**Yohanes Baptista Dafferianto Trinugroho**
Affiliation:	University of Agder, Faculty of Engineering and Science, Jon Lilletuns vei 9, 4879 Grimstad, Norway
Published in:	*Future Internet*, vol. 6, no. 1, March 2014.

Information Integration Platform for Patient-Centric Healthcare Services: Design, Prototype, and Dependability Aspects

Yohanes Baptista Dafferianto Trinugroho
Department of Information and Communication Technology, University of Agder, Jon Lilletuns vei 9, Grimstad, Norway

Abstract — **Technology innovations have pushed today's healthcare sector to an unprecedented new level. Various portable and wearable medical and fitness devices are being sold in the consumer market to provide self-empowerment of healthier lifestyle to the society. Many vendors provide additional cloud-based services for devices they manufacture, enabling the users to visualise, store, and share the gathered information through the Internet. However, most of these services are integrated with the devices in a closed "silo" manner, where the devices can only be used with the provided services. To tackle this issue, an Information Integration Platform (IIP) has been developed to support communications between devices and Internet-based services in an event-driven fashion by adopting Service-Oriented Architecture (SOA) principles and a publish/subscribe messaging pattern. It follows the "Internet of Things" (IoT) idea of connecting everyday objects to various networks and to enable the dissemination of the gathered information to the global information space through the Internet. A patient-centric healthcare service environment is chosen as the target scenario for the deployment of the platform, as this is a domain where IoT can have a direct positive impact on quality of life enhancement. This paper describes the developed platform with emphasis on dependability aspects, including availability, scalability, and security.**

Keywords—Internet of Things; integration platform; middleware; REST; publish/subscribe; availability; scalability; security; healthcare

I. Introduction

The term "Internet of Things" (IoT) was popularised at the MIT Auto-ID Center in 1999 where a group of people started to design and propagate a cross-company RFID infrastructure [1, 2]. Advancements in Information and Communications Technology (ICT) have enabled IoT's vision to be realised by turning everyday objects to connected objects [3, 4], so that the information gathered (or "sensed") by these objects can be used in various different services. Everyday objects can be employed to capture and create information from the physical world instead of relying purely on people as normally done in traditional information systems [1]. This is mainly achieved by using RFID and sensor technologies. The ability to react to events in the physical world automatically not only opens up new opportunities for dealing with complex or critical situations, but also enables a wide variety of business processes to be optimised. The real-time interpretation of data from the physical world can lead to the introduction of various novel services and may deliver substantial economic and social benefits [4].

Although localised services, utilising information gathered from nearby objects, can be useful in many different scenarios, the global pervasiveness of things can only be realised when these everyday objects are connected to the Internet [5]. The Internet has become the de facto standard backbone for deploying services beyond time and space barriers, which enables people or other services to consume them 24/7 from all over the world. Cloud computing has pushed the boundary even further, enabling developers with innovative ideas to develop and deploy services without large capital outlays in hardware [6]. Myriad Internet-based services can make use of the collected information from various different everyday objects for value-added functionalities. However, everyday objects may have limitations in terms of processing power or battery lifetime, which makes it infeasible to incorporate a full TCP/IP stack [7] in order to communicate with the current Internet infrastructure. Other more power-preserving communications protocol stacks (e.g. Bluetooth, ZigBee, ANT) are more commonly used in embedded devices. In order to be connected to the Internet, additional gateway devices that have implemented full TCP/IP stacks are needed. These gateway devices should have at least two network interfaces, one facing the connected objects and another one facing the Internet, and physically can range from dedicated servers to mobile devices (e.g. smartphones, tablets). The latter is particularly useful for scenarios that involve mobility of the users [8]. With the increasing needs of everyday objects to be connected to the Internet, new technologies have been developed to extend the Internet to small devices [9, 10], such as IPv6 over Low Power Area Networks (6LoWPAN) [11, 12] and GLoWBAL IPv6 [13].

When everyday objects are connected to the Internet (e.g. through a smartphone gateway), an application-layer protocol is needed to communicate with Internet-based services that are interested in using the gathered information. Within the Web domain, Hypertext Transfer Protocol (HTTP) [14] has been widely used to exchange contents over the Internet [15] since the inception of the World Wide Web (WWW) [16]. Web services, which are software systems designed to support interoperable machine-to-machine interaction over a network, normally use HTTP to convey messages as well. This is true for both traditional Web services [17] and RESTful Web services [18]. The majority of Internet-based services provide Web service interfaces to enable message exchange with external systems. Thus the HTTP protocol, combined with Web service approach, are good combination to be used for message exchange in the application layer.

The IoT plays an important role in healthcare, for example in general remote vital sign monitoring of patients [19] as well as in specific chronic disease treatment such as diabetes therapy management [20]. Within the personal healthcare sector, many portable and wearable medical and fitness devices are being pushed to the consumer market by various vendors. This can be seen as a positive trend towards self-empowerment of healthier lifestyle, and enables healthcare workers to more efficiently keep track of their patients' health conditions by means of telecare, if such a feature is provided. Many vendors provide additional online services for devices they sell, enabling users to better visualise, store, and share the gathered information from the devices through the Internet. However, many of these services are integrated with the devices following a closed vertical "silo" approach [21], where the devices can only be used with the provided services, and different services from other vendors cannot make use of the gathered information. The main disadvantage of this situation is the inability to combine information gathered from different devices produced by different vendors for better reasoning and decision making [22, 23]. To solve this issue, open interfaces (e.g. Web service interfaces) have to be provided by device vendors so that service developers can incorporate the information collected from the devices in their services.

A common way to integrate devices and Internet-based services is to directly exchange messages between the two parties in a point-to-point manner. The downside of this approach is that devices which "sense" new information should deliver it to all services that are interested in consuming it, either through a push approach from the devices or in a pull fashion from the services. This can be a major drawback from an energy efficiency standpoint as portable and wearable connected devices commonly run on batteries, and thus, sending similar information to many destinations (i.e. services) will lead to shorter lifetime of the devices. With the proliferation of mobile cloud computing usage in recent years [24, 25, 26], a brokered approach with a service broker being deployed in the cloud can be a good alternative since the devices need to connect to the Internet in order to communicate with Internet-based services anyway. The service broker will handle the message delivery tasks to all interested services, so that the devices only need to send the collected information once. A publish/subscribe messaging pattern is advantageous in such a broker so that services interested in specific information can subscribe to that particular information and get notification from the broker whenever new information is available. However, such a service broker can be seen as a single point of failure since various devices and services rely on it, and thus its dependability is very crucial. An Information Integration Platform (IIP), which acts as a service broker between connected devices and Internet-based services, has been proposed and developed as a prototype. Several services within the healthcare domain, mainly related to telecare, have been developed on top of the platform as well. This paper will briefly describe the main functionalities of the platform while focusing more on its dependability aspects. Prototype services on top of it will be described to put the platform into a deployment context within the healthcare domain.

II. Related Work

There are several existing works related to the proposed platform that have been conducted in the past. This section describes some of them briefly.

II.1 Publish/Subscribe Middlewares

The publish/subscribe messaging pattern was introduced more than a decade ago, and is still considered to be one of the most important communications mechanisms as it is well adapted to the loosely coupled nature of distributed interaction in large-scale applications. Subscribers have the ability to express their interest in an event, and are subsequently notified of any event which is generated by a publisher and matches their registered interest [27]. This complies with event-driven architecture where an event is asynchronously propagated to all subscribers.

Many existing middleware prototypes make use of the publish/subscribe messaging pattern both in research and production arenas. In the research domain, for example, there are Gryphon [28], Hermes [29], JEDI [30], Scribe [31], and SIENA [32]. Although these systems provide relatively complex topic-based, content-based, or type-based subscription schemes, they mainly provide progamming language-specific application programming interfaces (APIs), such as for Java or C++.

Within the Web services arena, WS-Eventing and WS-Notification are two major competing specifications for "big" Web services to incorporate the publish/subscribe message exchange paradigm. Although their architectures differ from each other, convergence between the two was realised to a certain degree [33]. WS-Messenger [34] is a middleware prototype that aims to mediate WS-Eventing and WS-Notification specifications by enabling the utilisation of existing messaging systems to provide scalable subscription management and message delivery.

However, there is no standard initiative for RESTful Web services to include publish/subscribe for supporting an asynchronous communcations model. Various projects proposed different approaches to add a publish/subscribe functionality to RESTful Web services-based middleware. UnivPS was proposed in [35] which provides publish/subscribe functionality for a presence service by using REST interfaces in the telecommunications domain. The authors in [36] proposed a RESTful gateway for integrating Smart Meters in future houses that makes use of topic-based publish/subscribe mechanisms through Web push techniques so that any computing device that runs a Web server can be a subscriber that is notified through HTTP POST requests. The Å publish/subscribe framework [37] was proposed following a content-based subscription scheme where event patterns can be defined using scripts with many common script languages supported. The RESTful paradigm is used together with a client event cache so that clients can periodically query the HTTP endpoint. PubSubHubBub [38] is a protocol for publish/subscribe messaging on the Internet which extends Atom [39] and RSS [40] protocols for data feeds. It provides push Atom/RSS update notifications instead of requiring clients to poll the whole feeds. Constrained Application Protocol (CoAP) [41] is a RESTful application layer protocol that is intended for use in resource-constrained nodes and networks. Through the Observe option [42], a client can conditionally observe a resource on a CoAP server, only being informed about state changes meeting a specific condition or set of conditions.

II.2 Event-Driven Service-Oriented Architecture

The SOA paradigm has been adopted in different application domains as it supports modularisation of service components to be reused in heterogeneous platforms, especially using Web services technology [43]. Traditional Web services approach uses point-to-point mechanism between service providers and service consumers, where service consumers normally request some information from service providers, and service providers provide the requested information in a synchronous manner. From this perspective, service providers become information providers. When the number of services increases drastically, this approach can potentially create management and integration issues, especially when the information providers are everyday objects.

The Event-Driven Architecture (EDA) [44] defines a methodology for designing and implementing applications and systems in which events are transmitted between decoupled software components and services. Information captured from objects can be treated as events which may trigger some operations in different services. If a point-to-point approach is used, an event should be propagated to all services that are interested in using it, and the event source is in charge of carrying out this task. A brokered approach is better when the number of services that are interested in a specific event grow significantly; a publish/subscribe mechanism fits well to fill the void.

The Enterprise Service Bus (ESB) [45] concept combines EDA and SOA approaches to simplify integration tasks, bridging heterogeneous platforms and environments. It facilitates interactions between service providers and service consumers both in synchronous and asynchronous manners. There is no standardised specification for ESB implementations. In general, it should support message routing, message transformation, protocol mediation, and event handling. Many ESB implementations provide various sophisticated functionalities for integrating new and legacy services and information systems. However, many of these functionalities require demanding programming efforts which make ESB as an integration platform require quite a steep learning curve.

In this paper, a publish/subscribe platform and its working prototype is presented, providing information through information channels for event-driven information dissemination from everyday objects to Internet-based services. The platform itself is designed to be flexible enough, following the SOA paradigm, to accept not only information from physical devices but also from any type of information provider. It aims to simplify the information integration process while still following the event-driven SOA concept as ESB does, so that the platform can be used out-of-the-box without modifying anything in the platform programmaticaly. This is mainly achieved by imposing strong constraint that both service providers and service consumers should exchange messages through RESTful Web services. Object gateways are responsible for encapsulating captured information from the objects as HTTP requests to be sent to the platform in case the objects cannot send HTTP requests directly. An overview of the platform's functionalities are discussed in the next section.

III. Conceptual Design

The IIP aims to bridge the communications between everyday objects (i.e. information providers) and Internet-based services (i.e. information consumers), acting as a service broker between the two entities. This broker is expected to break the information reusability issue in a vertical "silo" integration approach that has been chosen by many personal wearable device vendors, including within the healthcare sector; see Figure F.1(a).

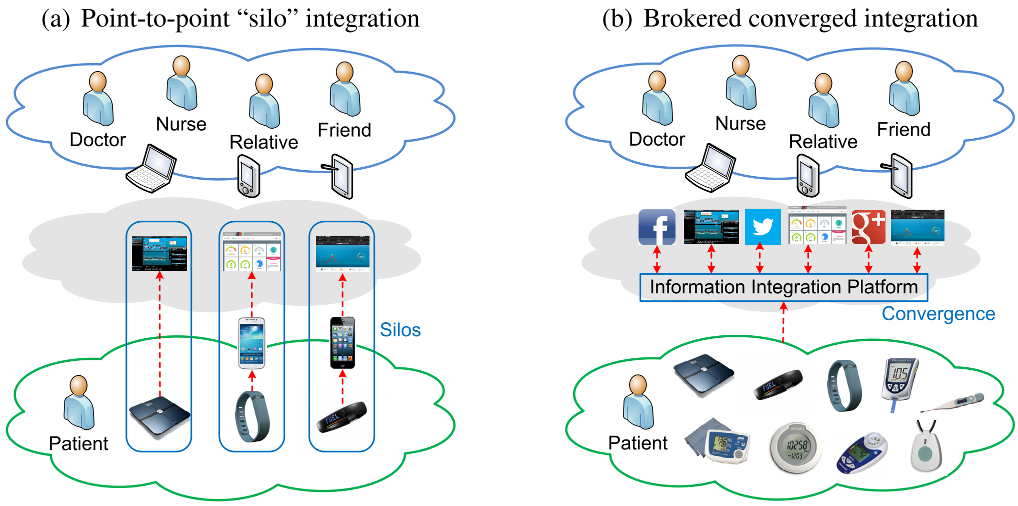


Figure F.1: "Silo" vs. converged integration approaches

Patients are faced with various healthcare-related devices from different vendors in their daily activities, and the information these devices gather is commonly only used by specific services provided by the devices' vendors. Other Internet-based services have no or very limited possibilities to utilise such information, so it is common that similar information is redundantly gathered by different devices for their own services. This tight coupling between devices and services can be solved if device vendors provide open APIs which enable service developers to make use of the gathered information in their services. However, the integration normally still follows a point-to-point approach, as shown in Figure F.1(a), where each service is directly communicating with each device that provides the information. The downside of such point-to-point integration is, from information providers' (i.e. devices) perspective, that they have to send newly gathered information to different services that are interested in using it. This is particularly an issue for battery-powered wireless devices. A brokered approach, as shown in Figure F.1(b), tackles this issue by delegating the information distribution task to the broker, so that information providers only need to send newly collected information once to the service broker. The service broker provides convergence for information gathering from various different devices that the patients encounter.

III.1 General Architecture

The IIP, which plays the service broker role, primarily aims to be deployed in the cloud for global reachability, although it is possible to deploy it in closed environment such as in smart homes. RESTful Web service interfaces are used facing both information providers (i.e. devices) and information consumers (i.e. Internet-based services) as they are widely used within the Web domain. Figure F.2(a) shows the main functionalities of the IIP.

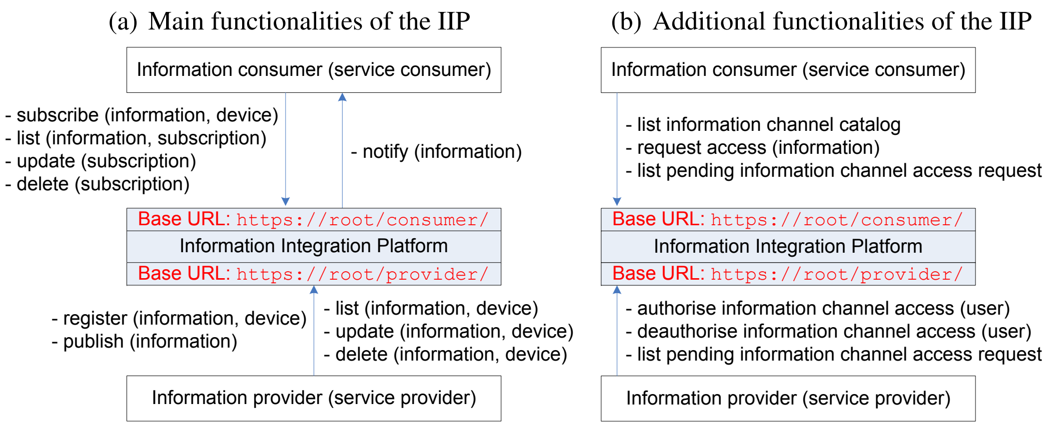


Figure F.2: Functionalities of the IIP

In general, the IIP provides RESTful Web service interfaces for both information providers (service providers from the SOA standpoint) and information consumers (service consumers from the SOA perspective) that manage the information flow from information providers to information consumers. These resources are divided into two categories with two main Web resource base Uniform Resource Identifiers (URIs), namely `https://root/provider/` and `https://root/consumer/`. The `root` part of the URI refers to the domain of the specific IIP deployment. Information is managed in different information channels, and information providers should initially create information channels before being able to pass through information they gather. The IIP provides resources for information providers to register new information channels with varying parameters, to list their information channels, to update/modify their information channels, to publish new information to their information channels, and to delete/remove their information channels. In addition, the IIP provides resources for information consumers to list existing information channels, to subscribe to existing information channels, to update/modify their subscriptions to existing information channels, and to delete/remove their subscriptions to existing information channels. When an information consumer subscribes to an existing information channel, it should provide a notification Uniform Resource Locator (URL) which acts as an end-point for IIP to deliver notifications of newly published information from an information provider who owns that particular information channel. An information provider can register many information channels, but an information channel can only be associated with one information provider (i.e. the owner). This makes the relationship between information provider and information channel a one-to-many relationship. On the other hand, an information channel can be accessed/subscribed to by many information consumers, and an information consumer can access/subscribe to many information channels. This makes the relationship between information channel and information consumer a many-to-many relationship. These relationships are shown in Figure F.3. An article, which was written earlier by the author, describes how these functionalities work in a more detailed manner [46], including sequence diagrams and the contents of message exchanges. This work is a continuation of the author's previous work, addressing several aspects in the future work section of the previous article.

In addition to the main functionalities as previously described, the IIP provides resources for access control between different credentials within the IIP, as shown in Figure F.2(b).

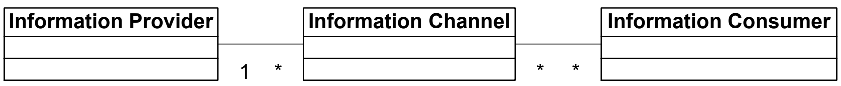


Figure F.3: Relationships between information provider, information channel (in the IIP), and information consumer

Since the IIP acts as a broker between information providers and information consumers, dependability becomes a crucial aspect that should be investigated. There are several definitions of dependability, but in general it is commonly recognised as an integrative concept that encompasses different attributes. Littlewood and Strigini [47] suggested that dependability attributes include reliability, safety, security, and availability. Avižienis et al. [48] defined attributes of dependability to comprise availability, reliability, safety, confidentiality, integrity, and maintainability. In this latter view, security is not seen as a standalone attribute, but rather as a combination of three attributes, namely confidentiality, integrity, and availability. This is in line with the CIA triad model [49], which commonly acts as a fundamental guideline to help secure information systems by providing a measurement tool for security implementations. The following subsections will cover dependability aspects of the IIP that relate to security and availability, including scalability.

III.2 Security and Privacy

Information is passed around between information providers and information consumers through the IIP, and thus communications between the three entities should be secured. Since REST interfaces are used in IIP, the communications security relies heavily on the application layer protocol used by these REST interfaces. In contrast to traditional Web services technology that has a solid standardised security stack such as WS-Security [50], RESTful Web services do not have predefined and/or standardised security methods. Although RESTful Web services were initially designed to be technology-agnostic [18], it has been commonly associated with the HTTP protocol. Thus, many of its security features are mainly inheritted or simply adopted from the ones used for HTTP-based applications. Transport Layer Security (TLS), in the form of HTTPS, has been the main ground for RESTful Web services security, which provides a secure point-to-point communications channel on top of the transport layer.

HTTP Basic Access Authentication scheme, which was initially specified in the HTTP/1.0 specification, provides a simple authentication mechanism to access resources on a Web server (based on URIs) by means of username and password. This scheme is not considered to be a secure method of user authentication as the username and password are transmitted through the network as plain text (unless used in conjunction with other secure transport mechanism such as HTTPS). Nevertheless, the combination of HTTPS and HTTP Basic authentication in many cases is enough for securing resources on a Web server, as everything being sent through the wire is encrypted.

From IIP's perspective (as shown in Figure F.2(a)), information providers are applications that relay information from devices used by patients to information channels in the IIP. Likewise, information consumers are applications that consume/use information by subscribing to information channels in the IIP. To strengthen the privacy of information being exchanged between different applications through the IIP, access control to information channels should be maintained by the IIP. Identity-based access control is utilised, and an access control matrix is maintained. Information channels are registered by information providers (i.e. the owners), and each information channel has exactly one owner who can publish information (add new information) to the registered information channel (write access to the information channel). The access control matrix is used for authorising information consumers to access/subscribe to information channels (read access to information channels). An information provider has the privilege to specify which credentials (e.g. usernames in HTTP Basic authentication scheme) have read access (i.e. can subscribe) to information channels it owns. Table F.1 shows an example of an access control matrix maintained by the IIP.

	channel_1	**channel_2**	...	**channel_n**
username_1	YES	YES		NO
username_2	NO	YES		YES
⋮				
username_m	NO	NO		YES

Table F.1: Access control matrix for read access to information channels

The rows in Table F.1 represent capabilities of users (information consumers) in the IIP, and the columns represent access control lists of information channels. Information consumers can list all information channels (by utilising a catalogue service provided by the IIP), but they can only access/subscribe to information channels listed in their capabilities lists. Since one of the IIP's main responsibilities is to manage information channels, the access control matrix can be simplified to access control lists only (i.e. the columns in Table F.1). Whenever an information provider adds/removes a user (i.e. an information consumer) from/to its allowed user list of a specific information channel it owns, the IIP will update the access control list of the corresponding information channel. Figure F.4 shows the relationship between an information provider and its registered information channels with their allowed user lists. The IIP provides additional functionalities, as shown in Figure F.2(b), for managing access to information channels. Information consumers are provided with resources for listing information channels (information channel catalogue service), requesting access to information channels, and listing their pending access requests to information channels. Information providers, on the other hand, are provided with resources for authorising access to information channels that belong to them (adding allowed users to their allowed user lists), deauthorising access to information channels that belong to them (removing allowed users from their allowed user lists), and listing pending access requests to their information channels.

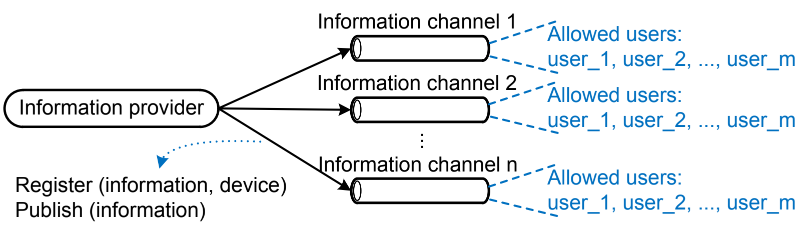


Figure F.4: Allowed user lists of information channels belonging to an information provider

The identity-based access control to different information channels is meant to be used by applications (both information providers and information consumers).

User identities in this case are application identities, not users of the applications (e.g. patients, nurses). These identities are used for controlling what information different applications can or cannot use, where the information is generated by other applications. However, these identities, which are used in the IIP, can also be directly mapped to user identities in the information consumer applications if required (e.g. a patient ID as a username in the IIP). Finer degrees of access control can be implemented by information consumer applications, for example utilising a Role-Based Access Control (RBAC) to group individual users of the applications.

III.3 High Availability and Scalability

Despite the positive aspects of the IIP as an integration platform between things and services that provides information distribution convergence as shown in Figure F.1(b), centralised service brokers such as the IIP are architecture-wise a single point of failure since all information providers and information consumers communicate with and rely on it. High availability becomes a crucial factor for successful deployment of the IIP to ensure services receive information from devices in a timely manner and continue to work properly (i.e. reliable). High availability is not a new topic in itself, as typical client-server systems require servers to run 24/7 with uptime as close to 100% as possible, accepting requests from client applications. Redundancy is the key to high availability, where service components are duplicated in different nodes, so that if one service component fails, another similar component will take over its tasks. In general, high availability can be achieved in either master/slave or master/master mode. In master/slave mode, a server instance (i.e. the master) is in charge of providing services to the clients' requests while another server instance (i.e. the slave) is running idle. When the master instance fails, a monitoring entity (i.e. the manager) will handover the master's tasks to the slave instance. On the contrary, all server instances are treated as masters in master/master mode, all providing services to client requests. The manager is responsible for monitoring and handling any conflict that might arise between concurrent changes made by different master instances. A load balancer can be added in front of master instances so that client requests can be distributed according to the processing capability of the master nodes (e.g. requests are evenly distributed among master nodes when they have similar processing capability).

The IIP utilises master/master mode for high availability with an additional load balancer as proxy for handling client requests (both from information providers and consumers). This will make the IIP seem a single entity from both information providers' and consumers' perspectives, but its components are redundantly distributed among different nodes. This approach allows the scalability aspect to be incorporated as well, enabling new nodes to be added when the current serving nodes are reaching their peak (i.e. fully loaded) in handling incoming requests. Scalability becomes important when the number of information providers and consumers using the IIP is not fixed. IIP nodes should be flexible enough to scale horizontally by the addition of new commodity servers when the number of participating information providers and consumers grows.

With regard to high availability and scalability, a simple 3-layer system architecture is used by the IIP for deployment as shown in Figure F.5.

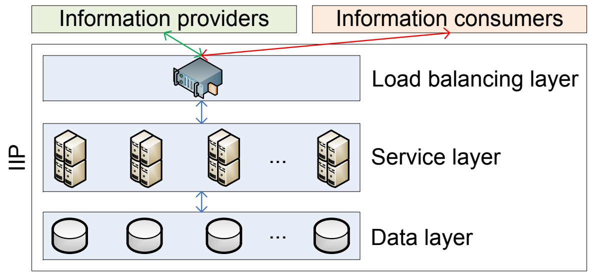


Figure F.5: 3-layer system architecture for IIP deployment

The load balancing layer acts as a proxy service where both information providers and consumers send requests to. The requests are then forwarded to one of the application servers in the service layer that hosts the main logic of the IIP (i.e. the IIP application) based on the load balancing criteria maintained by the load balancer. In the service layer, the IIP's functionalities, as shown in Figure F.2(a) and F.2(b), are realised and exposed through RESTful Web service interfaces. All information that needs to be stored, such as information channels, allowed user lists, information channel subscriptions, and the actual information of the information channels, are persisted in the data layer. By adopting this 3-layer architecture, the IIP's components can be made redundant, and can be scaled up according to deployment needs (e.g. add application nodes when more processing capability is needed, add data nodes when bigger storage capacity is required).

IV. Prototype Implementation

The IIP and several service prototypes within the healthcare domain have been implemented following the conceptual design described in the previous section by utilising open source software. This section will describe the prototype implementations that have been conducted as proof-of-concept of the conceptual design.

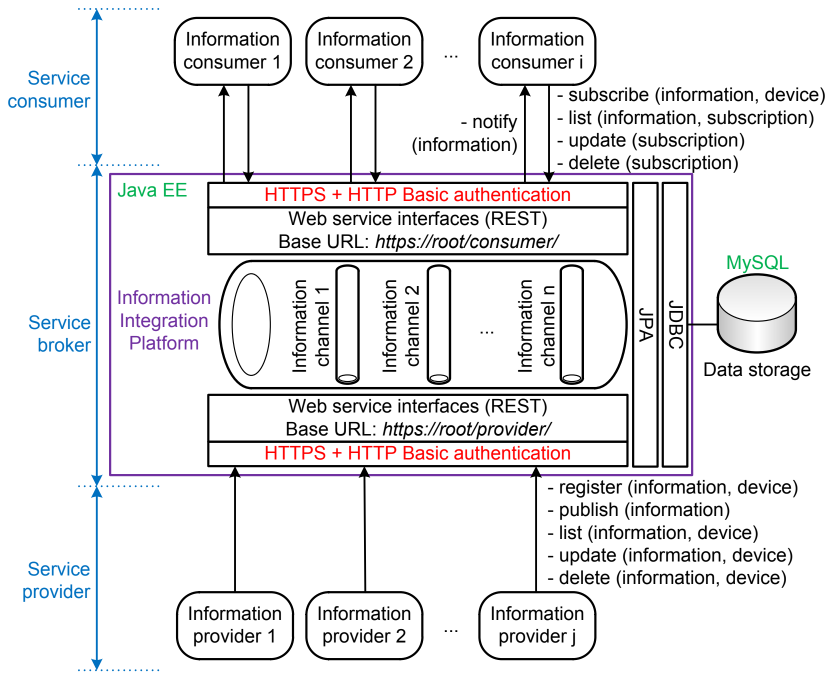
IV.1 The IIP Prototype

The current prototype of the IIP has been implemented as a Java Enterprise Edition (EE) 6 application in the service layer following Figure F.5, deployed on Glassfish 3.1.2.2 open source application server. Open source MySQL Cluster 7.3.2 is used for data storage in a clustered environment for high availability and scalability in the data layer, and the Java Persistence API (JPA) is utilised for mapping relational tables in the database to entity objects in the application through Java Database Connectivity (JDBC). EclipseLink 2.3.2 (JPA 2.0) is used as the JPA provider. HTTPS is employed for encrypting all message exchanges between the IIP and both information providers and consumers, and HTTP Basic authentication is utilised for simple authentication to access the exposed Web resources in the IIP. Figure F.6(a) shows the prototype implementation architecture of the IIP.

IV.1.1 Information Channel Access Management

Information channels are managed internally by the IIP, and they can be created (registered), modified (updated), and deleted (removed) dynamically through the provided REST interfaces without having to recompile and redeploy the application. In the current prototype, the representation of an information channel can be retrieved in XML format as depicted in Figure F.6(b). Each information channel maintains its own allowed user list, and the HTTP Basic authentication's credentials are directly used for accessing information channels. As described in the previous section, these credentials are used by client applications to authenticate themselves when communicating with the IIP. IIP users are categorised into three groups, namely *admin*, *provider*, and *consumer*. *Admin* users are mainly responsible for adding, removing, and assigning groups to users of the IIP. Users in the *provider* group are by default added to the *consumer* group since information providers should be able to consume their own information (acting as information consumers). When an information provider creates (registers) a new information channel, its own username is added by default to the information channel's allowed user list as the first information consumer that is allowed to access the information in the information channel. Users in the *consumer* group that are not included in the *provider* group are strictly information consuming-only users, and they cannot create (register) new information channels. Information consumers can retrieve a list of available information channels through the information channel catalogue service provided by the IIP at URL `https://root/consumer/catalogue/`, which can be requested by using HTTP GET as shown in Figure F.7(a). Only *infoId*, *name*, and *description* of information channels are returned in the cat-

(a) The IIP prototype implementation architecture



(b) Information channel representation example in XML format

```xml
<informationChannel>
    <infoId>info:782527135</infoId>
    <name>GPS</name>
    <description>GPS information channel</description>
    <deviceId>dev:125503922</deviceId>
    <creationDate>2013-05-05T21:37:12</creationDate>
    <lastupdateDate>2013-05-05T21:37:12</lastupdateDate>
    <ownerUsername>provider1</ownerUsername>
    <parameters>
      <latitude />
      <longitude />
    </parameters>
    <allowedUsers>
      <username>provider1</username>
      <username>consumer2</username>
      <username>consumer1</username>
    </allowedUsers>
</informationChannel>
```

Allowed user list

Figure F.6: The IIP prototype architecture and information channel representation example

alogue, while the ownerships are not revealed. Information consumers can request access to information channels (read-only) by sending HTTP POST to URL `https://root/consumer/authorization/infoId/`, where *infoId* is a unique ID of an information channel. Figure F.7(b) shows a sequence diagram of an information consumer requesting access to an information channel.

Access requests to information channels are stored by the IIP, and a resource for listing pending access requests is made available to information providers through

(a) Information channel catalogue request sequence diagram

(b) Information channel access request sequence diagram

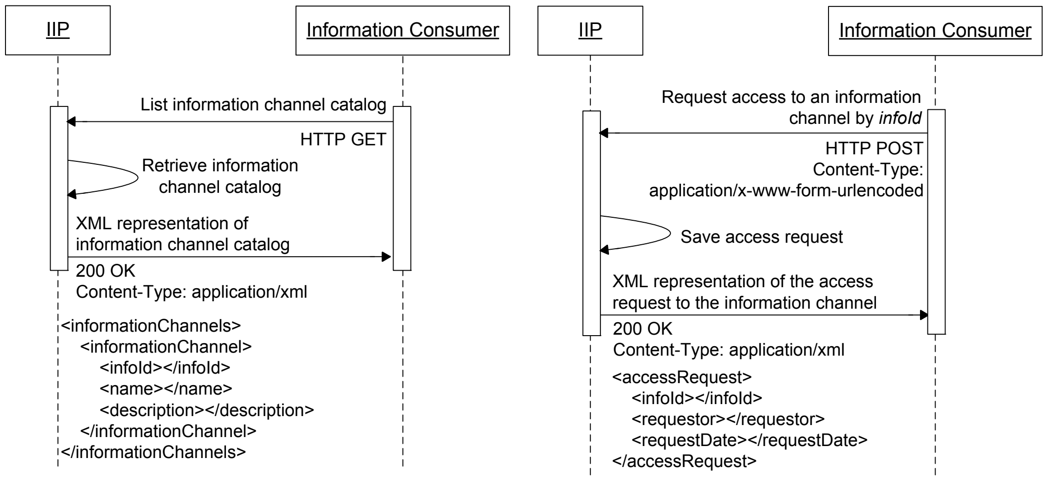


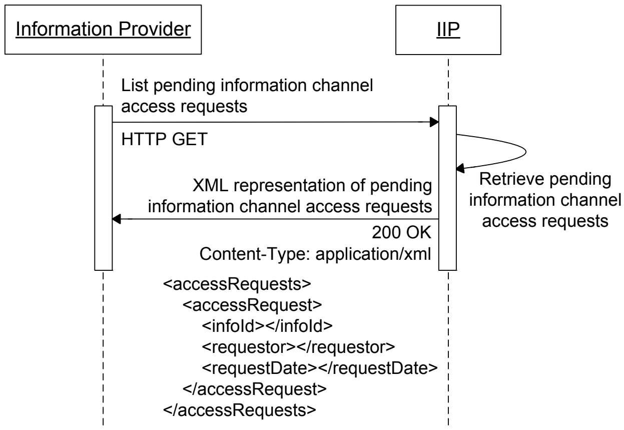
Figure F.7: Information channel catalogue request and access request sequence diagrams

URL `https://root/provider/authorization/` that can be accessed via HTTP GET request. This resource will return all pending access requests to information channels belonging to the caller (i.e. the information provider which accesses this resource) in XML format as shown in Figure F.8(a). Information providers can add information consumer users to their information channels' allowed user lists (for read access) by sending HTTP POST request to a resource provided by the IIP at URL `https://root/provider/authorization/infoId/`. The content of this request should be of type `application/x-www-form-urlencoded`, and contains a parameter called *username* which refers to the information consumer user being added to the allowed user list. This resource will return a representation of the affected information channel in XML format as shown in Figure F.8(b).

IV.1.2 High Availability and Scalability

In order to provide high availability of service, the IIP must run multiple redundant instances of service-providing entities so that, if one instance stops to function, other instances can still serve client requests. In relation to high availability, the IIP must also be able to scale to larger deployments in order to accommodate an increasing number of clients using its service. Application server clustering, which is suported by the Glassfish application server, can address the needs of both high availability and scalability. All application instances in a cluster, which can reside in different hosts, can be administered as a single unit, and user sessions can be automatically replicated between application instances in a cluster. However, fol-

(a) Pending information channel access requests listing sequence diagram



(b) Adding an information consumer user to allowed user list of an information channel sequence diagram

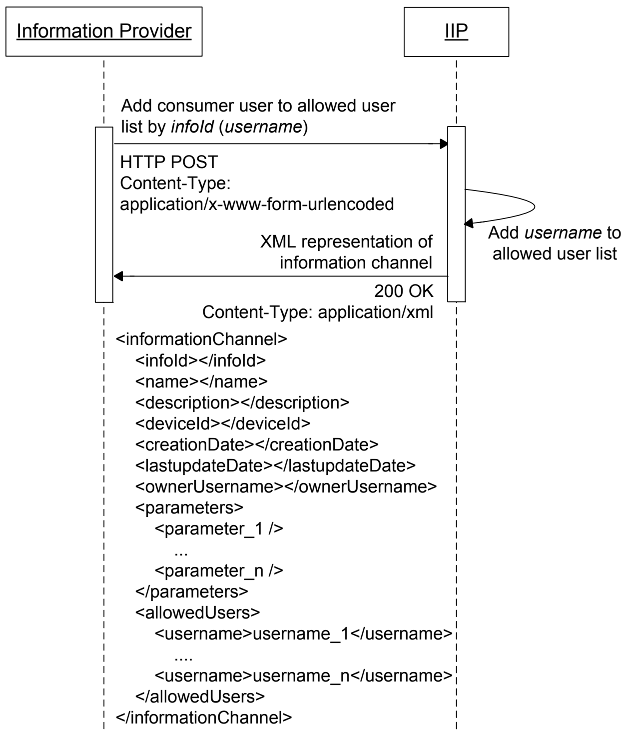


Figure F.8: Listing pending information channel access request and adding an information consumer to allowed user list sequence diagrams

lowing REST principles the services being provided by the IIP are stateless, so no client session from each request is maintained by the IIP. All server resource states are persisted in the database. From this point of view, clustering at the service layer (following Figure F.5) does not give much advantage except simpler administration.

The open-source Apache Web server with `mod_jk` module is used for load balancing requests coming from clients (both information providers and consumers as shown in Figure F.5), acting as a proxy. The load balancing factor is currently set to be equal for all IIP instances (hosted in different hosts) so that requests are forwarded equally (i.e. evenly distributed) among all IIP application instances. New IIP application instances in different hosts can be added (scaled up) in the service layer, and load balanced through this proxy.

In the data layer (following Figure F.5), MySQL Cluster is used for storing all of IIP's information. The IIP application instance in the service layer is responsible for handling requests from clients, but it does not store any state or information. Instead, it communicates with the back-end database to store information. MySQL Cluster follows a master/master architecture with no single point of failure, providing high availability and scalability of data storage and access. It makes use of a specialised storage engine called *NDBCLUSTER*, which is not used in the standard MySQL server, and employs a synchronous replication to guarantee that data is written to multiple nodes upon committing the data. Unlike in the service layer, data replication in the data layer is essential to maintain high availability of data, which in turn will make the IIP run properly (i.e. reliable). Three node types of MySQL Cluster are used in the prototype, namely *management*, *data*, and *SQL*. *Management* nodes are utilised for managing the entire cluster. *Data* nodes are mainly responsible for storing and retrieving data from memory and disk. *SQL* nodes are used for providing application access to the cluster. Application instances communicate with the MySQL Cluster through the SQL nodes by using JDBC, which is responsible for load balancing queries across the SQL nodes. Figure F.9 shows the current prototype's deployment architecture to accommodate high availability and scalability aspects.

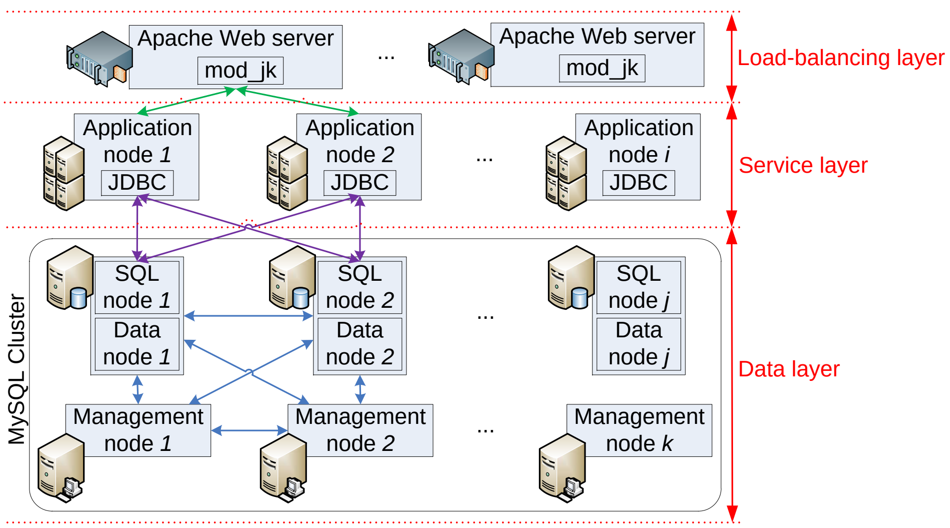


Figure F.9: High-available and scalable IIP prototype deployment

The current prototype consists of seven hosts (virtual machines) for deploying different nodes of the IIP: one for load balancer, two for application nodes, two for data and SQL nodes, and two for management nodes. By following the deployment architecture in Figure F.9, new nodes can be added when needed.

IV.2 Healthcare Services Prototype

Three prototype services have been implemented within the healthcare domain for patient-centric well-being as proof-of-concept. The services are developed as information consumers that make use of information from the IIP. Figure F.10 shows an end-to-end perspective of the developed service prototypes. Information is gathered from several devices relayed through an application gateway running on an Android smartphone, and forwarded to the IIP. All implemented services subscribe to information channels of interest, and provide services to different healthcare actors.

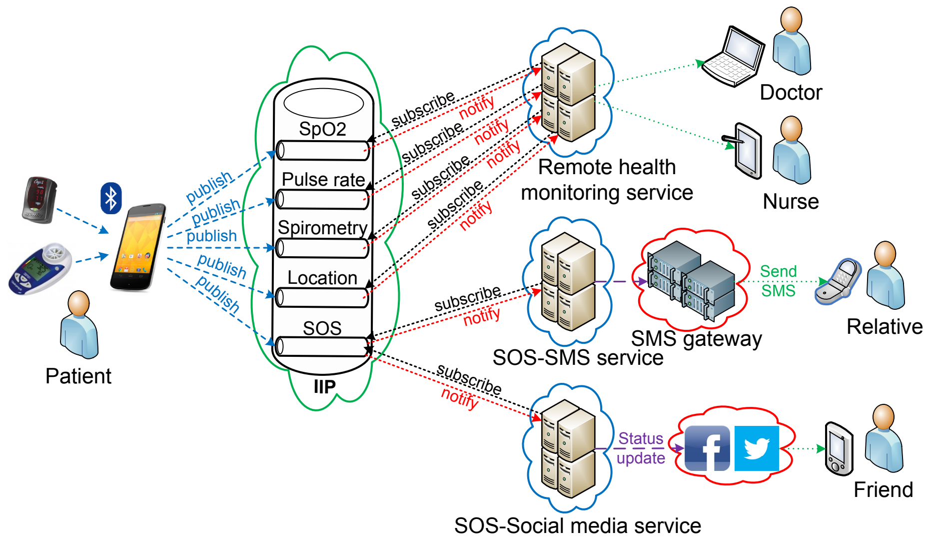


Figure F.10: End-to-end perspective of implemented service prototypes

Two commercial off-the-shelf wireless bluetooth medical devices are currently used for gathering the vital signs (i.e. Nonin Onyx II 9560 for SpO2 and pulse rate data, Vitalograph copd-6 bt for spirometry data).

IV.2.1 Remote Health Monitoring Service

The remote health monitoring service is implemented as a Web application that is accessible by the users (e.g. doctors, nurses) via Web browsers. It utilises Java servlet technology for handling incoming HTTP POST notifications from the IIP, and simple HTML with AJAX for presenting the output to the users. This service shows the latest measurements information from the remote patient, which currently includes pulse rate, blood oxygen saturation (SpO2), spirometry, and location. The service subscribes to these four information channels and receives almost real-time notifications from the IIP. At the patient's side, an android application has been implemented for gathering measurements from a wireless pulse oximeter and spirometer devices through bluetooth connections. This application sends all measurements to their corresponding information channels in the IIP. Location information is gathered directly by the Android application from the smartphone's built-in Global Positioning System (GPS) sensor. The Web application's user interface is shown in Figure F.11. By utilising this service, healthcare personnel can monitor the patient's health condition remotely. Since mobile devices are used at the patient's side, the patient is not restricted to measurements from a specific location.

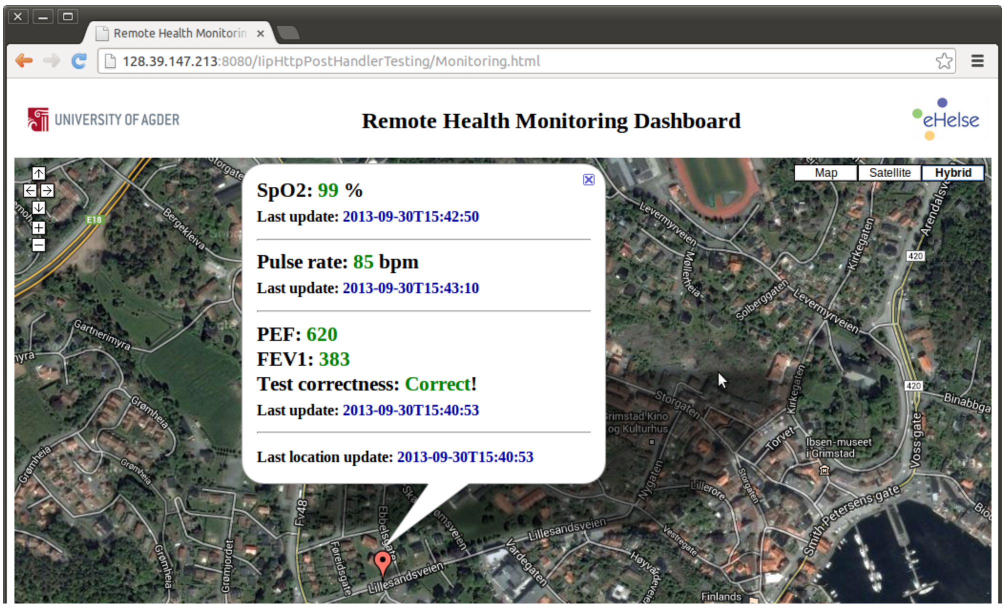


Figure F.11: Remote health monitoring service user interface

IV.2.2 SOS-SMS Service

This service provides emergency situation notifications via Short Message Service (SMS) to specific recipients, enabling the patient to inform selected persons (e.g. relatives, friends, nurses) that he/she needs immediate help. The ideal interface to the patient would be a physical alarm button that can be easily pushed in case of emergency. For simplification, an Android application that mimics an alarm button is used in the current prototype. An SOS information channel is registered in the IIP, and a server-side application, based on Java servlet technology, is implemented to subscribe and to handle HTTP POST notifications from the IIP whenever the patient sends SOS messages. A simple Web page is provided to the patient to administer which mobile numbers should be notified in case of emergency, and also to enable or disable this service. Figure F.12(a) shows a Web page for the patient to enable emergency notifications via SMS to a list of mobile numbers, and Figure F.12(b) shows a Web page confirming that the service is enabled. When the patient enables this service, the Web application subscribes to SOS information channel in the IIP for notifications of emergency events. In reverse, the Web application unsubscribes from SOS information channel when this service is disabled by the patient.

(a) Web interface for enabling specific mobile numbers to be notified in case of emergency

(b) Web interface confirming that SOS-SMS service is enabled

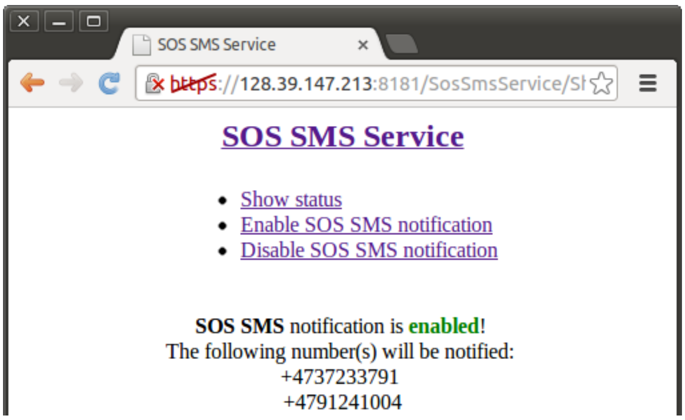
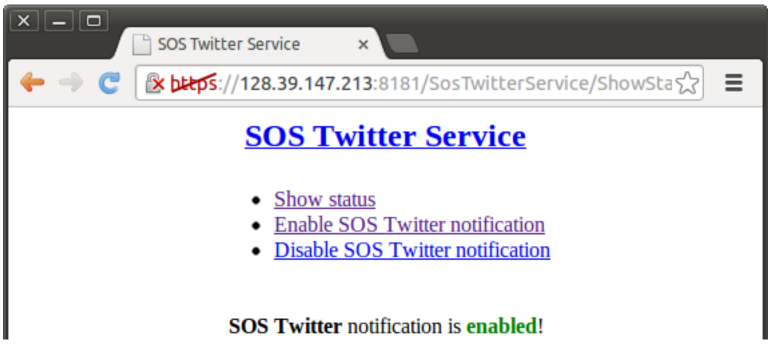


Figure F.12: SOS-SMS service prototype

IV.2.3 SOS-Social Media Service

This service is similar to the SOS-SMS service, except that it uses social media as emergency notification medium. Social media have been used extensively in the last couple of years, and can be extended further to support emergency situations for the patient. People within the patient's social media circle can be the first responders in case of emergency (e.g. due to close proximity to the patient). A prototype that uses a Twitter account to disseminate emergency information has been implemented, utilising the same SOS information channel used by the SOS-SMS service. The author's Twitter account is used in the current prototype. This service can be enabled or disabled by the patient through a Web page similar to the SOS-SMS service (the two services are deployed as different applications). Figure F.13(a) shows a Web page confirming that the SOS-Twitter service is enabled, and Figure F.13(b) shows a tweet of an emergency notification from the patient (i.e. the author).

(a) Web interface confirming that SOS-Twitter service is enabled



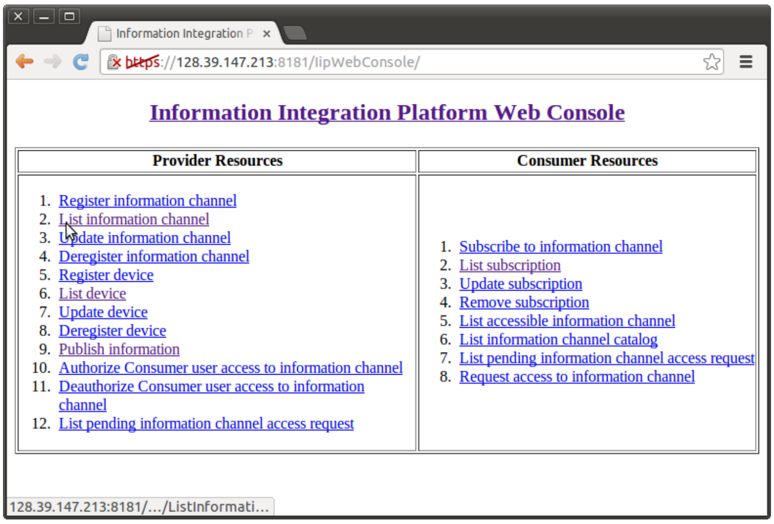
(b) A tweet of an emergency notification

Figure F.13: SOS-social media service prototype

IV.3 The IIP Web Console Prototype

The IIP's functionalities are exposed as RESTful Web services. While this can be seen as a positive way to enable brokered machine-to-machine (M2M) communications, human intervention is rather difficult to accommodate. Although communications between devices, the IIP, and services should ideally be automated, human involvement is sometimes needed, especially during the development process of new services. A Web console for the IIP has been implemented to ease management of resources in the IIP (e.g. information channels, devices, subscriptions, access control) for both information providers and consumers with simple HTML pages. This Web application utilises the same REST interfaces that information providers and consumers use. Figure F.14(a) shows the main page of the Web console which contains functionalities for both information providers and consumers. Figure F.14(b) shows a Web page for information providers to list their information channels. Each resource requires the user to enter HTTP Basic authentication credentials like normal applications do.

(a) The IIP Web console's main page



(b) A Web page for information providers to list their information channels

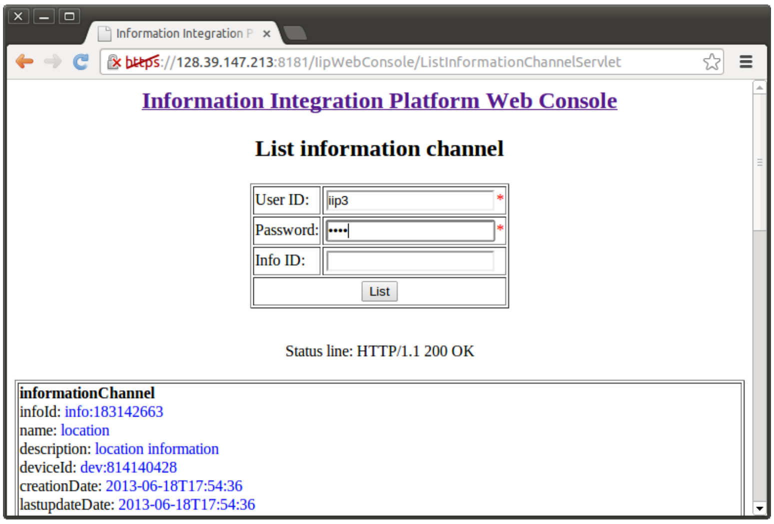


Figure F.14: The IIP Web console prototype

V. Evaluation

The implemented IIP prototypes, both standalone (i.e. all components in one dedicated host) and clustered (i.e. the components are spread in several different hosts), have been deployed and tested to work as intended in laboratory environment. This section will briefly present several aspects of the IIP that have been tested and verified, as well as performance benchmarking which provide inputs on how to further improve the current implementation.

V.1 Information Channel Access Management

As described in the previous section, HTTPS is used for securing message exchanges between information providers, consumers, and the IIP, by means of encryption. This will minimise the success rate of man-in-the-middle attacks. To maintain information privacy between applications, HTTP Basic authentication credentials are directly used in IIP for authorising applications to access different information channels. Information consumers that are not listed in an information channel's allowed user list will not be able to access or subscribe to that particular channel. Figure F.15(a) shows a test case where an HTTP POST request is sent from a test client application, acting as an information consumer, to the IIP for subscribing to an information channel with *infoId* `info:829055923`. The username which is used by the information consumer is listed in the information channel's allowed user list, and thus, a `200 OK` response is returned by the IIP, containing a representation of the subscription in XML format in its body. Figure F.15(b) shows another test case where similar HTTP POST request for subscription to the same information channel is sent from an information consumer. This time, however, the username used by the information consumer is not listed in the information channel's allowed user list, and therefore, the IIP returns a `403 Forbidden` response with an XML-formatted message in its body, informing that the username is unauthorised to subscribe to the targeted information channel. All messages in both tests (Figure F.15(a) and F.15(b)) are captured using Wireshark, and HTTPS is not used (otherwise all packets are encrypted and cannot be interpreted). It can be concluded from the conducted tests that the implemented information channel access management scheme by directly utilising HTTP Basic credentials works as intended.

The tests were conducted manually, since they were only intended to verify whether the access management implementation worked correctly. Automated unit testing could be used instead for a more formal way of verification, and protocol conformance testing could be applied as well to find out whether the implemented features comply to the chosen standards.

V.2 High Availability and Load Balancing

Seven hosts (machines) are used for deploying the clustered prototype of the IIP,

(a) An information consumer successfully subscribes to an information channel

Request

```
POST /IipDev/consumer/subscription/ HTTP/1.1
Content-Length: 112
Content-Type: application/x-www-form-urlencoded
Host: 192.168.1.3:8080
Connection: Keep-Alive
Authorization: Basic aWlwMzppaXAz

infoId=info%3A829055923&name=Sub&description=A+
subscription&notificationUrl=http%3A%2F%2Fwww.d
afferianto.info%2F
```

Response

```
HTTP/1.1 200 OK
X-Powered-By: Servlet/3.0 JSP/2.2 (GlassFish
Server Open Source Edition 3.1.2.2 Java/Sun
Microsystems Inc./1.6)
Server: GlassFish Server Open Source Edition
3.1.2.2
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Fri, 18 Oct 2013 16:33:26 GMT

<?xml version="1.0" encoding="UTF-8"
standalone="yes"?><subscriptions><subscription>
<subscriptionId>sub:184684472</
subscriptionId><name>Sub</name><description>A
subscription</
description><infoId>info:829055923</
infoId><notificationUrl>http://
www.dafferianto.info/</
notificationUrl><creationDate>2013-10-
18T18:33:26</creationDate><lastupdateDate>2013-
10-18T18:33:26</
lastupdateDate><ownerUsername>iip3</
ownerUsername></subscription></subscriptions>
```

(b) An information consumer is rejected when trying to subscribe to an information channel

Request

```
POST /IipDev/consumer/subscription/ HTTP/1.1
Content-Length: 112
Content-Type: application/x-www-form-urlencoded
Host: 192.168.1.3:8080
Connection: Keep-Alive
Authorization: Basic
c29tZXRoaW5nOnNvbWV0aGluZw==

infoId=info%3A829055923&name=Sub&description=A+
subscription&notificationUrl=http%3A%2F%2Fwww.d
afferianto.info%2F
```

Response

```
HTTP/1.1 403 Forbidden
X-Powered-By: Servlet/3.0 JSP/2.2 (GlassFish
Server Open Source Edition 3.1.2.2 Java/Sun
Microsystems Inc./1.6)
Server: GlassFish Server Open Source Edition
3.1.2.2
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Fri, 18 Oct 2013 16:43:57 GMT

<?xml version="1.0" encoding="UTF-8"
standalone="yes"?><AuthorizationError>You are
not authorized for this action!</
AuthorizationError>
```

Figure F.15: Wireshark captures of two test cases

where one is utilised as a proxy server for load balancing requests from both information providers and consumers, two are used for deploying the main applications that handle requests forwarded (load balanced) by the proxy/load balancer host, another two are employed for data and SQL nodes of the MySQL Cluster that act as the main storage for the IIP, and the last two are utilised for management nodes of the MySQL Cluster. This set-up conforms to Figure F.9, and can be considered as the minimum requirement for IIP's high availability where all nodes have exactly one redundant backup (except the load balancer). New nodes can be added when deemed needed. A test was conducted to review the general functionality of the load balancing between the redundant nodes being deployed. A test client application was developed, playing the role of information provider that keeps sending one publication message per second to one of its information channels through the load balancer host. Initially, both application servers' hosts are up and running, and the load balancer host distributes the requests evenly to both application servers. As can be seen in Figure F.16(a), the load balancer host (acting as a proxy) successfully forwards all requests evenly to both application servers' hosts (i.e. 53 and 52 requests for *worker1* and *worker2* respectively). One of the two application servers' hosts was then turned off during the test, leaving only one application server available for handling all requests. From Figure F.16(b) it can be seen that the load balancer host forwards all requests to the available application server's host (*worker2*), and *worker1*'s state was changed to error. From this test it can be concluded that the redundancy of the application nodes works well for providing highly available service while enabling new nodes to be added (horizontal scaling) and load balanced to serve incoming requests.

(a) Both application nodes are up and running

Name	Act	State	D	F	M	V	Acc
worker1	ACT	OK	0	50	1	6	53 (0/sec)
worker2	ACT	OK	0	50	1	7	52 (0/sec)

(b) One application node (*worker1*) is down

Name	Act	State	D	F	M	V	Acc
worker1	ACT	ERR	0	50	1	1	53 (0/sec)
worker2	ACT	OK/IDLE	0	50	1	1	148 (0/sec)

Figure F.16: Snapshots of `mod_jk` status worker

In the data layer, synchronous replication among data nodes is handled automatically by the MySQL Cluster, which is monitored and managed by the management nodes. Figure F.17(a) depicts a snapshot of a MySQL Cluster management client that describes the most current configuration of the cluster and the status of each node. In this set-up, any data committed from the service layer is inserted into both data nodes synchronously. One of the two data and SQL nodes' hosts (with IP address 192.168.1.7) was shut down. The MySQL Cluster then saves all data writes to only one existing data node, as shown in Figure F.17(b). When the host is up and running again, MySQL Cluster automatically synchronises all changes to the newly running data node.

V.3 Performance Benchmark

A dedicated multi-threaded client application was developed for performance benchmarking. The application acts as an information provider that constantly publishes information to the IIP with a predefined time interval for a certain period. Another client application, which subscribes to an information channel that belongs to the information provider application, was developed to receive notifications from the IIP, playing the role of information consumer. The main aim of this evaluation was to compare the performance of the standalone IIP prototype with the clustered version in terms of one complete flow of publication and notification average time. In addition, comparisons between the inclusion of the security and privacy scheme and also without it were conducted.

All experiments were carried out in a laboratory environment, where all hosts (machines) were deployed in an isolated network with a switch as the connecting

(a) Both data and SQL node hosts are up and running

```
Cluster Configuration
---------------------
[ndbd(NDB)]     2 node(s)
id=1 @192.168.1.6  (mysql-5.6.11 ndb-7.3.2, Nodegroup: 0, Master)
id=2 @192.168.1.7  (mysql-5.6.11 ndb-7.3.2, Nodegroup: 0)

[ndb_mgmd(MGM)] 2 node(s)
id=49@192.168.1.8  (mysql-5.6.11 ndb-7.3.2)
id=52@192.168.1.9  (mysql-5.6.11 ndb-7.3.2)

[mysqld(API)]   2 node(s)
id=55@192.168.1.6  (mysql-5.6.11 ndb-7.3.2)
id=56@192.168.1.7  (mysql-5.6.11 ndb-7.3.2)
```

(b) One data and SQL node host (192.168.1.7) are down

```
Cluster Configuration
---------------------
[ndbd(NDB)]     2 node(s)
id=1 @192.168.1.6  (mysql-5.6.11 ndb-7.3.2, Nodegroup: 0, Master)
id=2 (not connected, accepting connect from 192.168.1.7)

[ndb_mgmd(MGM)] 2 node(s)
id=49@192.168.1.8  (mysql-5.6.11 ndb-7.3.2)
id=52@192.168.1.9  (mysql-5.6.11 ndb-7.3.2)

[mysqld(API)]   2 node(s)
id=55@192.168.1.6  (mysql-5.6.11 ndb-7.3.2)
id=56 (not connected, accepting connect from 192.168.1.7)
```

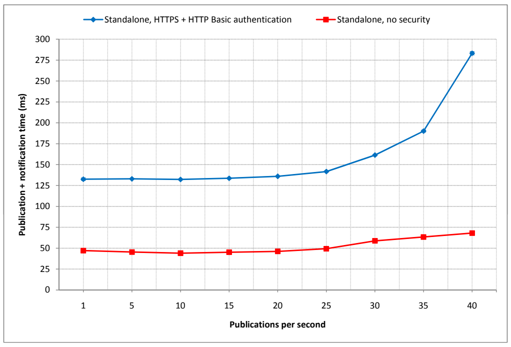
Figure F.17: Snapshot of `ndb_mgm` client showing the most current MySQL Cluster configuration

point (there is no connection to an external network such as the Internet). Experiments in a standalone set-up involve three hosts: one for the information provider application, one for the IIP (all-in-one, single point of failure), and one for the information consumer application. Experiments in a clustered set-up make use of nine hosts: one for the information provider application, seven for the IIP (as described earlier), and one for the information consumer application. All hosts that are used for deploying the IIP (both the standalone version and its clustered counterpart) have similar specifications (i.e. Intel Core 2 Duo 2.4GHz processor, 8GB RAM running Linux Ubuntu 12.04 LTS). Another two hosts that deploy the information provider and consumer applications also have similar specifications (i.e. Intel Core 2 Duo 2.4GHz processor, 4GB RAM running Linux Ubuntu 12.04 LTS). All application servers' configurations are kept similar with almost no optimisation from their default settings to ensure fairness in the comparisons.

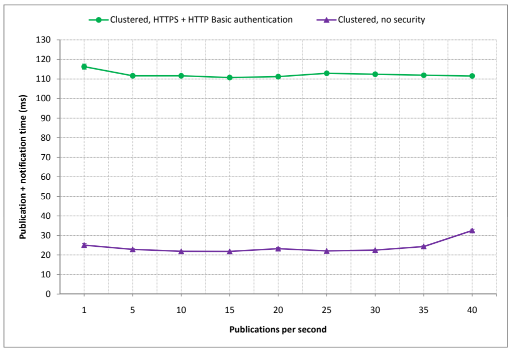
V.3.1 Publication Rate as Variable

Four experiments were conducted in this category. All variables are fixed except publication rates of the information provider application, which were varied between 1 and 40 publications per second, and all publication and notification messages contain only one parameter. The first experiment applied the security measures (i.e. HTTPS and HTTP Basic authentication) in a standalone IIP set-up, while the second experiment did not incorporate any security mechanism, also in a standalone set-up. The third and fourth experiments were conducted in a clustered IIP set-up, where security was applied in the third experiment and was ignored in the fourth experiment. Each measurement in all experiments lasted five minutes, generating 300 to 12000 data messages (depending on the publication rate). Each measurement was performed three times to ensure data consistency, and the first 1% of the captured data are removed from every measurement to avoid the start-up effect of the application servers in serving incoming requests. All measurement data are averaged, and plotted alongside confidence intervals at 95% confidence level.

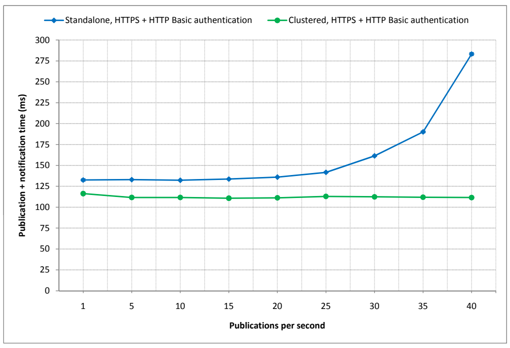
(a) Standalone secure vs. insecure



(b) Clustered secure vs. insecure



(c) Secure standalone vs. clustered



(d) Insecure standalone vs. clustered

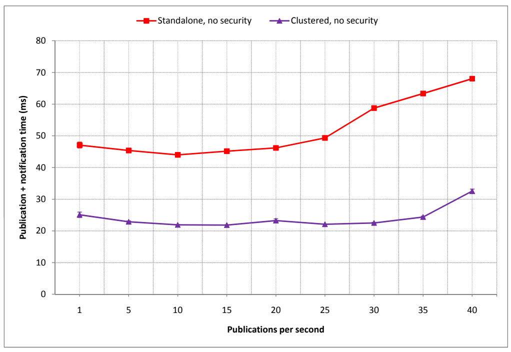


Figure F.18: Publication and notification time comparisons with publication rate as variable

Figure F.18(a) shows a comparison of total average publication and notification times between the secured and non-secured standalone IIP set-ups. It can be seen that the average difference at 1 publication per second is about 85 ms, which is the rough estimate of the security scheme's overhead in the standalone set-up. This average difference does not change much with the increase of publication rate until around 30 publications per second. From there, the difference gap grows larger significantly.

Figure F.18(b) shows a comparison of similar latency measurements as in Figure F.18(a), except that the compared experimental results are between secured and non-secured clustered versions of the IIP. In a clustered set-up, the starting difference between secured and non-secured implementations is about 90 ms, slightly higher than in the standalone set-up. This difference gap is maintained in a relatively stable manner up to 40 publications per second. From Figure F.18(a) and F.18(b) it can be seen that the clustered version of the IIP can handle the increase of publication rate better than its standalone counterpart, especially with the security scheme being applied.

A latency comparison between secured standalone and clustered IIP set-ups is depicted in Figure F.18(c). The average difference gap is stabled at about 20 ms up to around 25 publications per second, and then it grows more than two times. This can be viewed as the lesser ability of the standalone IIP set-up in handling faster publication rates compared to the clustered set-up.

Figure F.18(d) shows a latency comparison for publication and notification between non-secured standalone and clustered IIP set-ups. The average difference stays almost unchanged at about 20 ms from 1 to 25 publications per second, and the gap slightly widens as the publication rate increases.

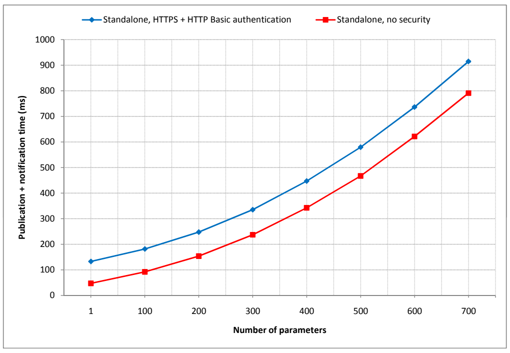
From the four experiments conducted in this category (i.e. publication rate as variable), it can be concluded that the clustered deployment of the IIP handles higher rates of publications better than the standalone set-up. Even at lower rates, the total average latency for publication and notification are smaller in the clustered set-up although only by a small margin. The inclusion of the security scheme adds additional overhead to the overall processing times in both standalone and clustered set-ups. The confidence intervals are very small compared to the mean values in all experiments. However, all experiments were carried out with only one parameter inside the publication and notification messages. In the next category of experiments, the number of parameters will be used as variable.

V.3.2 Number of Parameters as Variable

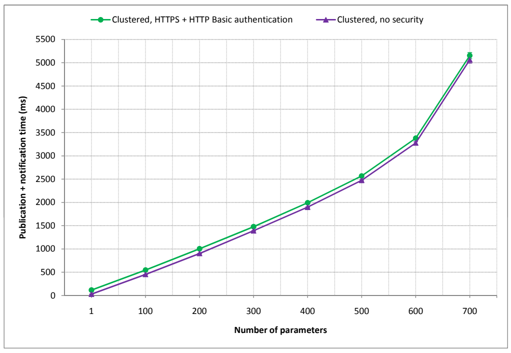
Experiments in this category were carried out to see how different number of parameters affects the overall performance of the implemented platform in both standalone and clustered versions. In Figure F.10, for example, the SpO2, the pulse rate, and the SOS information channels use 1 parameter each, the location information channel has 2 parameters, while the spirometry information channel has 13 parameters for each measurement.

Four experiments were conducted in this category, but unlike the previous category, the publication rate is fixed at one publication per second in order not to load the application servers. The number of parameters is varied instead, ranging from 1 to 700 parameters. Each measurement in all experiments lasted five minutes, generating 300 data. Each measurement was performed three times to ensure data consistency, and the first 1% of the captured data are removed from every measurement to avoid the start-up effect of the application servers in serving incoming requests. All measurement data is averaged, and plotted alongside their confidence intervals at 95% confidence level, just like in the previous category.

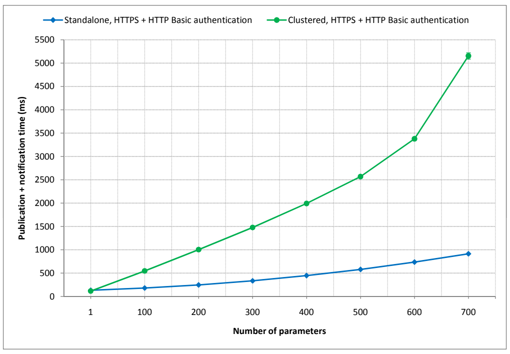
(a) Standalone secure vs. insecure



(b) Clustered secure vs. insecure



(c) Secure standalone vs. clustered



(d) Insecure standalone vs. clustered

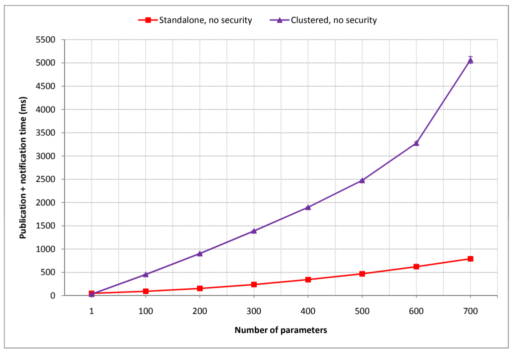


Figure F.19: Publication and notification time comparisons with number of parameters as variable

A comparison of total average publication and notification times between the secured and non-secured standalone IIP deployments is shown in Figure F.19(a). The time difference starts at about 85 ms when 1 parameter is used, and the gap increases almost linearly to around 125 ms when 700 parameters are used. This gap represents the implemented security mechanism's overhead in the standalone IIP set-up.

Figure F.19(b) depicts a comparison of publication and notification latency between the secured and non-secured clustered IIP set-ups. The time difference is stabled across all measurements at about 90 ms. From Figure F.19(a) and F.19(b) it can be seen that the additional overhead of the security mechanism does not change much with the increased number of parameters being used for both standalone and clustered deployments of the IIP.

A comparison of average latency difference between secured standalone and clustered IIP deployments is shown in Figure F.19(c). When only 1 parameter is used, the clustered set-up outperforms its standalone rival with around 20 ms difference. However, the clustered deployment suffers more latency overhead compared to the standalone version with the increasing number of parameters being used. With 100 parameters used, the clustered deployment performs worse than its standalone counterpart by around 365 ms, and it is further worsened as the number of parameters being used is increased.

Figure F.19(d) shows a comparison of average publication and notification times between non-secured standalone and clustered IIP set-ups. Almost similar to Figure F.19(c), the clustered version in this experiment wins in terms of latency difference compared to its standalone counterpart by about 20 ms when only 1 parameter is used, but it suffers when the number of parameters increases. When 100 parameters are used, the clustered set-up falls short around 360 ms in latency performance compared to the standalone deployment.

From the four experiments in this category (i.e. number of parameters as variable), it can be concluded that the clustered set-up of the IIP performs better than the standalone set-up when the number of parameters being used is low. The latency increase rate is higher for the clustered version compared to the standalone deployment as the number of parameters being used grows. This can be seen as a direct impact of the synchronisation process between data nodes in the data layer since synchronising large amounts of incoming new data takes much more time than writing directly to one data store. The use of the security mechanism adds overhead to both standalone and clustered set-ups, but the processing times are not effected significantly with the increasing number of parameters being used. The confidence intervals are very small compared to the mean values in all experiments.

VI. Conclusions and Future Work

An information integration platform (i.e. the IIP) has been designed and developed to bridge communications between everyday objects and Internet-based services, breaking the traditional vertical "silo" approach of integration. This broker platform follows an event-driven SOA paradigm with a publish/subscribe messaging pattern and exposes its functionalities through a set of RESTful Web services. An identity-based access control is used and has been implemented in the prototype to ensure information privacy between service clients (i.e. information providers at the everyday objects' side and information consumers at the Internet-based services' side). Only the owners of information channels in the IIP can publish new information to their information channels (i.e. write access), and only information consumers that are listed in an information channel's allowed user list can subscribe to that particular information channel for notifications (i.e. read access). Information consumers can request access to different information channels, and information providers have full rights to add or remove information consumers from/to the allowed user lists of information channels they own. Three services within the healthcare domain have been developed, namely remote health monitoring service, SOS-SMS service, and SOS-social media service. This shows how the platform can be utilised to enhance quality of life by means of novel personalised services for patients in particular, and to society in general. To avoid a single point of failure, a 3-layer deployment architecture of the IIP has been implemented, supporting high availability and scalability by employing redundancy of service components as well as clustering technology with load balancer. The IIP prototype has been tested to work as intended, and some experiments have been conducted to compare the average total publication and notification times between the standalone IIP deployment and the clustered version, as well as between the inclusion of the security scheme and without it. From the experiments with the current prototypes, it can be concluded that the clustered deployment of the IIP can handle better higher publication rates compared to its standalone counterpart when the number of parameters being used is low. The standalone set-up outperforms the clustered version when the number of parameters increases. In both cases, the incorporation of the security mechanism adds latency overhead.

The HTTP protocol is used by the IIP for message exchanges with both information providers and consumers due to its pervasive usage on the Web. Newer and lighter protocols that are specifically designed for embedded devices such as the CoAP and the MQ Telemetry Transport (MQTT) are planned to be supported in the next version of the IIP, especially for interfacing with information providers. JavaScript Object Notation (JSON) will also be supported in the next implementation iteration as an alternative data format to the currently used XML, so that information consumers can choose which data format they prefer for the notifications.

The developed healthcare services described in this article are rather simplistic and straightforward, utilising only a handful of devices as data sources. On the other hand, the IIP is designed to mediate a wide spectrum of information from a variety of information providers, supporting different application areas. More sophisticated context-aware services that combine information from various different devices, such as home appliances in a smart home environment to assist the patients in living independently in their homes, are planned to be developed in the near future.

Optimisations in all three layers of the proposed architecture for deployment are planned to be conducted in the continuation of this work, and further security and privacy enhancements will be investigated and incorporated in the next prototyping round. Additionally, the current IIP prototype is planned to be used in several pilot projects that include real-life patients within the healthcare domain in collaboration with several hospitals and partner companies. In turn, they will provide feedback on how the system could further be improved.

ACKNOWLEDGEMENTS

This work is primarily sponsored by the University of Agder under PhD research fellowship project grant number 63730.

REFERENCES

[1] K. Ashton, "That Internet of Things Thing," *RFiD Journal*, vol. 22, pp. 97-114, June 2009.

[2] S. Sarma, D. L. Brock, and K. Ashton, "The Networked Physical World," Auto-ID Center White Paper MIT-AUTOID-WH-001, 2000.

[3] D. Lake, A. Rayes, and M. Morrow, "The Internet of Things," *The Internet Protocol Journal*, vol. 15, no. 3, pp. 10-19, September 2012.

[4] F. Mattern and C. Floerkemeier, "From the Internet of Computers to the Internet of Things," *Lecture Notes in Computer Science*, vol. 6462, pp. 242-259, 2010.

[5] N. Gershenfeld, R. Krikorian, and D. Cohen, "The Internet of things," *Scientific American*, pp. 76-81, October 2004.

[6] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A View of Cloud Computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50-58, April 2010.

[7] W. R. Stevens and G. R. Wright, “TCP/IP Illustrated: Vol. 2: The Implementation,” Addison-Wesley Professional, 1995.

[8] D. Trossen and D. Pavel, “Building a Ubiquitous Platform for Remote Sensing Using Smartphones,” in *Proc. of 2nd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services,* San Diego, CA, USA, July 2005, pp. 485-489.

[9] A. J. Jara, S. Varakliotis, A. F. Skarmeta, P. Kirstein, “Extending the Internet of Things to the Future Internet through IPv6 support,” *Mobile Information Systems*, vol. 10, no. 1, pp. 3-17, 2014.

[10] A. J. Jara, P. Moreno-Sanchez, A. F. Skarmeta, S. Varakliotis, P. Kirstein, “IPv6 Addressing Proxy: Mapping Native Addressing from Legacy Technologies and Devices to the Internet of Things (IPv6),” *Sensors*, vol. 13, no. 5, pp. 6687-6712, May 2013.

[11] G. Mulligan, “The 6LoWPAN Architecture,” in *Proc. of the 4th Workshop on Embedded Networked Sensors,* Cork, Ireland, June 2007, pp. 78-82.

[12] A. Ludovici, A. Calveras, J. Casademont, “Forwarding Techniques for IP Fragmented Packets in a Real 6LoWPAN Network,” *Sensors*, vol. 11, no. 1, pp. 992-1008, January 2011.

[13] A. J. Jara, M. A. Zamora, A. Skarmeta, “Glowbal IP: An adaptive and transparent IPv6 integration in the Internet of Things,” *Mobile Information Systems*, vol. 8, no. 3, pp. 177-197, 2012.

[14] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, “Hypertext Transfer Protocol – HTTP/1.1,” RFC 2616, June 1999.

[15] B. Krishnamurthy and J. Rexford, “Web Protocols and Practice: HTTP/1.1, Networking Protocols, Caching, and Traffic Measurement,” Addison-Wesley, 2001.

[16] T. Berners-Lee, R. Cailliau, A. Luotonen, H. F. Nielsen, and A. Secret, “The World-Wide Web,” *Communications of the ACM*, vol. 37, no. 8, pp. 76-82, August 1994.

[17] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, and D. F. Ferguson, “Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging and More,” Prentice Hall PTR, 2005.

[18] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," Ph.D. dissertation, University of California, 2000.

[19] A. J. Jara, M. A. Zamora-Izquierdo, A. F. Skarmeta, "Interconnection Framework for mHealth and Remote Monitoring Based on the Internet of Things," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 47-65, September 2013.

[20] A. J. Jara, M. A. Zamora, A. F. Skarmeta, "An internet of thingsbased personal device for diabetes therapy management in ambient assisted living (AAL)," *Personal and Ubiquitous Computing*, vol. 15, no. 4, pp. 431-440, April 2011.

[21] M. Brenner and M. Unmehopa, "The Silo Syndrome and its Solution," *The Open Mobile Alliance: Delivering Service Enablers for Next-Generation Applications*, pp. 7-20, April 2008.

[22] J. Pansiot, D. Stoyanov, D. McIlwraith, B. Lo, and G. Yang, "Ambient and Wearable Sensor Fusion for Activity Recognition in Healthcare Monitoring Systems," in *Proc. of 4th International Workshop on Wearable and Implantable Body Sensor Networks (BSN 2007),* Aachen, Germany, March 2007, pp. 208-212.

[23] W. Y. Chung, S. Bhardwaj, A. Purwar, D. S. Lee, and R. Myllylae, "A Fusion Health Monitoring Using ECG and Accelerometer sensors for Elderly Persons at Home," in *Proc. of 29th IEEE EMBS Annual International Conference (EMBC),* Lyon, France, August 2007, pp. 3818-3821.

[24] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 84-106, January 2013.

[25] D. Huang, "Mobile Cloud Computing," *IEEE COMSOC Multimedia Communications Technical Committee (MMTC) E-Letter*, vol. 6, no. 10, pp. 27-31, October 2011.

[26] K. Kumar and Y. H. Lu, "Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?," *Computer*, vol. 43, no. 4, pp. 51-56, April 2010.

[27] P. T. Eugster, P. A. Felber, R. Guerraoui, and A. M. Kermarrec, "The Many Faces of Publish/Subscribe," *ACM Computing Surveys (CSUR)*, vol. 35, no. 2, pp. 114-131, June 2003.

[28] R. Strom, G. Banavar, T. Chandra, M. Kaplan, K. Miller, B. Mukherjee, D. Sturman, and M. Ward, "Gryphon: An Information Flow Based Approach to Message Brokering," in *Proc. of International Symposium on Software Reliability Engineering,* 1998.

[29] P. R. Pietzuch and J. M. Bacon, "Hermes: A Distributed Event-Based Middleware Architecture," in *Proc. of 22nd International Conference on Distributed Computing Systems Workshops (ICDCSW),* Vienna, Austria, July 2002, pp. 611-618.

[30] G. Cugola, E. Di Nitto, and A. Fuggetta, "The JEDI Event-Based Infrastructure and Its Application to the Development of the OPSS WFMS," *IEEE Transactions on Software Engineering*, vol. 27, no. 9, pp. 827-850, September 2001.

[31] A. Rowstron, A.-M. Kermarrec, M. Castro, and P. Druschel, "SCRIBE: The Design of a Large-Scale Event Notification Infrastructure," *Lecture Notes in Computer Science*, vol. 2233, pp. 30-43, 2001.

[32] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf, "Achieving Scalability and Expressiveness in an Internet-Scale Event Notification Service," in *Proc. of 19th Annual ACM Symposium on Principles of Distributed Computing (PODC),* Portland, OR, USA, July 2000, pp. 219-227.

[33] Y. Huang and D. Gannon, "A Comparative Study of Web Services-based Event Notification Specifications," in *Proc. of International Conference on Parallel Processing Workshops (ICPPW),* Columbus, OH, USA, August 2006, pp. 7-14.

[34] Y. Huang, A. Slominski, C. Herath, and D. Gannon, "WS-Messenger: A Web Services-based Messaging System for Service-Oriented Grid Computing," in *Proc. of 6th IEEE International Symposium on Cluster Computing and the Grid (CCGRID),* Singapore, May 2006, pp. 1-8.

[35] C. Fu, F. Belqasmi, and R. Glitho, "RESTful Web Services for Bridging Presence Service across Technologies and Domains: An Early Feasibility Prototype," *IEEE Communications Magazine*, vol. 48, no. 12, pp. 92-100, December 2010.

[36] A. Kamilaris, V. Trifa, and D. Guinard, "Building Web-based Infrastructures for Smart Meters," in *Proc. of Workshop on Energy Awareness and Conservation through Pervasive Applications,* Helsinki, Finland, May 2010, pp. 1-6.

[37] C. Heyer, "The Å Publish/Subscribe Framework," *Lecture Notes in Computer Science*, vol. 5585, pp. 99-110, 2009.

[38] B. Fitzpatrick, B. Slatkin, and M. Atkins, "PubSubHubbub Core 0.3 – Working Draft," [Online]. Available: `http://pubsubhubbub.googlecode.com/git/pubsubhubbub-core-0.3.html` [Accessed 12th November 2013].

[39] J. Gregorio and B. de Hora, "The Atom Publishing Protocol," RFC 5023, October 2007.

[40] D. Winer, "RSS 2.0 Specification," Berkman Center for Internet & Society at Harvard Law School, July 2003.

[41] Z. Shelby, K. Hartke, C. Bormann, "Constrained Application Protocol (CoAP)," Constrained Resources (CoRE) Working Group, Internet Engineering Task Force (IETF), draft-ietf-core-coap-18, June 2013.

[42] S. Li, J. Hoebeke, F. Van den Abeele, A. Jara, "Conditional observe in CoAP," Constrained Resources (CoRE) Working Group, Internet Engineering Task Force (IETF), draft-li-core-conditional-observe-04, June 2013.

[43] D. Barry, "Web Services and Service-Oriented Architecture: The Savvy Manager's Guide," Morgan Kaufmann Pub, 2003.

[44] B. M. Michelson, "Event-Driven Architecture Overview," Patricia Seybold Group, February 2006.

[45] D. Chappell, "Enterprise Service Bus," O'Reilly Media, Inc., 2004.

[46] Y. B. D. Trinugroho, M. Gerdes, M. M. M. Amjad, F. Reichert, and R. Fensli, "A REST-Based Publish/Subscribe Platform to Support Things-to-Services Communications," in *Proc. of 19th Asia-Pacific Conference on Communications (APCC),* Bali, Indonesia, August 2013, pp. 327-332.

[47] B. Littlewood and L. Strigini, "Software Reliability and Dependability: a Roadmap," in *Proc. of the Conference on The Future of Software Engineering,* Limerick, Ireland, June 2000, pp. 175-188.

[48] A. Avižienis, J. C. Laprie, and B. Randell, "Fundamental Concepts of Dependability," Research Report No. 1145, LAAS-CNRS, 2001.

[49] W. Stallings and L. Brown, "Computer Security: Principles and Practice," Prentice-Hall, 2008.

[50] A. Nadalin, C. Kaler, R. Monzillo, and P. Hallam-Baker, "Web Services Security: SOAP Message Security 1.0 (WS-Security 2004)," OASIS Standard 200401, March 2004.