DIGITALCOMMONS
@WAYNESTATE

Wayne State University

Wayne State University Dissertations

1-1-2018

Sofie: Smart Operating System For Internet Of Everything

Jie Cao
Wayne State University, jasonc1177@gmail.com

Follow this and additional works at: https://digitalcommons.wayne.edu/oa_dissertations

 Part of the Computer Sciences Commons

Recommended Citation

Cao, Jie, "Sofie: Smart Operating System For Internet Of Everything" (2018). *Wayne State University Dissertations*. 2092.
https://digitalcommons.wayne.edu/oa_dissertations/2092

This Open Access Dissertation is brought to you for free and open access by DigitalCommons@WayneState. It has been accepted for inclusion in Wayne State University Dissertations by an authorized administrator of DigitalCommons@WayneState.

www.manaraa.com

SOFIE: SMART OPERATING SYSTEM FOR INTERNET OF EVERYTHING

by

JIE CAO

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2018

MAJOR: COMPUTER SCIENCE

Approved By:

_____

Advisor Date

_____

_____

_____

_____

www.manaraa.com

DEDICATION

To my beloved family.

www.manaraa.com

ACKNOWLEDGEMENTS

I wish to express my sincere appreciation to those who supported and encouraged me in one way or another during the last six years.

First of all, I would like to give the deepest gratitude to my advisor, Dr. Weisong Shi, who has provided me with valuable guidance and endless patience. Dr. Shi taught me how to find interesting research problems and how to come out solutions step by step. His guidance encouraged me throughout the tough time in the Ph.D pursuit, as well as writing of this dissertation. He is not only a knowledgeable professor in academia, but also a kind and wise elder who impacts my career a lot. I cannot succeed without his support and guidance.

I also want to extend my thanks to Dr. Nathan Fisher, Dr. Dongxiao Zhu, and Dr. Paul Kilgore for serving as my committee members. Their professional suggestions on the prospectus are very valuable for me to improve my work. They also showed me how to be an excellent researcher and educator.

My sincere thanks also goes for Dr. Chonggang Wang, Dale Seed, and Bob Flynn who provided me opportunities to join their teams as intern. Without their precious support, it would not be possible to conduct these research.

Moreover, I would like to thank former and present MIST and LAST group members that I have had the pleasure to work with or alongside of. With these brilliant people, we had many exciting, stimulating, and thoughtful discussions in the laboratory.

Last but not least, I deeply appreciate the support, encourage and love from my wife Qianqian Qi and my son Oliver. They have been a constant source of strength and always there for me through the good time and bad time.

www.manaraa.com

TABLE OF CONTENTS

www.manaraa.com

www.manaraa.com

www.manaraa.com

www.manaraa.com

LIST OF FIGURES

www.manaraa.com

www.manaraa.com

LIST OF TABLES

www.manaraa.com

Chapter 1: Introduction

1.1 Edge computing in Internet of Things

Figure 1.1: The proliferation of Cloud computing.

Cloud computing has tremendously changed the way we live, work, and study since its inception around 2005 [16], as shown in Figure 1.1. For example, Software as a Service (SaaS) instances, such as Google Apps, Twitter, Facebook and Flickr, have been widely used in our daily life. Moreover, scalable infrastructures as well as processing engines developed to support cloud service are also significantly influencing the way of running business, for instance, Google File System [50], MapReduce [35], Apache Hadoop [112], Apache Spark [133], and so on.

Internet of Things (IoT) was firstly introduced to the community in 1999 for supply chain management [18], and then the concept of "making a computer sense information without the aid of human intervention" was widely adapted to other fields such as healthcare, home, environment, and transports [116, 55]. Now with IoT, we will arrive in the post-Cloud era, where there will be a large quality of data generated by things that are immersed in our daily life, and a lot of applications will also be deployed at the edge to consume these data. By 2019, data produced by people, machines, and things will reach 500 zettabytes, as

www.manaraa.com

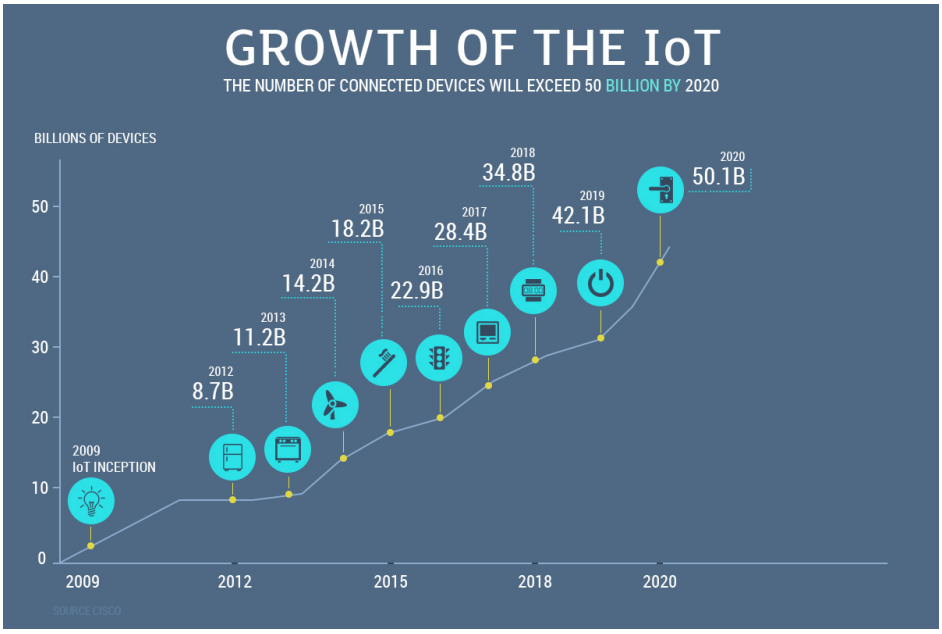

Figure 1.2: The proliferation of Internet of Things [12].

estimated by Cisco Global Cloud Index, however, the global data center IP traffic will only reach 10.4 zettabytes by that time [12]. By 2019, 45% of IoT-Created data will be stored, processed, analyzed, and acted upon close to, or at the Edge of, the network [13]. There will be 50 billion things connected to the Internet by 2020, as predicted by Cisco Internet Business Solutions Group [38] and shown in Figure 1.2.

Moreover, the end user in Cloud computing paradigm is also changing from data consumer to data producer. As we can see from Figure 1.3, more and more data in cloud service is generated by end users. This changing of role also requires data to be computed at the Edge of network in order to save bandwidth and cloud computing resource. Some IoT applications might require very short response time, some might involve private data, and some might produce a large quantity of data which could be a heavy load for networks. Cloud computing is not efficient enough to support these applications.

1.1 Two scenarios for Edge computing

Edge computing could be a good solution for IoT applications by allowing computation to be performed at the edge of the network, on downstream data on behalf of cloud

www.manaraa.com

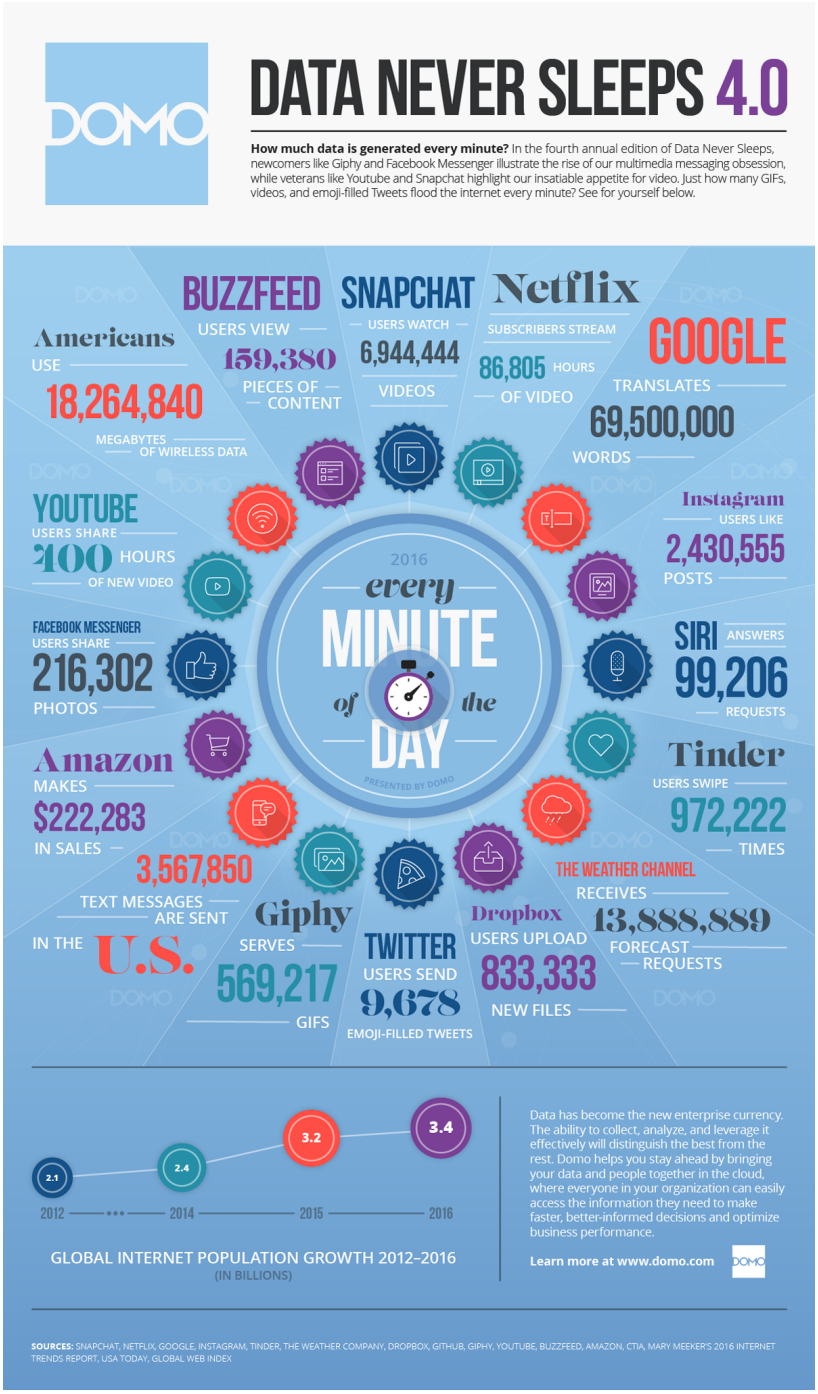

Figure 1.3: Data generated by end user [3].

services and upstream data on behalf of IoT services. Here we use Smart home and AMBER alert as two scenarios to further illustrate how Edge computing could benefit our daily life.

www.manaraa.com

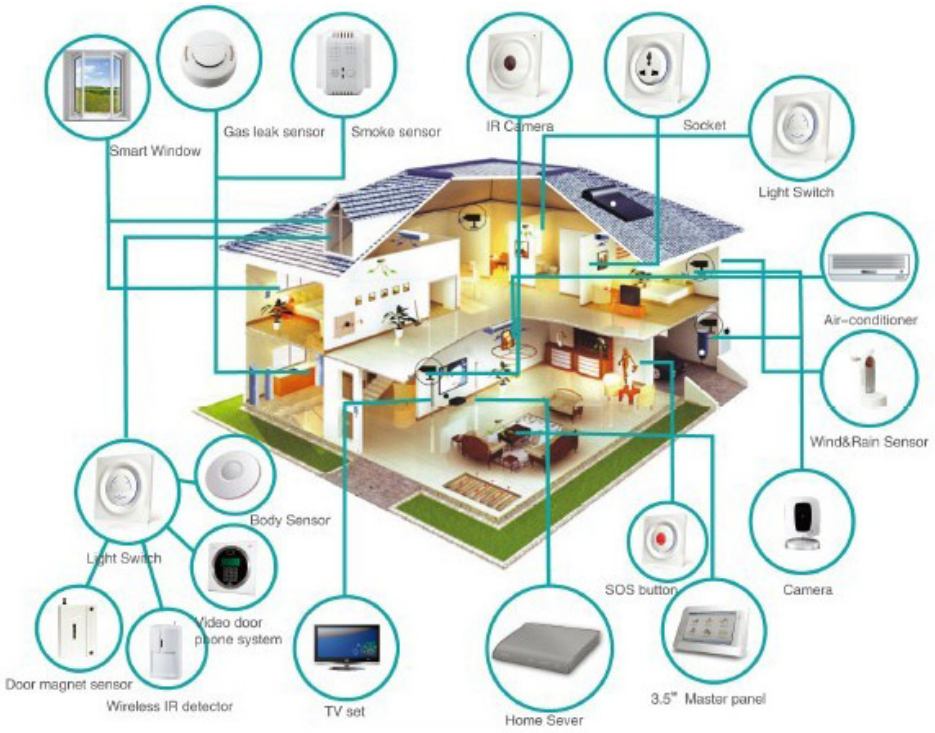

Figure 1.4: Smart home overview [7].

Smart home

First we take smart home as a scenario, as shown in Figure 1.4. With more and more connected things at home, people could implement a lot of smart services and automation jobs. IoT would benefit the home environment a lot. Some products have been developed and are available on the market such as smart light, smart TV and robot vacuum. However, just adding a Wi-Fi module to the current electrical device and connecting it to the cloud is not enough for a smart home. In a smart home environment, besides the connected device, cheap wireless sensors and controllers should be deployed to room, pipe, and even floor and wall. These things would report an impressive amount of data and for the consideration of data transportation pressure and privacy protection, this data should be mostly consumed in the home. This feature makes the Cloud computing paradigm unsuitable for a smart home. Nevertheless, Edge computing is considered perfect for building a smart home: with an edge gateway running a specialized EdgeOS (Edge Operating System) in the home, the

www.manaraa.com

things can be connected and managed easily in the home, the data can be processed locally to release the burdens for Internet bandwidth, and the service can also be deployed on the EdgeOS for better management and delivery. Researchers in this community have put some effort in building the smart home. Power management is smart home is discussed in [57, 60, 71, 94]. Assistant service in smart home for a more convenient life is proposed in [78, 88, 20]. Current smart home systems such as HomeOS from Microsoft, HomeKit

Figure 1.5: Smart home systems.

from Apple, and SmartThings from Samsung provide the connectivity of their own devices, as shown in Figure 1.5. However, there are some missing part in those smart home systems. First is that they only support their own communication protocol, which means you can only use their supported devices. Second is that these systems only provide connection between gateway and Edge devices, but they do not satisfy other performance requirements such as differentiation, extensibility and isolation. Details about these requirements will be discussed in the following Chapter.When applying Edge computing in smart home, practitioners should take several characteristics into consideration, including resource constrains in end devices and sensors, usage privacy and heterogeneous communication channels. These features of smart home bring about several new challenges that remains unsolved.

www.manaraa.com

- First is the missing of an operating system specialized for Edge computing system. Without an operating system, it is very hard to manage the device as well as the service in any Edge computing application.
- Secondly, device management is an issue, without an efficient naming mechanism of components in smart home, it is impossible to easily deploy new service or add/remove/replace any device at home.
- Thirdly, component selection is another challenge in Edge computing. When developing Edge computing applications, system designers and practitioners usually face several performance requirements such as the accuracy, battery life and system reliability. Given the hard requirements in system performance, how to choose an optimal combination from various sensors, algorithms and Edge computing systems to form the application is the most important problem that practitioners need to address.
- Last is privacy and security, such as identity and access management, network security, privacy, as well as secure data storage and computation at different layers.

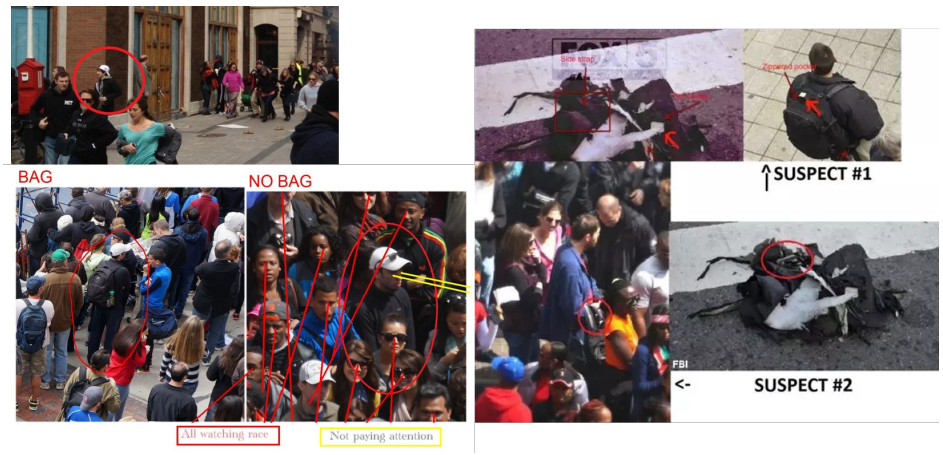

Figure 1.6: How social media helps in evidence collection [4].

AMBER Alert

Considering AMBER Alert, the child abduction alert system which originated in the US in 1996 as another scenario. Through commercial radio channel, cable TV, email, and

www.manaraa.com

SMS text message, the alert message could be distributed efficiently to each individual, however, individualâĂŹs contribution is very limited in these communication channels. Social media which leverage Cloud computing could help in this situation. Twitter and Facebook are proved to be very important policing tools in the Boston marathon bombing attack in 2013 since a lot of onsite photos were uploaded to the centralized cloud server via individualâĂŹs social media account, which were used for evidence collection, as shown in Figure 1.6. Now with Edge computing, it would be much easier to find the missing child. With Edge computing, the computing task of searching the missing child can be distributed to every Edge device, such as smartphone, security camera, smart vehicle, etc. Moreover, the Edge device can automatically perform video and image analysis locally and eventually the result will be collected via different layers of Edge and reach Cloud for decision making. In this scenario, two key questions are remaining unsolved.

- How can practitioners program for the computing task that can be deployed on various Edge devices including cloud server, PC, mobile phone, and smart camera with computing capability?

- How can the raw data and fractional computing result from various Edge device being collected and abstracted efficiently through multiple layers and eventually summarized as the final integral computing result?

1.1 SOFIE: Smart Operating System For Internet Of Everything

To realize the vision of Edge computing, we argue that the systems and network community need to work together. For the challenges mentioned in smart home and AMBER Alert scenarios, we further summarized them as *programmability*, *naming*, *data abstraction*, *service management*, *privacy and security* and *optimization metrics*. To address these challenges, we propose SOFIE: Smart Operating System For Internet Of Everything. SOFIE will manage and maintain the connections between gateway and Edge devices via multiple communication channels; SOFIE provides programmability for Edge application practitioners to

www.manaraa.com

design and implement their applications easily on SOFIE; SOFIE also supports a naming mechanism to support better service management and device management; data abstraction layer is introduced in SOFIE to isolate Edge device and Edge applications/services, which cloud help protect privacy and security of the data. We also argue that following four fundamental features should be supported in SOFIE to guarantee a reliable system, including Differentiation, Extensibility, Isolation, and Reliability (DEIR). Moreover, SOFIE also comes with a Wellness Management model to handle the health of the Edge devices. Data quality will be detected to guarantee the accuracy of the data; Lifetime of the Edge devices will be managed by the life-cycle management model; In order to make sure that the selection of the optimized component combination for each service can achieve the best performance, SURF framework is also included in wellness management model.

1.2 Summary of Contributions

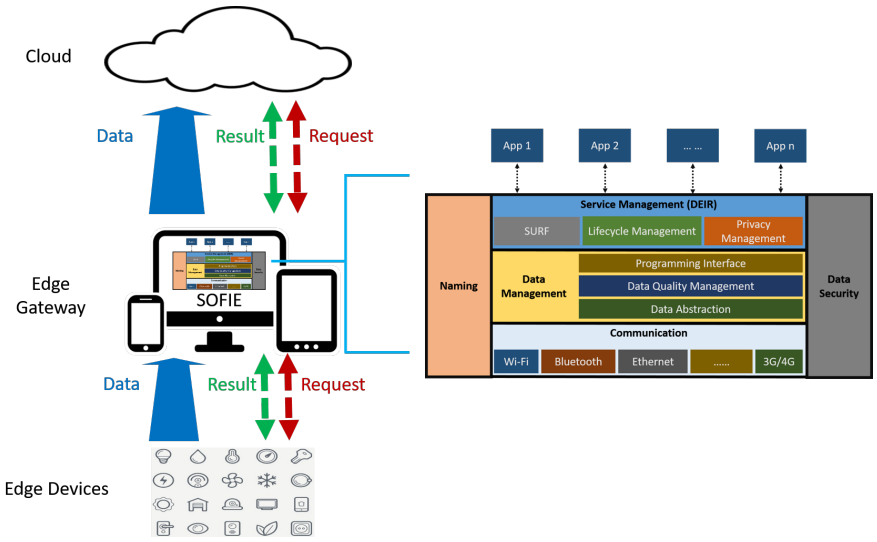

Figure 1.7: Contribution overview.

Figure 1.7 shows the overview structure of our work. On top of the computing paradigm is Cloud computing, and at the Edge of the paradigm are IoT devices, which also known as Edge devices. Edge will handle the data and computing between Cloud and Edge devices. Edge Gateway sitting in the middle will connecting Cloud and Edge devices; downstream data on behalf of cloud services; and upstream data on behalf of IoT services.

www.manaraa.com

SOFIE will run on the Edge gateway to handle the communication with Edge devices; abstract and per-process data; and manage the service running on the gateway. Moreover, SOFIE will also manage the wellness of the system, including data quality management to handle the accuracy of the data, life-cycle management to handle the battery-life and component failure on Edge devices, and SURF to choose the optimized combination of Edge devices for different services based on their performance requirements such as battery-life, data precision, data quantity, etc. The main contributions of our work are:

1. we came up with our understanding of Edge computing, with the rationale that computing should happen at the proximity of data sources. Then we list several cases whereby Edge computing could flourish from cloud offloading to a smart environment such as home and city. We also introduce Collaborative Edge, since Edge can connect end user and cloud both physically and logically so not only the conventional Cloud computing paradigm is still supported, but also it can connect long distance networks together for data sharing and collaboration because of the closeness of data. At last, we put forward the challenges and opportunities that are worth working on, including programmability, naming, data abstraction, service management, privacy and security, as well as optimization metrics.

2. We have designed SOFIE, a Smart Operating System For Internet of Everything. With the design of SOFIE, challenges of programmability, naming, data abstraction, service management, privacy and security, as well as optimization metrics could be addressed efficiently in Edge computing applications and systems.

3. To help the Edge computing practitioners to select the optimal component combinations that can meet the performance requirements and reduce cost as much as possible at the same time, we introduce SURF: a framework for component selection in Edge computing application development. SURF includes performance evaluation metrics and a methodology for using performance vector. We tested our methodology through

www.manaraa.com

a case study, and the result showed that our methodology is very efficient in finding the optimal component combination.

4. We implemented SOFIE on a mature IoT/M2M system, which is oneM2M SDP. We also supported semantic service for the system meanwhile provide access control as the security feature.

1.3 Outline

The rest of this document is organized as follows: Chapter 1.3 illustrate our vision and understanding of Edge computing. We also give several case studies to further explain how Edge computing could be adapted to the current computing paradigm. Then we summarize the challenges in detail and bring forward some potential solutions and opportunities worth further research, including programmability, naming, data abstraction, service management, privacy and security and optimization metrics; Chapter 2.4 introduce the design of SOFIE and further illustrate how SOFIE can help address the challenges in Chapter 1.3; In order to explain how SOFIE could be used to solve the component selection problem in Edge computing, in Chapter 3.8 we introduce SURF, which is a model in SOFIE aiming to address the component selection problem, we also illustrate how this model can be applied to real life applications through a case study, and discuss challenging issues and two interesting finds from our implementation; Chapter 4.6 introduces how we deploy SOFIE on an IoT/M2M system and support semantics with access control. Finally, this dissertation concludes and my future research agenda lists in Chapter 5.6 and Chapter 5.6.

www.manaraa.com

Chapter 2: Edge Computing

The proliferation of Internet of Things and the success of rich cloud services have pushed the horizon of a new computing paradigm, Edge computing, which calls for processing the data at the edge of the network. Edge computing has the potential to address the concerns of response time requirement, battery life constraint, bandwidth cost saving, as well as data safety and privacy. In this Chapter, we introduce the definition of Edge computing, followed by several case studies, ranging from cloud offloading to smart home and city, as well as collaborative Edge to materialize the concept of Edge computing. Finally, we present several challenges and opportunities in the field of Edge computing.

2.1 What is Edge computing

Data is increasingly produced at the edge of the network, therefore, it would be more efficient to also process the data at the edge of the network. Previous work such as micro DataCenter [52, 33], Cloudlet [102], and fog computing [22] has been introduced to the community because Cloud computing is not always efficient for data processing when the data is produced at the edge of the network. In this section, we list some reasons why Edge computing is more efficient than Cloud computing for some computing services, then we give our definition and understanding of Edge computing.

2.1 Why do we need Edge computing

Push from cloud services

Putting all the computing tasks on the cloud has been proved to be an efficient way for data processing since the computing power on the cloud outclasses the capability of the things at the edge. However, compared to the fast developing data processing speed, the bandwidth of the network has come to a standstill. With the growing quantity of data generated at the edge, speed of data transportation is becoming the bottleneck for the Cloud based computing paradigm. For example, about 5 Gigabyte data will be generated by a Boeing 787 every second [2], but the bandwidth between the airplane and either satellite or base

www.manaraa.com

station on the ground is not large enough for data transmission. Consider an autonomous vehicle as another example. 1 Gigabyte data will be generated by the car every second and it requires real-time processing for the vehicle to make correct decisions [6]. If all the data needs to be sent to the cloud for processing, the response time would be too long. Not to mention that current network bandwidth and reliability would be challenged for its capability of supporting a large number of vehicles in one area. In this case, the data needs to be processed at the edge for shorter response time, more efficient processing and smaller network pressure.

Pull from Internet of Things

Almost all kinds of electrical devices will become part of IoT, and they will play the role of data producers as well as consumers, such as air quality sensors, LED bars, streetlights and even an Internet-connected microwave oven. It is safe to infer that the number of things at the Edge of the network will develop to more than billions in a few years. Thus, raw data produced by them will be enormous, making conventional Cloud computing not efficient enough to handle all these data. This means most of the data produced by IoT will never be transmitted to the cloud, instead it will be consumed at the edge of the network.

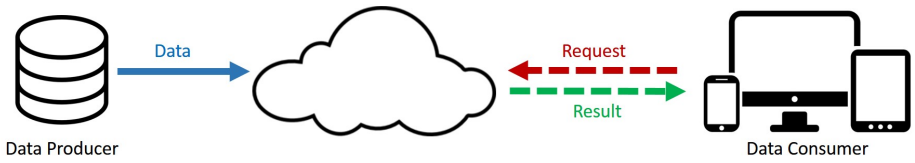

Figure 2.1: Cloud computing paradigm.

Figure 2.1 shows the conventional Cloud computing structure. Data producers generate raw data and transfer it to cloud, and data consumers send request for consuming data to cloud, as noted by the blue solid line. The red dotted line indicates the request for consuming data being sent from data consumers to cloud, and the result from cloud is represented by the green dotted line. However, this structure is not sufficient for IoT. Firstly, data quantity at the edge is too large, which will lead to huge unnecessary bandwidth and

www.manaraa.com

computing resource usage. Secondly, the privacy protection requirement will pose an obstacle for Cloud computing in IoT. Lastly, most of the end nodes in IoT are energy constrained things, and the wireless communication module is usually very energy hungry, so offloading some computing tasks to the edge could be more energy efficient.

Change from data consumer to producer

In the Cloud computing paradigm, the end devices at the edge usually play as data consumer, for example, watching a YouTube video on your smart phone. However, people are also producing data nowadays from their mobile devices. The change from data consumer to data producer/consumer requires more function placement at the edge. For example, it is very normal that people today take photos or do video recording then share the data through a cloud service such as YouTube, Facebook, Twitter or Instagram. Moreover, every single minute, YouTube users upload 72 hours of new video content; Facebook users share nearly 2.5 million pieces of content; Twitter users tweet nearly 300,000 times; Instagram users post nearly 220,000 new photos [3]. However, the image or video clip could be fairly large and it would occupy a lot of bandwidth for uploading. In this case, the video clip should be demised and adjusted to suitable resolution at the edge before uploading to cloud. Another example would be wearable health devices. Since the physical data collected by the things at the Edge of the network is usually private, processing the data at the edge could protect user privacy better than uploading raw data to cloud.

2.1 What is Edge computing

Edge computing refers to the enabling technologies allowing computation to be performed at the edge of the network, on downstream data on behalf of cloud services and upstream data on behalf of IoT services. Here we define “Edge" as any computing and network resources along the path between data sources and cloud data centers. For example, a smart phone is the edge between body things and cloud, a gateway in a smart home is the edge between home things and cloud, a Micro Data Center

www.manaraa.com

(MDC) and a Cloudlet [102] is the edge between a mobile device and cloud. The rationale of Edge computing is that ***computing should happen at the proximity of data sources***. From our point of view, Edge computing is interchangeable with Fog computing [14], but Edge computing focus more toward the Things side, while Fog computing focus more on the infrastructure side. We envision that Edge computing could have as big an impact on our society as has the Cloud computing.

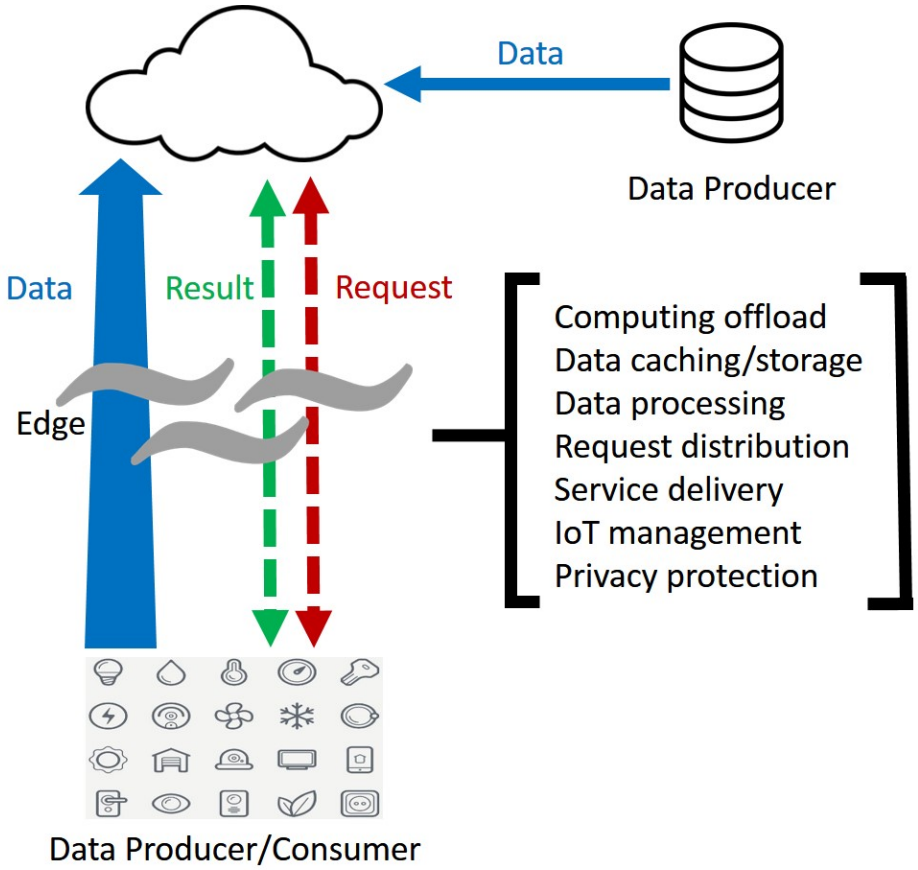

Figure 2.2: Edge computing paradigm.

Figure 4.3 illustrates the two-way computing streams in Edge computing. In the Edge computing paradigm, the things not only are data consumers, but also play as data producers. At the edge, the things can not only request service and content from the cloud, but also perform the computing tasks from the cloud. Edge can perform computing offloading, data storage, caching and processing, as well as distribute request and delivery service from cloud

www.manaraa.com

to user. With those jobs in the network, the edge itself needs to be well designed to meet the requirement efficiently in service such as reliability, security and privacy protection.

2.1 Edge computing benefits

In Edge Computing we want to put the computing at the proximity of data sources. This have several benefits compared to traditional Cloud based computing paradigm. Here we use several early results from the community to demonstrate the potential benefits. Researchers built a proof-of-concept platform to run face recognition application in [129], and the response time is reduced from 900ms to 169ms by moving computation from cloud to the Edge. In [58], the researchers use Cloudlets to offload computing tasks for wearable cognitive assistance, and the result shows that the improvement of response time is between 80ms to 200ms. Moreover, the energy consumption could also be reduced by 30-40% by cloudlet offloading. CloneCloud in [32] combine partitioning, migration with merging, and on-demand instantiation of partitioning between mobile and the cloud, and their prototype could reduce 20x running time and energy for tested applications.

2.2 Case study for Edge Computing

In this section, we give several case studies where Edge computing could shine to further illustrate our vision of Edge computing.

2.2 Cloud Offloading

In the Cloud computing paradigm, most of the computations happen in the cloud, which means data and requests are processed in the centralized cloud. However, such a computing paradigm may suffer longer latency (*e.g.,* long tail latency), which weakens the user experience. Numbers of researches have addressed the cloud offloading in terms of energy-performance tradeoff in a mobile-cloud environment [99, 63, 76, 74]. In Edge computing, the edge has certain computation resources, and this provides a chance to offload part of the workload from cloud.

www.manaraa.com

In the traditional content delivery network (CDN), only the data is cached at the edge servers. This is based on the fact that the content provider provides the data on the Internet, which is true for the past decades. In the IoT, the data is produced and consumed at the edge. Thus, in the Edge computing paradigm, not only data but also operations applied on the data should be cached at the edge.

One potential application that could benefit from Edge computing is online shopping services. A customer may manipulate the shopping cart frequently. By default, all these changes on his/her shopping cart will be done in the cloud, and then the new shopping cart view is updated on the customer's device. This process may take a long time depending on network speed and the load level of servers. It could be even longer for mobile devices due to the relatively low bandwidth of a mobile network. As shopping with mobile devices is becoming more and more popular, it is important to improve the user experience, especially latency related. In such a scenario, if the shopping cart updating is offloaded from cloud servers to edge nodes, the latency will be dramatically reduced. As we mentioned, the users' shopping cart data and related operations (*e.g.,* add an item, update an item, delete an item) both can be cached at the edge node. The new shopping cart view can be generated immediately upon the user request reaching the edge node. Of course, the data at the edge node should be synchronized with the cloud, however, this can be done in the background.

Another issue involves the collaboration of multiple edges when a user moves from one edge node to another. One simple solution is to cache the data to all edges the user may reach. Then the synchronization issue between edge nodes rises up. All these issues could become challenges for future investigation. At the bottom line, we can improve the interactive services quality by reducing the latency. Similar applications also include: i) Navigation applications can move the navigating or searching services to the edge for a local area, in which case only a few map blocks are involved; ii) content filtering/aggregating could be done at the edge nodes to reduce the data volume to be transferred; and iii) real-time applications such as vision-aid entertainment games, augmented reality, and connected

www.manaraa.com

health, could make fast responses by using edge nodes. Thus, by leveraging Edge computing, the latency and consequently the user experience for time-sensitive application could be improved significantly.

2.2 Video Analytics

The widespread of mobilephones and network cameras make video analytics an emerging technology. Cloud computing is no longer suitable for applications that requires video analytics due to the long data transmission latency and privacy concerns. Here we give an example of finding a lost child in the city. Nowadays, different kinds of cameras are widely deployed in the urban area and in each vehicle. When a child is missing, it is very possible that this child can be captured by a camera. However, the data from the camera will usually not be uploaded to the cloud because of privacy issues or traffic cost, which makes it extremely difficult to leverage the wide area camera data. Even if the data is accessible on the cloud, uploading and searching a huge quantity of data could take a long time, which is not tolerable for searching a missing child. With the Edge computing paradigm, the request of searching a child can be generated from the cloud and pushed to all the things in a target area. Each thing, for example a smart phone, can perform the request and search its local camera data and only report the result back to the cloud. In this paradigm, it is possible to leverage the data and computing power on every thing and get the result much faster compared with solitary Cloud computing.

2.2 Smart City

The Edge computing paradigm can be flexibly expanded from a single home to community, or even city scale. Edge computing claims that computing should happen as close as possible to the data source. With this design, a request could be generated from the top of the computing paradigm and be actually processed at the edge. Edge computing could be an ideal platform for smart city considering the following characteristics:

www.manaraa.com

large data quantity

A city populated by 1 million people will produce 180 PB data per day by 2019 [12], contributed by public safety, health, utility, and transports, etc. Building centralized cloud data centers to handle all of the data is unrealistic because the traffic workload would be too heavy. In this case, Edge computing could be an efficient solution by processing the data at the Edge of the network.

low latency

For applications that require predictable and low latency such as health emergency or public safety, Edge computing is also an appropriate paradigm since it could save the data transmission time as well as simplify the network structure. Decision and diagnosis could be made as well as distributed from the Edge of the network, which is more efficient compared with collecting information and making decision at central cloud.

location awareness

For geographic based applications such as transportation and utility management, Edge computing exceed cloud computing due to the location awareness. In Edge computing, data could be collected and processed based on geographic location without being transported to cloud.

2.2 Collaborative Edge

Cloud, arguably, has become the de facto computing platform for the big data processing by academia and industry. A key promise behind cloud computing is that the data should be already held or is being transmitted to the cloud and will eventually be processed in the cloud. In many cases, however, the data owned by stakeholders is rarely shared to each other due to privacy concerns and the formidable cost of data transportation. Thus, the chance of collaboration among multiple stake-holders is limited. Edge, as a physical small data center that connects cloud and end user with data processing capability, can also be part of the logical concept. *Collaborative Edge*, which connects the edges of multiple

www.manaraa.com

stakeholders that are geographically distributed despite their physical location and network structure is proposed [22]. Those ad hoc-like connected edges provide the opportunity for stakeholders to share and cooperate data.

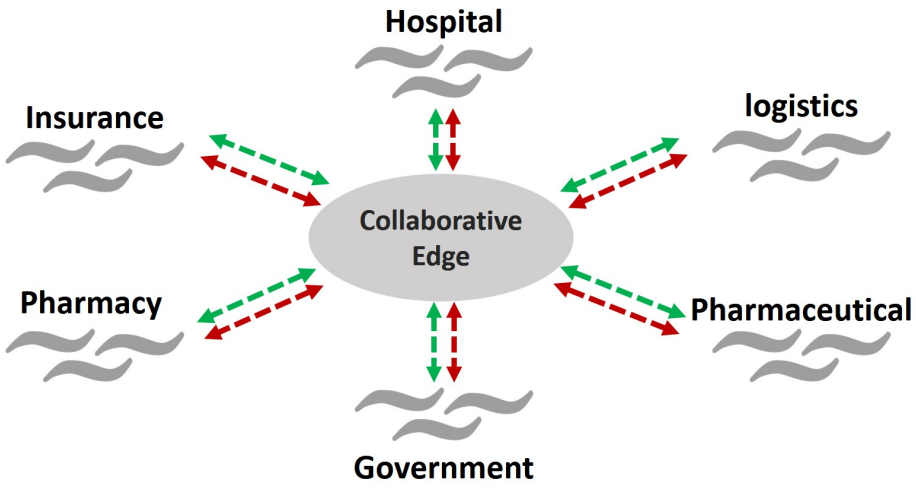

Figure 2.3: Collaborative Edge example: connected health.

One of the promising applications in the near future is connected health, as shown in Figure 2.3. The demand of geographically distributed data processing applications, *i.e.,* healthcare, requires data sharing and collaboration among enterprises in multiple domains. To attack this challenge, Collaborative Edge can fuse geographically distributed data by creating virtual shared data views. The virtual shared data is exposed to end users via a predefined service interface. An application will leverage this public interface to compose complex services for end users. These public services are provided by participants of Collaborative Edge, and the computation only occurs in the participant's data facility such that the data privacy and integrity can be ensured.

To show the potential benefits of Collaborative Edge, we use connected healthcare as a case study. We use a flu outbreak as the beginning of our case study. The patients flow to hospitals, and the Electronic Medical Record (EMR) of the patients will be updated. The hospital summarizes and shares the information for this flu outbreak, such as the average cost, the symptoms, and the population, etc. A patient theoretically will follow the prescription

www.manaraa.com

to get the pills from a pharmacy. One possibility is that a patient did not follow the therapy. Then the hospital has to take the responsibility for re-hospitalization since it cannot get the proof that the patient did not take the pills. Now, via Collaborative Edge, the pharmacy can provide the purchasing record of a patient to the hospital, which significantly facilitates healthcare accountability.

At the same time, the pharmacies retrieve the population of the flu outbreak using the Collaborative Edge services provided by hospitals. An apparent benefit is that the pharmacies have enough inventory to obtain much more profits. Behind the drug purchasing, the pharmacy can leverage data provided by pharmaceutical companies and retrieve the locations, prices and inventories of all drug warehouses. It also sends a transport price query request to the logistics companies. Then the pharmacy can make an order plan by solving the total cost optimization problem according to retrieved information. The pharmaceutical companies also receive a bunch of flu drug orders from pharmacies. At this point, a pharmaceutical company can reschedule the production plan and re-balance the inventories of the warehouses. Meanwhile, the Centers for Disease Control and Prevention (CDC), as our government representative in our case, is monitoring the flu population increasing at wide range areas, can consequently raise a flu alert to the people in the involved areas. Besides, further actions can be taken to prevent the spread of flu outbreak.

After the flu outbreak, the insurance companies have to pay the bill for the patients based on the policy. The insurance companies can analyze the proportion of people who has the flu during the outbreak. This proportion and the cost for flu treatment are significant factors to adjust the policy price for the next year. Furthermore, the insurance companies can also provide a personalized healthcare policy based on their EMR if the patient would like to share it.

Through this simple case, most of the participants can benefit from Collaborative Edge in terms of reducing operational cost and improving profitability. However, some of

www.manaraa.com

them, like hospitals in our case, could be a pure contributor to the healthcare community since they are the major information collector in this community.

2.3 Challenges in Edge Computing

We have described five potential applications of Edge computing in the related work. To realize the vision of Edge computing, we argue that the systems and network community need to work together. In this section, we will further summarize these challenges in detail and bring forward some potential solutions and opportunities worth further research, including *programmability*, *naming*, *data abstraction*, *service management*, *privacy and security* and *optimization metrics*.

2.3 Programmability

In Cloud computing, users program their code and deploy them on the cloud. The cloud provider is in charge to decide where the computing is conducted in a cloud. Users have zero or partial knowledge of how the application runs. This is one of the benefits of Cloud computing that the infrastructure is transparent to the user. Usually, the program is written in one programing language and compiled for a certain target platform, since the program only runs in the cloud. However, in the Edge computing, computation is offloaded from the cloud, and the edge nodes are most likely heterogeneous platforms. In this case, the runtime of these nodes differ from each other, and the programmer faces huge difficulties to write an application that may be deployed in the Edge computing paradigm.

2.3 Naming

In Edge computing, one important assumption is that the number of things is tremendously large. Atop the edge nodes, there are a lot of applications running, and each application has its own structure about how the service is provided. Similar to all computer systems, the naming scheme in Edge computing is very important for programing, addressing, things identification, and data communication. However, an efficient naming mechanism for the Edge computing paradigm has not been built and standardized yet. Edge practi-

www.manaraa.com

tioners usually needs to learn various communication and network protocols in order to communicate with the heterogeneous things in their system. The naming scheme for Edge computing needs to handle the mobility of things, highly dynamic network topology, privacy and security protection, as well as the scalability targeting the tremendously large amount of unreliable things.

Traditional naming mechanisms such as DNS and URI (Uniform Resource Identifier) satisfy most of the current networks very well. However, they are not flexible enough to serve the dynamic Edge network since sometimes most of the things at Edge could be highly mobile and resource constrained. Moreover, for some resource constrained things at the Edge of the network, IP based naming scheme could be too heavy to support considering its complexity and overhead.

New naming mechanisms such as Named Data Networking (NDN) [135] and Mobility-First [95] could also be applied to Edge computing. NDN provide a hierarchically structured name for content/data centric network, and it is human friendly for service management and provides good scalability for Edge. However, it would need extra proxy in order to fit into other communication protocols such as BlueTooth or Zigbee, and so on. Another issue associated with NDN is security, since it is very hard to isolate things hardware information with service providers. MobileFirst can separate name from network address in order to provide better mobility support, and it would be very efficient if applied to Edge services where things are of highly mobility. Neverless, a global unique identification (GUID) needs to be used for naming is MobileFirst, and this is not required in related fixed information aggregation service at the Edge of the network such as home environment. Another disadvantage of MobileFirst for Edge is the difficulty in service management since GUID is not human friendly.

2.3 Data Abstraction

Various applications can run on the EdgeOS consuming data or providing service by communicating through the APIs from the service management layer. Data abstraction

www.manaraa.com

has been well discussed and researched in the wireless sensor network and Cloud computing paradigm. However, in Edge computing, this issue becomes more challenging. With IoT, there would be a huge number of data generators in the network, and here we take a smart home environment as an example. In a smart home, almost all of the things will report data to the EdgeOS, not to mention the large number of things deployed all around the home. However, most of the things at the Edge of the network, only periodically report sensed data to the gateway. For example, the thermometer could report the temperature every minute, but this data will most likely only be consumed by the real user several times a day. Another example could be a security camera in the home which might keep recording and sending the video to the gateway, but the data will just be stored in the database for a certain time with nobody actually consuming it, and then be flushed by the latest video.

2.3 Service Management

In terms of service management at the Edge of the network, we argue that following four fundamental features should be supported to guarantee a reliable system, including Differentiation, Extensibility, Isolation, and Reliability (DEIR).

Differentiation: With the fast growth of IoT deployment, we expected multiple services will be deployed at the Edge of the network, such as Smart Home. These services will have different priorities. For example, critical services such as things diagnosis and failure alarm should be processed earlier than ordinary service. Health related service, for example, fall detection or heart failure detection should also have a higher priority compared with other service such as entertainment.

Extensibility: Extensibility could be a huge challenge at the Edge of the network, unlike a mobile system, the things in the IoT could be very dynamic. When the owner purchases a new thing, can it be easily added to the current service without any problem? Or when one thing is replaced due to wearing out, can the previous service adopt a new node easily? These problems should be solved with a flexible and extensible design of service management layer in the EdgeOS.

www.manaraa.com

Isolation: Isolation would be another issue at the Edge of the network. In mobile OS, if an application fails or crashes, the whole system will usually crash and reboot. Or in a distributed system the shared resource could be managed with different synchronization mechanisms such as a lock or token ring. However, in a smart EdgeOS, this issue might be more complicated. There could be several applications that share the same data resource, for example, the control of light. If one application failed or was not responding, a user should still be able to control their lights, without crashing the whole EdgeOS. Or when a user removes the only application that controls lights from the system, the lights should still be alive rather than experiencing a lost connection to the EdgeOS. This challenge could be potentially solved by introducing a deployment/undeployment framework. If the conflict could be detected by the OS before an application is installed, then a user can be warned and avoid the potential access issue. Another side of the isolation challenge is how to isolate a user's private data from third party applications. For example, your activity tracking application should not be able to access your electricity usage data. To solve this challenge, a well-designed control access mechanism should be added to the service management layer in the EdgeOS.

Reliability: Last but not least, reliability is also a key challenge at the Edge of the network. We identify the challenges in reliability from the different views of service, system, and data here.

- From the service point of view, it is sometimes very hard to identify the reason for a service failure accurately at field. For example, if an air conditioner is not working, a potential reason could be that a power cord is cut, compressor failure, or even a temperature controller has run out of battery. A sensor node could have lost connection very easily to the system due to battery outage, bad connection condition, component wear out, etc. At the Edge of the network, it is not enough to just maintain a current service when some nodes lose connection, but to provide the action after node failure makes more sense to the user. For example, it would be very nice if the EdgeOS could

www.manaraa.com

inform the user which component in the service is not responding, or even alert the user ahead if some parts in the system have a high risk of failure. Potential solutions for this challenge could be adapted from a wireless sensor network, or industrial network such as PROFINET [40].

- From the system point of view, it is very important for the EdgeOS to maintain the network topology of the whole system, and each component in the system is able to send status/diagnosis information to the EdgeOS. With this feature, services such as failure detection, thing replacement and data quality detection could be easily deployed at the system level.

- From the data point of view, reliability challenge rise mostly from the data sensing and communication part. As previously researched and discussed, things at the Edge of the network could fail due to various reasons and they could also report low fidelity data under unreliable condition such as low battery level [28]. Also various new communication protocols for IoT data collection are also proposed. These protocols serves well for the support of huge number of sensor nodes and the highly dynamic network condition [34]. However, the connection reliability is not as good as BlueTooth or WiFi. If both sensing data and communication is not reliable, how can the system still provide reliable service by leveraging multiple reference data source and historical data record is still an open challenge.

2.3 Privacy and Security

To illustrate the privacy and security of Edge computing more clearly, we will use smart home as a scenario in this section.

At the Edge of the network, usage privacy and data security protection are the most important services that should be provided. If a home is deployed with IoT, a lot of privacy information can be learned from the sensed usage data. For example, with the reading of the electricity or water usage, one can easily speculate if the house is vacant or not. In this

www.manaraa.com

case, how to support service without harming privacy is a challenge. Some of the private information could be removed from data before processing such as masking all the faces in the video. We think that keeping the computing at the edge of data resource, which means in the home, could be a decent method to protect privacy and data security. To protect the data security and usage privacy at the Edge of the network, several challenges remain open.

First is the awareness of privacy and security to the community. We take WiFi networks security as an example. Among the 439 million households who use wireless connections, 49% of WiFi networks are unsecured, and 80% of households still have their routers set on default passwords. For public WiFi hotspots, 89% of them are unsecured [10]. All the stake holders including service provider, system and application developer and end user need to aware that the users' privacy would be harmed without notice at the Edge of the network. For example, ip camera, health monitor, or even some WiFi enabled toys could easily be connected by others if not protected properly.

Second is the ownership of the data collected from things at Edge. Just as what happened with mobile applications, the data of end user collected by things will be stored and analyzed at the service provider side. However, leave the data at the Edge where it is collected and let the user fully own the data will be a better solution for privacy protection. Similar to the health record data, end user data collected at the Edge of the network should be stored at the Edge and the user should be able to control if the data should be used by service providers. During the process of authorization, highly private data could also be removed by the things to further protect user privacy.

Third is the missing of efficient tools to protect data privacy and security at the Edge of the network. Some of the things are highly resource constrained so the current methods for security protection might not be able to be deployed on thing because they are resource hungry. Moreover, the highly dynamic environment at the Edge of the network also makes the network become vulnerable or unprotected. For privacy protection, some platform such

www.manaraa.com

as Open mHealth is proposed to standardize and store health data [5], but more tools are still missing to handle diverse data attributes for Edge Computing.

The proliferation of Internet of Things is pushing the development of smart home. Composed of dependent devices or sensors which depend on edge nodes to compute and transmit data, independent gateway (edge) which are powerful enough do computation, and connected smart phone applications, smart home has become a typical scenario both in Internet of Things and Edge Computing. As shown in Figure 2.4, physical devices and sensors located at the periphery of the network will report data to edge (including routers, laptop, desktop computer, etc.), and most of them only do it periodically since not all data will be utilized. Some physical devices such as light bulbs, oven are not necessary to interact with the outside network while other devices like IP camera and smart TV may often expose themselves to the Cloud. On one hand, communicating with the Cloud could significantly increase the probability of attack. On the other hand, as mentioned in section II, the independent devices only interacting with edge nodes also take the risk of being attacked because of their limited capability. Due to unique characteristics of smart home, security and privacy are the most important services to be introduced and deployed in smart home operating system design. In this section, we will discuss new issues raised by smart home that related with security and privacy.

Why new issues in smart home

In a smart home, all household devices have the ability to connect to network, including doors, cameras, oven, thermostat, etc. Located at the Edge of network, these huge amount of IoT devices can generate a significant amount of data, while not all data are useful to users. In this situation, data should be preprocessed and then wait for further utilization. Periphery physical devices communicate with Edge operating system through multiple radio protocols such as Wi-Fi, Bluetooth, ZigBee or a cellular network. These different types of data, including device status and device info, then be gathered and processed by data ab-

www.manaraa.com

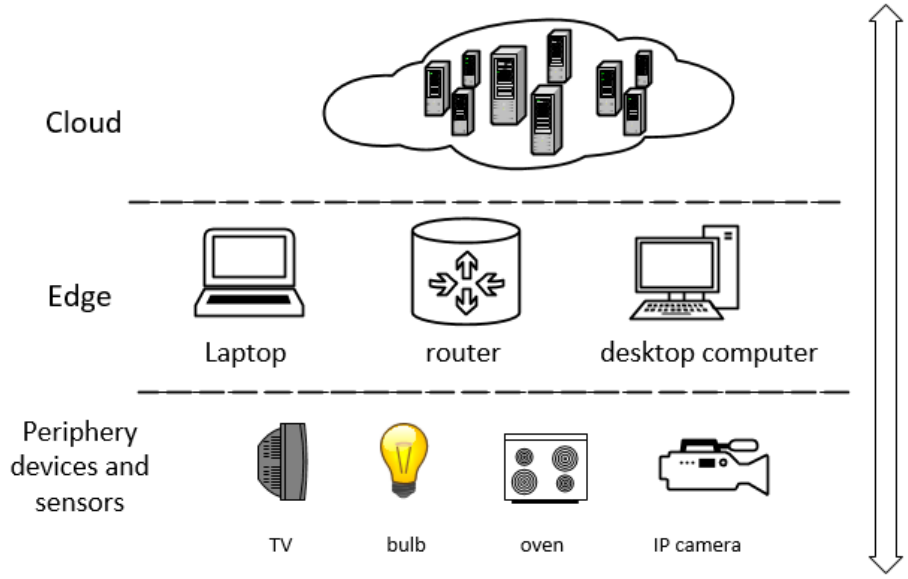

Figure 2.4: Edge computing in smart home.

straction layer. Service Management layer provides APIs for applications running on top of it to consume data or provide service.

Limited resources and computational capabilities, as well as multiple communication protocols makes it very difficult to protect smart home from attackers. What's more, security and privacy problems in smart home can always raise people's concerns. The reasons are mainly on the following several unique characteristics in smart home:

Resource Constrain. Devices in IoT always have the properties of small memory space, limited battery life, and low computational capability. Smart home has no exception but to take these factors into consideration. Constrained resource makes it impossible for IoT devices to employ some traditional security solutions, which have high requirements on computational ability or memory space.

Usage Privacy. Privacy issues can always draw people's attention, especially in a home environment. User data are keeping collected and analyzed by smart home system, then transmitted to provider's server and store there. It is hard to tell whether there are malicious applications which steal data for their own use. Besides, even if users have a strong sense of privacy, it is hard for them to tell which kind of smart home data should have the

www.manaraa.com

highest or lowest privacy. For example, some users do not care about data on their utility reports and set it as low privacy, while burglars can deduce family demographic composition and living standards from utility data, even what time the house could be empty.

Heterogeneous Communication. IoT devices in smart home are versatile both in function and capability. For example, home security surveillance system runs complicated algorithm (e.g., face recognition, voice detection) and report potential dangerous to users in real-time. In contrast, automatically controlled lights only perform simple computations like when to turn on or off the light. Different requirements on capability and complexity of IoT devices lead to various communicate methods when the devices exchange information with the smart home operating system, which also leads to different security levels in the same OS.

Although smart home system has experienced rapid development in recent decades, and early complicated device deployment systems had been improved into recently more friendly ones [41], security and privacy issues have not been taken great care of [114]. Several researchers and news reports illustrate that smart home system is vulnerable and easy to disclose users' privacy when facing attacks. For example, Forbes.com reported that hackers broke into a baby monitor and yelled at the baby [1]. Seizures in epileptic users by causing compact florescent lights to rapidly power cycle is introduced in [90]. Samsung's SmartThings framework has several design flaws that make door lock and alarm system exposed to attackers is found in [41]. CNN Money also reported that smart home devices were exposed security flaws to hackers [11]. These attacks could produce a very bad effect and severe losses to users.

It is easy to find out that cases are more complex in smart home when comparing with traditional smart phone or personal computer. Normally, if a malicious program infects a laptop, user can run a malware detector and reboot the machine. But if same thing happens in a smart home where various kinds of devices dividing labor and cooperating with each other, it is impossible to just simply turn everything off or to locate the problem

www.manaraa.com

among devices. Various reasons are responsible for this problem. On one hand, many device producers may have little expertise in security, unintentionally using the operating system and protocols only half patched or even malicious tampered. On the other hand, users may not be able to tell high priority data from tremendous amount of data produced by all devices he is using. It makes the whole system vulnerable for attackers to make use of seems meaningless data to achieve his evil goal. As mentioned above, consequences of insecure smart home are much more severe and pose heavier threat to users than PC or smart phone. Victims could suffer from property or psychological damage, even life threaten. Since neither users nor device manufacturers can guarantee the safety, the operating system in smart home is duty-bound to take care of potential security and privacy issues for users.

What are the new issues

1. Identity and Access Management.

   Challenge The first challenge is access control around applications and events at service management layer. In personal computer and smart phone, access control is defined by users. When an application requests for some resources, the operating system asks for user permission first, then decide what kind of resources could be allocated depending on user response or authorization. In Edge operating system, users still have the highest privilege and authority to assign resources, but the system should consider more since it has tight relationship with physical world and devices.

   Unbalanced operations and corresponding security risk. Similar operations could have widely different security risks. For example, turning on/off light is less risky than performing the same operation on oven. Even to the same device, opposite operations takes different risk. Locking doors mistakenly will block authorized users and cause some inconvenience, while unlocking doors to burglars will threat home safety.

   Arbitrary events and resource trust. Largely deployed IoT devices are responsible for diverse tasks. Different applications accept various events and interact with the

www.manaraa.com

operating system, which increase the probability of being attacked. A novel security mechanism that can verify the identity of event source, and selectively make sensitive data visible to authorized apps needs to be developed.

Possible solutions To assign authorizations correctly, it is indispensable to classify privilege level for applications. Granting application permissions by users is reasonable, but this is not enough. As mentioned before, some users are not aware of potential security and privacy issues hidden in seems meaningless data, or hackers could make use of low privilege devices to achieve malicious attack to high privilege ones. In this situation, Edge operating system should be *smart* enough to have a pre-defined privilege level for all categories of applications, and have the ability to deny abnormal raised events. i.e., to introduce access control and intrusion detection into Edge operating system. Attribute-based encryption (ABE), proxy re-encryption, and lazy re-encryption are combined to propose a fine-grained data access control scheme in [132]. A security framework that to fight against intrusions to distance clouds, security attacks on mobile devices, and malicious access to cloud is proposed in [110]. To Edge operating system, it is a challenge to design fine-grained access control and intrusion detection mechanism to ensure the system security and preserve user privacy.

2. Connection Security.

   Considering structure and components in Edge operating system, challenge in connection security can be viewed from two aspects.

   Connection Diversity. Isolated with service management layer, communication layer is encouraged to have its own security mechanism. As mentioned before, versatile IoT devices in smart home have different hardware specifications and software systems. They may be deployed with variant operating systems and communicate protocols. Therefore, different levels of security solutions are taken on theses various devices considering their computational capabilities and communication channels. As a result, because of

www.manaraa.com

connection diversity, the overall level of security is determined by the device that has the minimum level of security. It is possible to utilize some security algorithms if all devices are using the same communication protocol. For example, bluetooth security mode uses Secure Simple Pairing (SSP) to achieve service level enforced security [105], ZigBee has 128-bit keys to implement its security mechanisms. But it is a huge challenge to design a universal security solution for all these protocols. One possible way is making security strategy spun off from device specifications, operating systems and communication channels.

Network Security. The predominance of wireless communication in Edge operating system raises great concerns on network security, especially for normal users who have little awareness on this kind of security. According to a community research, among the 439 million households using wireless connections, 49% of Wi-Fi networks are unsecured, and 80% of households even still use default passwords on their routers [9]. What's more, highly dynamic edge nodes also makes the network becomes vulnerable to attackers. Although it is possible to enhance community's awareness of security and privacy, other stake holders in Edge operating system, including service provider, system and application developers are supposed to take more responsibility on it.

3. Privacy.

   Like cloud computing raises community's attention on privacy, smart home in edge computing also has to face the same problem. Deployed with IoT, a smart home can generate a lot of privacy information along with sensed usage data. These data are even more sensitive than those in cloud computing since they generate directly from daily life, other than in the core network. For example, by reading the utility usage data, one can deduce the family demographic composition and living standards, even what time the house could be empty. In this situation, the challenge is how to provide

www.manaraa.com

reliable service and protect *data* and *usage* privacy at the same time. In Edge operating system, there are three main problems:

4. Secure Data Storage and Computation in Different Layers.

   In Edge operating system, data could be stored and computed in different layers, including sensor/device, gateway, and cloud.

   In each layer the outsourced data could be intentionally modified or abused because of no more physical possession by users. Also, data transmitted through different layers could suffer from malicious use by unauthorized third part applications. Researchers conducted an attack by snooping door lock pin code in [41]. In this case, how to make sure the data is under protection in each layer is a new challenge.

2.3 Optimization Metrics

In Edge computing, we have multiple layers with different computation capability. Workload allocation becomes a big issue. We need to decide which layer to handle the workload or how many tasks to assign at each part. There are multiple allocation strategies to complete a workload, for instances, evenly distribute the workload on each layer or complete as much as possible on each layer. The extreme cases are fully operated on endpoint or fully operated on cloud. To choose an optimal allocation strategy, we discuss several optimization metrics in this section, including latency, bandwidth, energy and cost.

Latency: Latency is one of the most important metrics to evaluate the performance, especially in interaction applications/services [68, 79]. Servers in Cloud computing provide high computation capability. They can handle complex workloads in a relatively short time, such as image processing, voice recognition and so on. However, latency is not only determined by computation time. Long WAN delays can dramatically influence the real-time/interaction intensive applications' behavior [101]. To reduce the latency, the workload should better be finished in the nearest layer which has enough computation capability to the things at the Edge of the network. For example, in the smart city case, we can leverage

www.manaraa.com

phones to process their local photos first then send a potential missing child's info back to the cloud instead of uploading all photos. Due to the large amount of photos and their size, it will be much faster to pre-process at the edge. However, the nearest physical layer may not always be a good option. We need to consider the resource usage information to avoid unnecessary waiting time so that a logical optimal layer can be found. If a user is playing games, since the phone's computation resource is already occupied, it will be better to upload a photo to the nearest gateway or micro-center.

Bandwidth: From latency's point of view, high bandwidth can reduce transmission time, especially for large data (*e.g.,* video, etc.) [53, 16]. For short distance transmission, we can establish high bandwidth wireless access to send data to the edge. On one hand, if the workload can be handled at the edge, the latency can be greatly improved compared to work on the cloud. The bandwidth between the edge and the cloud is also saved. For example, in the smart home case, almost all the data can be handled in the home gateway through Wi-Fi or other high speed transmission methods. In addition, the transmission reliability is also enhanced as the transmission path is short. On the other hand, although the transmission distance cannot be reduced since the edge cannot satisfy the computation demand, at least the data is pre-processed at the edge and the upload data size will be significantly reduced. In the smart city case, it is better to pre-process photos before upload, so the data size can be greatly reduced. It saves the users' bandwidth, especially if they are using a carriers' data plan. From a global perspective, the bandwidth is saved in both situations, and it can be used by other edges to upload/download data. Hence, we need to evaluate if a high bandwidth connection is needed and which speed is suitable for an edge. Besides, to correctly determine the workload allocation in each layer, we need to consider the computation capability and bandwidth usage information in layers to avoid competition and delay.

Energy: Battery is the most precious resource for things at the Edge of the network. For the endpoint layer, offloading workload to the edge can be treated as an energy free

www.manaraa.com

method [86, 32]. So for a given workload, is it energy efficient to offload the whole workload (or part of it) to the edge rather than compute locally? The key is the trade-off between the computation energy consumption and transmission energy consumption. Generally speaking, we first need to consider the power characteristics of the workload. Is it computation intensive? How much resource will it use to run locally? Besides the network signal strength [37], the data size and available bandwidth will also influence the transmission energy overhead [28]. We prefer to use Edge computing only if the transmission overhead is smaller than computing locally. However, if we care about the whole Edge computing process rather than only focus on endpoints, total energy consumption should be the accumulation of each used layer's energy cost. Similar to the endpoint layer, each layer's energy consumption can be estimated as local computation cost plus transmission cost. In this case, the optimal workload allocation strategy may change. For example, the local data center layer is busy, so the workload is continuously uploaded to the upper layer. Comparing with computing on endpoints, the multi-hop transmission may dramatically increase the overhead which causes more energy consumption.

Cost: From the service providers' perspective, *e.g.,* YouTube, Amazon, etc., Edge computing provides them less latency and energy consumption, potential increased throughput and improved user experience. As a result, they can earn more money for handling the same unit of workload. For example, based on most residents' interest, we can put a popular video on the building layer edge. The city layer edge can free from this task and handle more complex work. The total throughput can be increased. The investment of the service providers is the cost to build and maintain the things in each layer. To fully utilize the local data in each layer, providers can charge users based on the data location. New cost models need to be developed to guarantee the profit of the service provider as well as acceptability of users.

Workload allocation is not an easy task. The metrics are closely related to each other. For example, due to the energy constraints, a workload needs to be complete on the city

www.manaraa.com

data center layer. Comparing with the building server layer, the energy limitation inevitably affects the latency. Metrics should be given priority (or weight) for different workloads so that a reasonable allocation strategy can be selected. Besides, the cost analysis needs to be done in runtime. The interference and resource usage of concurrent workloads should be considered as well.

2.4 Summary

In this Chapter, we illustrate our vision and understanding of Edge computing. We also give several case studies to further explain how Edge computing could be adapted to the current computing paradigm. Then we summarize the challenges in detail and bring forward some potential solutions and opportunities worth further research, including programmability, naming, data abstraction, service management, privacy and security and optimization metrics. In the next Chapter, we will introduce the design of SOFIE and talk about how we want to address these challenges using SOFIE.

www.manaraa.com

Chapter 3: SOFIE - Smart Operating System For Internet Of Everything

In this Chapter we introduce the design of SOFIE, a Smart Operating System For Internet of Everything. SOFIE will manage and maintain the connections between gateway and Edge devices via multiple communication channels. We also argue that following four fundamental features should be supported in SOFIE to guarantee a reliable system, including Differentiation, Extensibility, Isolation, and Reliability (DEIR). Moreover, we also introduce lifetime management model in service management layer to detect the remaining lifetime for batteries and Edge devices in the system, as well as SURF model to optimize the system performance. SOFIE provides programmability for Edge application practitioners to design and implement their applications on SOFIE. SOFIE also supports a naming mechanism to support better service management and device management. Data management layer is introduced in SOFIE to isolate Edge device and Edge applications/services, which cloud help protect privacy and security of the data. In data management layer we will support data quality management, data abstraction, and programming interface.

www.manaraa.com

3.1 Related work

In this Section, we summarize research topics and prior works that are closely related to this dissertation.

3.1 Distributed system

The distributed network computing has been widely adopted in the computer framework nowadays. Grid [46, 45, 47] is proposed as a large-scale distributed framework which can use the widely distributed computing resources for one computing task. Unlike cloud computing, each node in the grid could perform a different computing task or run a different application.

Cyber Foraging [19] is proposed to utilize the the computing resources on the ubiquitous network dynamically through nearby high-performance computing infrastructures. Although Goyal *et al.* [51] developed a lightweighted infrastructure for cyber foraging and can be deployed on resource-constrained devices, this virtual machine based solution is still too heavy in order to run on Edge devices.

Guan *et al.* [54] proposed a grid service infrastructure for mobile devices, which allowes users to access Grid services using mobile devices However, this system cannot be deployed in wireless sensor networks or IoT devices, which makes it is not a good candidate for Edge computing.

Slingshot [115] is a system which can deploying mobile grid services at wireless hotspots, and the deployment mechanism could be adopted by SOFIE for computing distribution.

Automatically partition applications such as Coign [64], Globus [44], Condor [118], and Legion [89] could also contribute to SOFIE as resource partition methods for client-server computing model. However, the APIs and drivers for various communication methods and devices are missing in those solutions.

www.manaraa.com

Cannataro *et al.* [27] discussed the possibility of using the semantic web on an open grid service architecture, which also inspires us for supporting the semantic service on SOFIE.

3.1 Edge computing applications

Inspired by low-latency analytics, edge computing [107] (a.k.a. fog computing [23], cloudlet [103]) is proposed to process data at the proximity of data sources. To leverage computing resources on edge nodes, mobile device cloud [39], femto clouds [59], mobile edge-clouds [42], and Foglets [104] have been proposed to orchestrate multiple edge devices for intensive applications that are difficult to run on a single device. Differencing from these systems, SOFIE leverages not only mobile devices and the cloud, but also edge nodes to complete big data processing task collaboratively, while aforementioned systems are not for large scale data processing and sharing among multiple stakeholders.

GigaSight [113] has been proposed as a reversed content distribution network using VM-based cloudlets for scalable crowd-sourcing of video from mobile devices. GigaSight collects personal video at the edges of Internet with denaturing for privacy that automatically applies contributor-specific privacy policy. The captured video in GigaSight is tagged for search and shared using network file system (NFS). However, GigaSight is designed to share video data and cannot apply video analytics functions. In contrast to GigaSight, SOFIE provides APIs for data owners to create customized video analytics functions, which can be used by a user to compose his/her IoE application.

Vigil [137] is a distributed wireless surveillance system that prioritizes video frames that are most relevant to the user's query and maximizes the number of query-specified objects while minimizing the wireless bandwidth cost. Vigil partitions video processing (e.g., object/face recognition or trajectory synthesis) between edge nodes and the cloud with a fixed configuration. Differencing from this prior work, SOFIE allows a user to define workload partitioning and deployment and provides dynamic workload migration (e.g., JVM migration) depending on the available resources on the edge nodes and the cloud.

www.manaraa.com

Wang *et al.* [124] propose OpenFace that is an open-source face recognition framework to provide real-time face recognition/tracking by using edge computing (i.e., cloudlet [103]). Integrated with video stream denaturing, OpenFace selectively blurs faces depending on user-specific privacy policy. However, OpenFace leverages only edge computing whose computation is conducted on edge nodes. In contrast to OpenFace, SOFIE leverages both edge nodes and the cloud to reduce the response latency and network bandwidth cost. A programming interface is provided to manipulate data from multiple data sources.

Panoptes [69] presents a cloud-based view virtualization system to share steerable cameras among multiple applications by moving the cameras in a timely manner to the expected view for each application. A mobility-aware scheduler prioritizes virtualized views based on motion prediction to minimize the impact on application performance caused by camera moving and network latency. Zhang *et al.* [134] propose VideoStorm to support real-time video streams analytics over large clusters. An offline profiler generates query-resource quality profile, and an online scheduler allocates resources to each query to maximize performance on quality and lag based on the quality profile. Resource demand for a query can be reduced by sacrificing the lag, accuracy, and quality of outputs. These prior works are orthogonal to *Firework*. Panoptes can be adopted by SOFIE to provide customized camera views which are most relevant to user interests. The online scheduling algorithm of VideoStorm can also be used to adjust the resources allocated for a *Firework.View*. Furthermore, SOFIE can reduce impact on application performance (e.g., latency and network bandwidth cost) by carrying out the analytics on edge nodes, while both Panoptes and VideoStorm assume video data are preloaded in the cloud and clusters, which is infeasible given the scale of zettabytes data.

Ananthanarayanan *et al.* [15] present a geo-distributed framework for large-scale video analytics that can meet the strict requirements of real-time. The proposed framework in [15] leverages public cloud, private clusters, and edge nodes to carry out different computation modules of vision analytics. The prior works, Panoptes [69] and VideoStream

www.manaraa.com

[134], are integrated with the framework to optimize the resource allocation and minimize latency. SOFIE differs from [15] because our work expands data sharing along with attached computing modules and provides programming interfaces for users to compose their IoE application.

3.1 Performance of Edge computing applications

Performance evaluation for Edge Computing applications has been studied in extensive research work [85, 81, 67, 119, 131, 100]. In [138], Zeigel *et al.* proposed the notion of weighted diagnostic distortion (WDD), which is a novel measure for data quality in a health care related application. Based on this work, a general methodology for developing quality of information for body sensor design was proposed in [17]. In this work, the authors claim that the performance of the application should be evaluated based on their capability in finishing the desired job. However, the work above did not give a general framework that can be used in evaluate the performance of Edge Computing applications from different requirement aspects.

In wireless sensor network area, various sensor selection schemes are developed to make trade off between power conservation and quality of service [98, 56, 24]. Moreover, several work mentioned that component selection is a important issue in system design [61, 77].

In Edge Computing we want to put the computing at the proximity of data sources. This have several benefits compared to traditional Cloud based computing paradigm. Here we use several early results from the community to demonstrate the potential benefits. Researchers built a proof-of-concept platform to run face recognition application in [129], and the response time is reduced from 900ms to 169ms by moving computation from cloud to the Edge. In [58], the researchers use Cloudlets to offload computing tasks for wearable cognitive assistance, and the result shows that the improvement of response time is between 80ms to 200ms. Moreover, the energy consumption could also be reduced by 30-40% by cloudlet offloading. CloneCloud in [32] combine partitioning, migration with merging, and

www.manaraa.com

on-demand instantiation of partitioning between mobile and the cloud, and their prototype could reduce 20x running time and energy for tested applications.

3.2 Overview of SOFIE

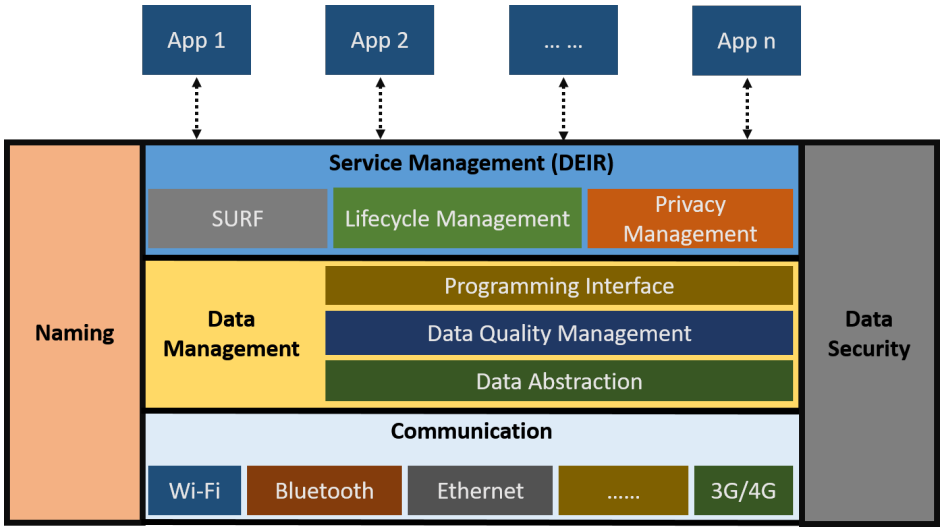

Figure 3.1: The structure of SOFIE.

Figure 3.1 shows the structure of a SOFIE. SOFIE needs to collect data from mobile devices and all kinds of things through multiple communication methods such as Wi-Fi, Bluetooth, ZigBee or a cellular network. Data from different sources needs to be fused and massaged in the data management layer. In data management layer, data abstraction model will fuse and massage the data into one table. On top of the data abstraction layer is the service management layer. Requirements including **D***ifferentiation*, **E***xtensibility*, **I***solation*, and **R***eliability* (DEIR) will be supported in this layer. In the following section, this issue will be further addressed. The naming mechanism is required for all layers with different requirements. Thus, we leave the Naming module in a cross-layer fashion. Challenges in naming are discussed later. Moreover, SOFIE will also manage the wellness of the system, including life-cycle management to handle the battery-life and component failure on Edge devices, and SURF to choose the optimized combination of Edge devices for different services based on their performance requirements such as battery-life, data precision, data quantity, etc.

www.manaraa.com

We have described five potential applications of Edge computing in the previous section. To realize the vision of Edge computing, we argue that the systems and network community need to work together. In this section, we will further bring forward some potential solutions for addressing these challenges using SOFIE, including *programmability*, *naming*, *data abstraction*, *service management*, *privacy and security* and *optimization metrics*.

3.3 Service Management Layer

In terms of service management at the Edge of the network, we argue that following four fundamental features should be supported in SOFIE to guarantee a reliable system, including Differentiation, Extensibility, Isolation, and Reliability (DEIR). In order to support DEIR features in SOFIE, we propose three models in service management layer, which are SURF, lifecycle management, and privacy management. SURF could be very helpful in supporting differentiation, isolation and extensibility of the system, since it can choose the most suitable devices for different tasks. Lifecycle management model is used to maintain the reliability and wellness of the system. At last, privacy management model is proposed also to support the isolation as well as the reliability requirement.

3.3 SURF

In the design and development of Edge computing applications, practitioners usually face several performance requirements from service providers, end users, and/or hardware constrains. Moreover, there could be a large number of available choices for different components in the application. To help the practitioners to select the optimal component combinations that can meet the performance requirements and reduce cost as much as possible at the same time, in SOFIE we will use the performance evaluation metrics we have proposed in SURF and also use the methodology for using performance vector. Moreover, when a critical services comes with higher priority, SURF could guarantee the critical service to use the desired device first, and provide lower quality device to other services as replacement, in this way, differentiation feature is supported in SOFIE. If one application failed, SURF

www.manaraa.com

could also make sure that the devices it is assigned will not be affected, so that other applications that share the same devices could still work normally, in this way, isolation feature is supported. Since SURF will be responsible for choosing the most suitable devices for each application, extensibility feature is also supported by SURF for easy replacement and new device adoption. The design and performance of SURF will be discussed in detail in the next Chapter.

3.3 Lifecycle Management

In a typical Edge computing system such as smart home, the number of Edge devices could be very large. Moreover, most of the devices are battery powered. According to our previous research [28], some device could provide very low quality data when the battery is about to die. In this case, it is very important for SOFIE to be able to detect and remind the user that some devices are running out of battery and need to be recharged before it lose effectiveness or report low-quality data.

Another goal of lifecycle management is detecting the remaining service life of the Edge devices. In Edge computing systems there could be some devices that are consumables with a limited operating life such as light bulb or fuse, etc. The lifecycle management

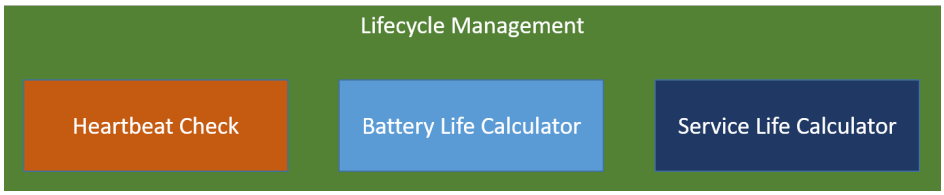

Figure 3.2: The Lifecycle management model of SOFIE.

model in SOFIE will detect and manage the lifetime of the Edge devices from three aspects, as shown in Figure 3.2. First is the heartbeat check. SOFIE will do a period life check for all the inactive devices. If they don't response of the check, SOFIE will then notify the system administrator for a maintenance. Second is the battery life calculator. SOFIE will ask for the time of last recharge and calculate the remaining battery life, and if the device is capable of reporting the remaining battery life, SOFIE could make the correction

www.manaraa.com

correspondingly. Last is the service life calculator. Based on the estimated MTTF/MTBF (Mean Time to Failure/Mean Time Between Failures) provided by the manufacturer and the time of installation, SOFIE could calculate the remaining service life and remind the user to be prepared for replacement when the device is not reliable due to wear out.

3.3 Privacy Management

Part of the Isolation feature is the requirement of privacy of the user. To protect the security of private data, Identity and Access Management (IAM) mechanism can be introduced to SOFIE. Different from traditional IAM system wherein assigns digital identity to users, SOFIE do identification of applications. With a digital identity, application can tell system who it is, and access to its authorized resources or devices or function. What's more, access control is provided to triggered events (i.e., door lock application can trigger the door to lock), avoiding malicious applications spoofing unauthorized events.

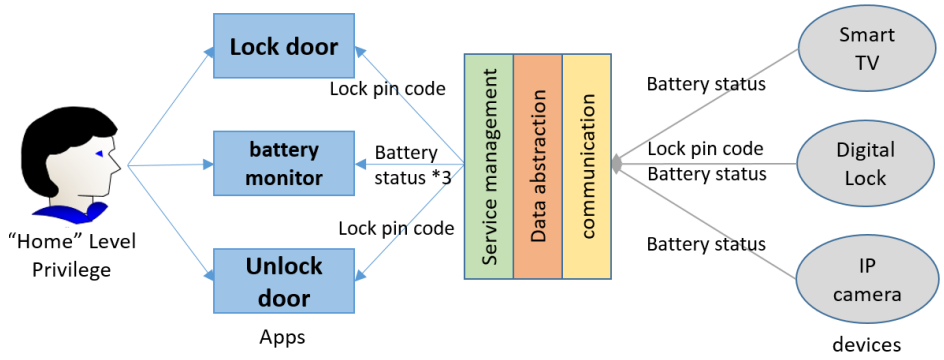

Figure 3.3: Identity and Access Management in smart home.

As shown in Figure 3.3, the user has the highest privilege, which is "home" level privilege. User authorizes applications to obtain a certain kind or kinds of capability and access to the corresponding resources. One application could have right to take charge of several devices for the same function and one device may also may communicate with several applications that taking care of different functions. On a fine granularity level, door lock and unlock are belong to different applications because of the asymmetric risks they have. Except these two apps which responsible for opening and closing the door respectively, digital locks

www.manaraa.com

also need to interact with a battery monitoring app that keeps tracking of its power status. This battery monitoring app works for other devices who require the same service at home as well, e.g., the IP camera or smart TV. The access management only allocates specific authority to the specified application, avoiding sensitive data dissemination effectively.

In many scenarios like wireless network, smart grid and cloud computing, one widely used privacy protect concept is privacy-preserving. A similar mechanism may be designed SOFIE. There are basically two types of privacy in SOFIE.

Data Privacy. To protect data privacy, periphery devices and sensors should be able to encrypt sensitive data before communicating with edge nodes. At edge, nodes can run privacy-preserving algorithm. In smart grid scenario, EPPA was proposed as an efficient and privacy-preserving aggregation scheme, using homomorphic Paillier cryptosystem to structure and encrypt multidimensional data [83]. Key generation algorithm can be utilized when edge nodes uploading data for further computation [49].

Usage Privacy. Usage pattern, which has been widely used to analysis user behavior in the context of cloud computing, smartphone, etc., can leak quantities of user privacy information. For example, during what time the house could be vacant, at what time the light in sitting room is turned on, and when will the householders fall to sleep. It is extremely easy for an intended person to monitor his target and schedule his malicious plan. Since there lacks a trusted third party, researchers proposed to hide privacy information by partitioning application which using offloaded resource in [130].

3.4 Data Management Layer

3.4 Data Abstraction

Figure 3.4: Data abstraction layer structure on SOFIE.

To address the data abstraction challenge, we envision that human involvement in edge computing should be minimized and the Edge node should consume/process all the data and interact with users in a proactive fashion. In this case, data should be preprocessed at

www.manaraa.com

the gateway level, such as noise/low-quality removal, event detection, and privacy protection, and so on. Processed data will be sent to the upper layer for future service providing. There will be several challenges in this process.

First, data reported from different things comes with various formats, as shown in Figure 3.4. For the concern of privacy and security, applications running on the gateway should be blinded from raw data. Moreover, they should extract the knowledge they are interested in from an integrated data table. We can easily define the table with id, time, name, data (*e.g.,*{0000, 12:34:56PM 01/01/2016, kitchen.oven2.temperature3, 78}) such that any Edge thing's data can be fitted in. However, the details of sensed data have been hidden, which may affect the usability of data.

Second, it is sometimes difficult to decide the degree of data abstraction. If too much raw data is filtered out, some applications or services could not learn enough knowledge. However, if we want to keep a large quantity of raw data, there would be a challenge for data storage. To address this challenge, we will add SURF to SOFIE's data abstraction layer.

Lastly, data reported by things at Edge could be not reliable sometime, due to the low precision sensor, hazard environment and unreliable wireless connection. In this case, how to abstract useful information from unreliable data source is still a challenge for IoT application and system developers. In the future work, we will add a data quality detection model to SOFIE to solve this issue.

One more issue with data abstraction is the applicable operations on the things. Collecting data is to serve the application and the application should be allowed to control (*e.g.,* read from and write to) the things in order to complete certain services the user desires. Combining the data representation and operations, the data abstraction layer will serve as an public interface for all things connected to SOFIE. Furthermore, due the heterogeneity of the things, both data representation and allowed operations could diverse a lot, which also increases the barrier of universal data abstraction.

www.manaraa.com

3.4 Data Quality Management

In Edge computing systems, data could easily falling into a certain pattern due to the usage pattern by user. To provide better service to Edge computing applications and devices, we will leverage the current data mining and machine learning algorithm to train a model for data quality detection model in SOFIE, as shown in Figure 3.5. This model could automatically detect abnormal data pattern from historical data record, and further analyze the reason for the abnormal pattern, which could be user behavior changing, device failure, communication interfacing, or system under attack. Moreover, in the Edge computing sys-

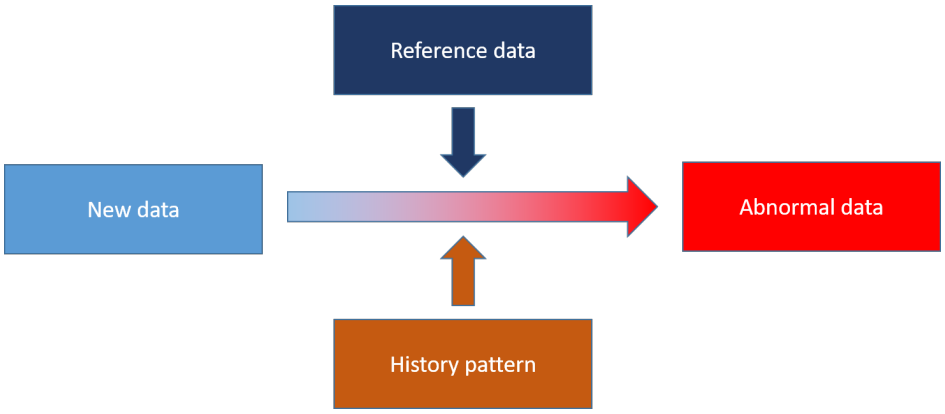

Figure 3.5: The data quality management model of SOFIE.

tem, same data could be reported from multiple source. For example, the room temperature could be reported from various thermometer, temperature controller, or weather forecast from online service. If one device report abnormal data, then the data from other sources could be used as the reference to detect if the abnormal comes from device defective.

3.4 Programming Interface

Without a flexible interface, developers would have to put a lot of effort into implementing applications for edge environments. They need to carefully coordinate Edge gateway, Edge devices and the cloud in their implementation, guaranteeing that their applications could enjoy the benefits provided by edge computing.

www.manaraa.com

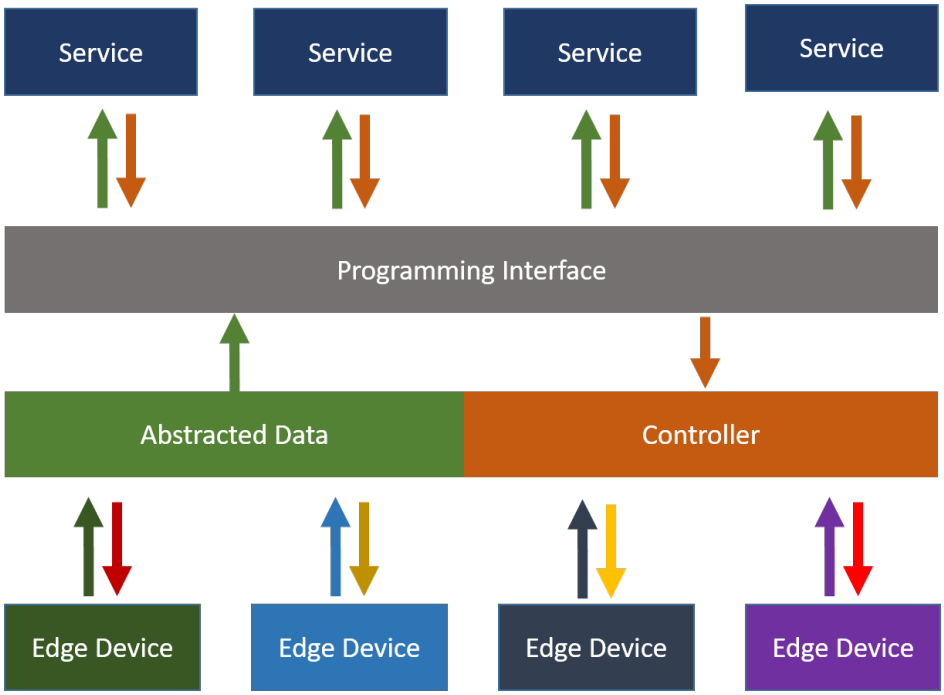

Figure 3.6: The programming interface in SOFIE.

In order to provides a flexible programming interface, through which the developer could implement user-specified policies with very little effort, helping the system efficiently execute the application on the edge infrastructure from Cloud to edge device, we propose Program Interface in data management layer. The main design goal of this interface is to provide satisfactory performance for user applications with minimum developer effort.

In this interface, SOFIE will manage the collection of data form all the Edge devices on behalf of each service. SOFIE will also record the controller for all the Edge devices. The collected data will be abstracted into a single table, and the control commands for various Edge devices will be record in a centralized controller, as shown in Figure 3.6. In this case, service providers do not need to handle various data collection protocols and control commands for different Edge devices. Instead, they could easily use a unified interface for getting data and sending commands from SOFIE to handle multiple Edge devices.

www.manaraa.com

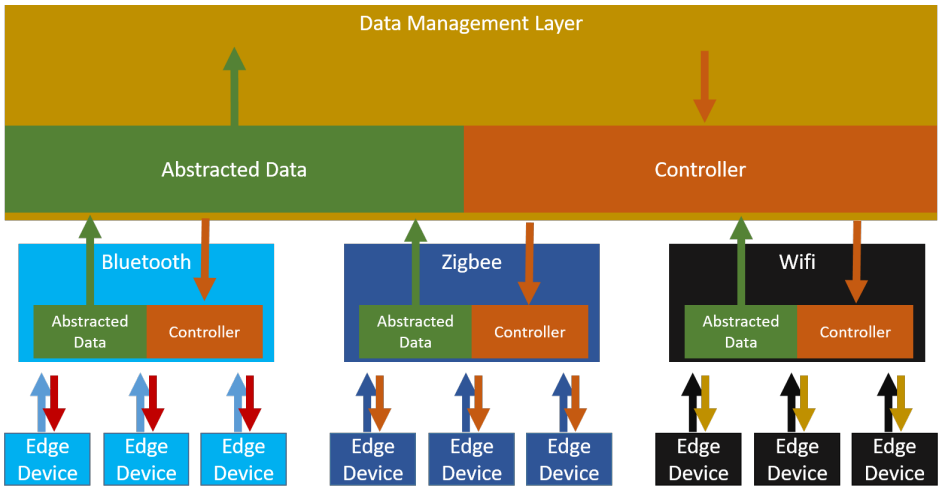

Figure 3.7: The communication layer in SOFIE.

3.5 Communication Layer

The communication layer will manage the connection between Edge gateway and Edge devious. In an Edge computing system, there could be various end devices connected to the gateway, and they might speak different languages such as Zigbee, Bluetooth, Wi-Fi, or cellular data such as 3G/4G. It would be a onerous task to handle the data collection and commands distribution through multiple communication protocols. In this case, the communication layer in SOFIE could integrate all the supported communication protocols into one package and open a unified interface to the upper layer, which is data management layer, to collect data from and distribute control commands to different devices, as shown in Figure 3.7.

3.6 Naming

Figure 3.8: The naming mechanism in SOFIE.

For a relative small and fixed Edge such as home environment, let SOFIE assign network address to each thing could be a solution. With in one system, each thing could have a unique human friendly name which describes the following information: location (where), role (who), and data description (what), for example, “kitchen.oven2.temperature3". Then

www.manaraa.com

SOFIE will assign identifier and network address to this thing, as shown in Figure 3.8. The human friendly name is unique for each thing and it will be used for service management, things diagnosis, and component replacement. For user and service provider, this naming mechanism makes management very easy. For example, the user will receive a message from SOFIE like “Bulb 3 (what) of the celling light (who) in living room (where) failed", and then the user can directly replace the failed bulb without searching for an error code or reconfigure the network address for the new bulb. Moreover, this naming mechanism provides better programmability to service providers and in the meanwhile, it blocks service providers from getting hardware information, which will protect data privacy and security better. Unique identifier and Network address could be mapped from human friendly name. Identifier will be used for things management in SOFIE. Network address such as IP address or MAC address will be used to support various communication protocols such as BlueTooth, ZigBee or WiFi, and so on. When targeting highly dynamic environment such as city level system, we think it is still an open problem and worth further investigation by the community.

3.7 Security

3.7 Connection Security

Considering the configuration in Edge computing which allowing computation to be performed at the edge of the network [108], massive computational edge nodes make it difficult to access for maintenance. To ease the network management and enhance network security, SDN (Software Defined Network) or similar architecture can be deployed in SOFIE. It may enable network security by controller’s central view of the network, and has ability to reprogram the data plane at any time. By allowing network administrators to manage network services through abstraction of lower-level functionality, SDN can support the dynamic, scalable computing and storage needs of SOFIE. It can benefit SOFIE’s security in several aspects.

www.manaraa.com

Network Access Control. Besides ABE we mentioned above, SDN technique also can be applyed in access control management. By assigning network only to authorized devices and sensors, system security could be improved at the device layer.

Network Traffic Scheduling. SDN technology can enhance data security by simplify extending VLANs (network segments) [48] beyond the building perimeter, or using VLAN ID to isolate malicious traffic. In SOFIE, variant devices and sensors compose nebulous network perimeters with vague boundaries, which makes it difficult to deploy security devices like firewalls. In this situation, SDN allows administrators to schedule routing for all periphery devices to go through one central firewall [8].

IDS and IPS. When network traffic flowing through a single point, SDN can help with real-time capture and analysis of Intrusion Detection System (IDS) and Intrusion Prevention System (IPS). *CloudWatcher* to provide monitoring services using OpenFlow in cloud computing, enabling administrator easily protect the dynamic network security is proposed in [111].

3.7 Secure Data Storage and Computation in Different Layers

In SOFIE, data could be stored and computed in different layers, including sensor/device, gateway, and cloud.

SOFIE could solve the challenges in the following three layers:

Communication Layer. Due to resource constraints, this layer is more responsible for data transmission rather than data storage and computation.

Data Abstraction Layer. In this layer, raw data from different sources will be fused and massaged, waiting for further utilization by upper layer. Acting as the backbone of the whole system, data abstraction layer is supposed to support verifiable computation [49]. Verifiable computation enables resource constrained devices outsource computation to randomly picked servers and verify returned proof to get trusted results.

Service Management Layer. Located on the top of SOFIE, service management layer featured with four elements (DEIR) to guarantee a secure system. This layer accepts pro-

www.manaraa.com

cessed data from data abstraction layer, as well as service requests sent from applications. In cloud computing, secure Third Party Auditor (TPA) was introduced to achieve a secure data storage. A privacy-preserving public cloud data auditing system combining homomorphic authenticator and random masking in cloud data storage is proposed in [123].

3.8 Summary

In this Chapter, we introduce the design of SOFIE, and explain the details about each layer of SOFIE, including Communication layer, Data Management Layer, Service Management Layer, Naming, and Security. Moreover, we further illustrate how SOFIE can help address the challenges of Edge computing in Chapter 2. SURF is part of the Service Management Layer and in the next Chapter, we will propose and evaluate SURF.

www.manaraa.com

Chapter 4: SURF - A Framework for Component Selection in Edge Computing Application Development

Wireless sensor network-based technologies and applications have attracted a lot of attention in the past two decades because of their huge potential to change people's way of life. These applications usually need close collaboration among multiple sensors, gateways, services and end users. When developing these applications, system designers and practitioners usually face several performance requirements such as the accuracy, battery life and system reliability. Given the hard requirements in system performance, how to choose an optimal combination from various sensors, algorithms and Edge computing systems to form the application is the most important problem that practitioners need to address. Ad hoc solutions were proposed in specific applications in the past; however, a general methodology that can be easily applied to future applications is lacking. In this work, we take the challenge and propose a general framework aiming to address the component selection problem, illustrate how this framework can be applied to real life applications through a case study, and discuss challenging issues and two interesting finds from our implementation.

4.1 Component selection in Edge Computing

The fast development and deployment of wireless communication technologies, and mobile devices, including sensors, robots, smart phones and tablets, have significantly changed the way we live [61, 77, 126, 82, 128, 91]. Wireless sensor network-based technologies and applications have been widely used in process management, health care monitoring, and environmental sensing, and so on. Trans-disciplinary collaboration is very common in these applications. For example, a wireless health application needs closely collaboration from health and medical research groups, mechanical engineers, computer scientists, doctors and nurses, and health insurance companies. System design is one of the most important tasks in the Edge Computing application development. There are several challenges that need to be addressed in the design of Edge Computing applications. One such challenge is meeting the performance requirements from service providers and end users. These requirements could

www.manaraa.com

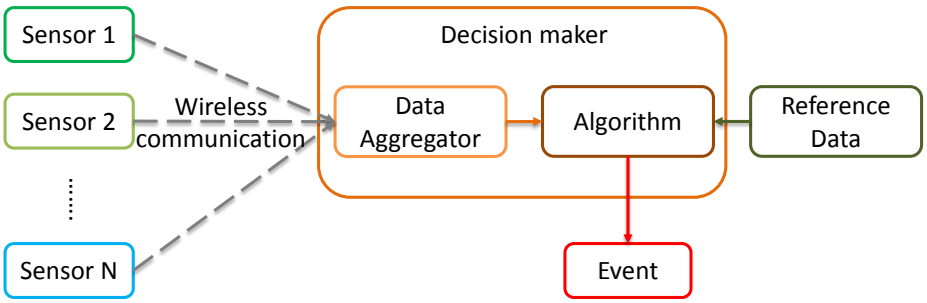

Figure 4.1: System overview of a Edge Computing application.

cover quality of information, battery life, hardware size and weight, and system cost, and so on [26, 125, 31]. Moreover, in some applications, the performance requirements could not be fixed, but are adaptive. For example, a sensor could working in a high power mode to achieve good performance when it is powered by a plug, and it could also turn to a worse performance to conserve energy and extend battery life when powered by the battery.

In designing Edge Computing applications, practitioners normally face a great deal of choices in the components of the applications. Figure 4.1 shows a conceptual view of a typical collaborative monitoring application's system structure. When a target event happens, raw data is collected on the spot by wireless sensors, and then sent to data aggregator through wireless communication channels such as Wi-Fi, Zig-bee or Blue-tooth. The raw data is then processed in the decision maker, and then a decision is made based on the result of the detection algorithm with processed data as input. In the decision making process, the decision maker could also refer to the data reported by the outside systems. For example, a fire detection system makes decision mainly on the smoke detection sensor, however, it could also use a real-time camera to find the fire source and use this information in making the decision.

Combination of system components such as sensors, data transfer methods, detection algorithms and reference data sources obviously have significant impact on system's performance. Moreover, the number of combinations could be considerable large and practitioners could feels no place to start when facing such a large dataset. In this way, how to choose the

www.manaraa.com

appropriate combination of system components that can meet the performance requirement with less cost is the key issue in Edge Computing application design. The practitioners could test all of the component combinations to find the best solution, if possible. However, it could be a tediously long process to go through the whole dataset. Best to the authors' knowledge, a much more efficient method to optimize the component selection is still a missing part in the Edge Computing area. To the best of our knowledge, a general framework for component selection has not been proposed yet in the literature.

4.2 Performance metrics vector

There are several performance requirements for a Edge computing application, including those from service providers, end users, and/or hardware constrains. In designing and developing Edge computing applications, practitioners need to understand the performance of the system very well in order to satisfy the requirements. With multiple requirements and a large number of component choices, it is very difficulty for the practitioners to evaluate and compare every choice. In this section, we propose performance vector, which is a metric can be used in evaluating the performance of different component combinations.

4.2 Performance requirements

Before designing the combination of components for an application, it is extremely significant for the practitioners to define the requirements of performance very clearly. Moreover, it is also very important that the proper metrics to describe the performance are selected. For instance, to demonstrate Quality of Information (QoI) [100, 36, 29], practitioners can choose root mean square error (RMSE), percentage RMS difference (PRD), or signal to noise ratios (SNR), and so on. Moreover, the requirements could cover various aspects such as accuracy, battery life, sensor size and weight, and so on. In this paper, we use R_x to indicate the performance requirements. For example, $R_{Accuracy} = 90\%$ means the application requires accuracy to be at least 90%, and $R_{BatteyLife} = 24$ hrs imply the system should work longer than one day before battery runs out.

www.manaraa.com

4.2 Performance score

To demonstrate the performance of a component combination i, we introduce $P_{i,x}$. For example, $P_{i,Accuracy}$= 90% means the accuracy of component combination i is 90%. We also come up with a performance score $S_{i,x}$ to denote the relationship between the performance and requirement for a component combination.

$$S_{i,x} = \begin{cases} P_{i,x} & \text{if performance is better than} \\ & \text{or equal to } R_x, \\ Null & \text{if performance is worse than } R_x. \end{cases} \tag{4.1}$$

From Equation 4.1 we can see that the performance score $S_{i,x}$ is $Null$ if the performance of component combination i cannot satisfy the requirement R_x. We also set the value of performance score to be $P_{i,x}$, so that the practitioners can easily compare the performance of different component combinations.

4.2 Performance vector

With performance score $S_{i,x}$ one can describe the component combination's performance for one requirement very clearly. Nevertheless, there are normally multiple requirements for a Edge computing application. In this case, practitioners need a metric to demonstrate a component combination's performance from several requirement aspects. To overall evaluate the performance of a component combination, we define performance vector V as follows:

$$V_i =< S_{i,1}, S_{i,2}, ..., S_{i,n} > \tag{4.2}$$

In this way, practitioners can easily eliminate the combinations that do not meet all of the performance requirements if there is $Null$ in their performance vector. Moreover, they

www.manaraa.com

can also compare the performance over multiple requirement aspects between component combinations intuitively.

4.2 A toy example

In order to better demonstrate how the performance vector can help practitioners in component combination selection, we build a very simple toy example. In this toy example, the target application has two requirements in performance, $R_{Accuracy}$ and $R_{BatteryLife}$. For the practitioners, there are four available component combinations, denoted as CC_1, CC_2, CC_3, and CC_4. After measurement and calculation, we get the performance vectors of these four combinations as $V_1 =< 0.3, Null >$, $V_2 =< 0.3, 3 >$, $V_3 =< Null, 4 >$, and $V_4 =< 0.5, 5 >$.

In this case, combination 1 can not be choose because $S_{1,BatteryLife} = Null$, which means the battery life of this component combination can not satisfy the requirement. Similarly, combination 3 should also be excluded since $S_{3,Accuracy}$ is $Null$. Combination 4 is the optimal choice because both $S_{4,Accuracy}$ and $S_{4,BatteryLife}$ are larger than that of combination 2, which means combination 4 has a better accuracy, while the battery life is longer than combination 2.

4.3 Methodology for using performance vector

As shown in the toy example in last section, with performance vector, practitioners can make the optimal choice of component combinations for an application. However, There could be a large number of possible combinations in a complex application, and only a small portion of them can meet the requirements. In this paper, we name these component combinations as 'possible combinations'. In order to conserve the workload and improve the efficient in system designing, practitioners should calculate the performance vector only for the possible combinations rather than the whole sets. In this section, we propose a methodology that can be applied easily to eliminate the impossible combinations and find the optimal choice through calculating the performance vector of possible combinations.

www.manaraa.com

4.3 Define performance requirement

Based on the performance requirements from service providers, end users, and/or hardware consratins, and so on, the practitioners need to define and quantize the requirements into a set of values $< R_1, R_2, R_3, ..., R_n >$.

4.3 List all the component combinations

In this step, the practitioners need firstly define the components in the system design, and then find all the available choices for each component. With the number of components m, and n_i for the number of available choices in component i, we can have

$$\text{Number of component combinations} = \prod_{i=1}^{m} n_i \tag{4.3}$$

4.3 Sort the list for each requirement

For one component, it is usually not difficult to order the available choices based on the given requirement by using the empirical approach or a simple experiment. In this step, practitioners should sort the list in previous step for each component and each requirement separately.

4.3 find and test the best combination for each requirement

When the practitioners have the sorted list of component combinations for each requirement, it is very simple to find the best performance combination for each requirement. Practitioners should then test the best combinations to see if they can meet the requirements. If any of the performance vectors has $Null$ inside, which means for some requirement, even the best combination can not satisfy it. In this case, practitioners should either consider lowering the requirement, or trying to change the system design. For example, leverage other technologies to make new components.

www.manaraa.com

4.3 Exclude impossible component combinations

To find the impossible component, practitioners can start with the best combinations in Step 4. By replacing only one component choice in the best combinations, practitioners can easily test if the replacement is possible or not for the application. For this test, practitioner should always start with the 'worst' choice in the sorted list in Step 3. This step has significant influence in reducing the workload for component combination selection. For example, in a dataset of 100 component combinations, if we can reduce the possible choices in one component from 5 to 4, which means we exclude only one choice for this component, the total dataset size will be reduced to $100 \times 4 \div 5 = 80$. In other word, we reduce 20 % of the dataset size by only eliminate one choice from one component.

4.3 Calculate performance vector for all the possible combinations

After exclude the impossible combinations, practitioners can calculate the performance vector for the remaining part. If any of the $S_{i,x}$ is $Null$, this combination i should be eliminated from the possible combinations, and there is no need to calculate other $S_{i,x}$ in V_i.

4.3 Find the optimal combination

Now practitioners should have a table of the performance vectors for all the possible combinations, and they can choose the optimal combination based on the highest score in the table. In some applications, there is no component combination that achieves best performance score for all the requirement. In this circumstance, the practitioners need to make a choice depend on the focus or key requirement.

4.4 Applying performance vector in a real application: Walking position detection

In this section, we declare how to use our methodology in a real Edge computing application. A wireless low-power fall detection application called Asgard [97] was used in this case study. An Asgard wireless sensor contains an accelerometer collecting acceleration, a flash memory that can store the data, and a CPU to process the data. Asgard sensor was

www.manaraa.com

fixed on the left ankle of each user, and each user was asked to walk 100 steps on flat ground and take stairs for another 100 steps. Practitioners wanted to identify the walking segment that is shown in the left part of Figure 4.2 and the stair segment that is depicted in the right part. In addition, they also wanted to count exactly how many steps were performed on flat ground and on stairs. The acceleration in the detection of gravity (Ag) was calculated by applying the Pythagorean Theorem.

$$Ag = \sqrt{x^2 + y^2 + z^2} \tag{4.4}$$

In this case study, the decision-making algorithm put a threshold on Ag and used it to identify the walking segment and the stair segment.

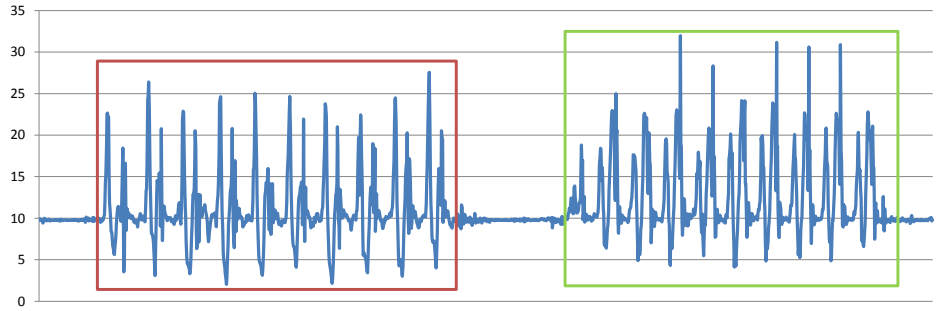

Figure 4.2: The walking segment and the stair segment.

4.4 Define performance requirement

In this application, there are three performance requirements that need to be satisfied:

Accuracy

There is no doubt that the quality of data is the most notable feature in Edge computing applications. No matter if we consider service providers or receivers, the fidelity of the application is always the highest priority issue to be taken care of [106, 17]. In order to measure the accuracy of the detection, we introduce three measures in statistics, which are precision, recall and F_{score}. In this application, there are two events, walking on flat ground and taking stairs, so the algorithm will have the following four outputs.

www.manaraa.com

True positive (TP): stair step correctly identified as stair step.

False positive (FP): flat ground step incorrectly identified as stair step.

True negative (TN): flat ground step correctly identified as flat ground step.

False negative (FN): stair step incorrectly identified as flat ground step.

Then we also list the definitions of precision and recall here.

$$Precision = \frac{TP}{TP + FP} \tag{4.5}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.6}$$

From the above equations we can deduce that both precision and recall scales are from 0 to 1, and in the ideal case, both precision and recall are equal to 1. In addition, we can understand that the larger precision and recall indicated the better performance our algorithm gives. In order to measure the accuracy of the system with precision and recall, we introduced F_{score}, which is the harmonic mean of precision and recall:

$$F_{score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.7}$$

We can also deduce that F_{score} scales from 0 to 1, and in the ideal case, $F_{score} = 1$. Similar to precision and recall, a larger F_{score} means our detection is more accurate. In this application, service providers came up with the requirement of F_{score} to be 0.9 for the fidelity of the data, so we set $R_{Accuracy} = 0.9$.

Battery Life

Because Asgard is powered by a battery, battery life is another key requirement to evaluate its performance [66, 87]. In this application, end user want that the Asgard sensor can work for more than one week without recharge the battery for convenience [136],

www.manaraa.com

which means the battery life should be longer than $24 \times 7 = 168$ hrs. In our expression, $R_{BatteryLife} = 168$.

Data quantity

The storage size of Asgard sensor is 512 MB, subjected to this hardware constrain, there should be a requirement in data quantity. In this case study, we set $R_{DataQuantity} = 512$ to restrict the amount of data collected by Asgard sensor.

As a conclusion, there are three performance requirements for this application, $R_{Accuracy} = 0.9$, $R_{BatteryLife} = 168$, and $R_{DataQuantity} = 512$.

4.4 Component combination analysis

Component with multiple choices

In this application, we found that there are three components that have multiple choices.

First is the number of axes of accelerometer we use in the application. Asgard sensor uses a 3-axes accelerometer, which means we can get the data from x, y, and z axes. In this case study, we have the following seven choices in axes: xyz, xy, xz, yz, x, y, and z. Using different number of axes does not affect battery life significantly since all the three axes are collecting data and we cannot shut down any axe while the sensor is working. However, it could affect the accuracy and data quantity a lot.

The second component that we can leverage is the sampling rate of the accelerometer. Asgard sensor allows us to configure the sampling rate to four distinctive values: 6 Hz, 15 Hz, 50 Hz, and 200 Hz. Using different sampling rate has influence on accuracy, battery life and data quantity.

The last component is decimal digits. In Asgard sensor, we can flexibly compress the data size to different decimal digits or even round it to integer. In this application, we tried four different decimal digits, which are a.bcd, a.bc, a.b, and integer part a only. Similar to the choice of axes, different decimal digits has influence on accuracy and data quantity,

www.manaraa.com

but can barely affect battery life because decimal digits is fixed to be a.bcd in raw data collection.

By applying equation 3, The total number of component combinations is $7 \times 4 \times 4 = 112$.

Sort the combinations

Common knowledge tells us that for the choice of axes, more axes could lead to a better accuracy, however, it could also use more space to store the data; and a higher sampling rate usually means a better accuracy, a larger data quantity and a shorter battery life, considering it could consume more power for data collection; similarly, more decimal digits stands for a better accuracy, but a larger data quantity.

Based on the above analysis, we sort all the component combinations for each of the requirements, as shown in the following table.

As we can see from this table, using more axes, more decimal digit, and higher sampling rate will take more space to store data and consumes more energy, however, accuracy could also be increased as a benefit.

The best combination

According to the above table, for accuracy, the best component combination is xyz + a.bcd + 200 Hz. We tested this combination and find out the accuracy $P_{Accuracy}$ is 1.0. Since $P_{Accuracy} > R_{Accuracy}$, it passes the test.

For battery life, the best combination is x + a + 6 Hz. The battery life is calculated by the following equation.

$$\text{battery life} = \frac{\text{battery capacity}}{\frac{\text{power dispassion}}{\text{working voltage}}} \tag{4.8}$$

www.manaraa.com

Asgard sensor has the battery capacity of 2100 mAh with working voltage at 4V , and we measured the power consumption for 6 Hz sampling rate is 29 mW. By applying Equation 8, we have $P_{BatteryLife} = 290$, which has a better performance than $R_{BatteryLife} = 168$.

For data quantity, combination x + a + 6 Hz is still the best combination. For each data entry, we need to have at least two Bytes to store it, one is sign bit and another is the number. Each second, this combination will generate $6 \times 2 = 12$ B data, and since $R_{BatteryLife} = 168$, the total data size $P_{DataQuantity}$ is $168 \times 3600 \times 12 = 7.2576$ MB, which is smaller than $R_{DataQuantity} = 512$.

Exclude impossible component combinations

In this case study, we first tested 200 Hz sampling rate. For 200 Hz sampling rate, the longest battery life it could achieve is using combination: x/y/z + 200 Hz + a. The $P_{BatteryLife}$ we got for this combination is 88, which is less than $R_{BatteryLife} = 168$. So in this test, we exclude 200 Hz from the possible choice.

We then replace the number of axes in the best combinations. We tried to use only two axes, which gives us the combinations xy/xz/yz + 50 Hz + a.bcd for the best accuracy. The test result is show in Figure 4.3.

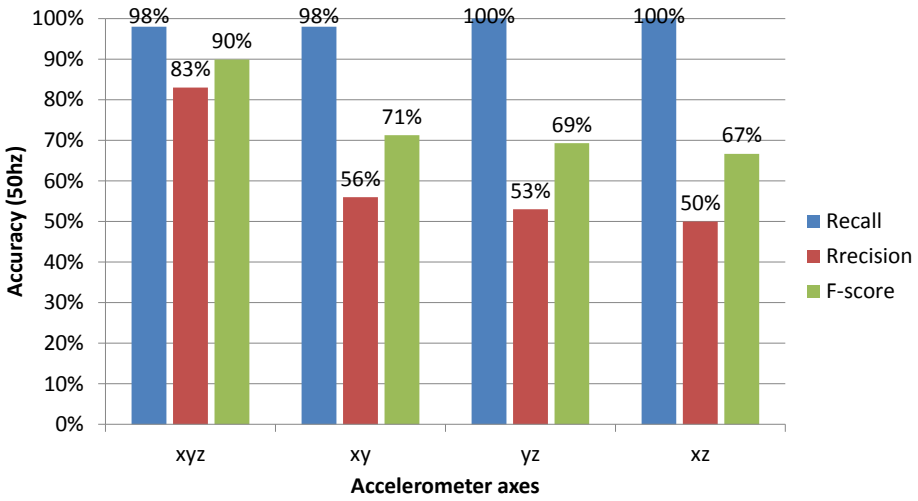

Figure 4.3: Accuracy of different axes combination.

www.manaraa.com

From Figure 4.3 we can see, only using all of the three axes can provide $P_{Accuracy} = 0.9 = R_{Accuracy}$, so in this test, we eliminate all the axes choices but xyz, since it is the only possible component choice.

We also tested 6 Hz sampling rate for the best accuracy requirement it can achieve. By testing combination xyz + 6 Hz + a.bcd, we got $P_{Accuracy} = 0.87$, which is smaller than $R_{Accuracy}$. So we exclude 6 Hz sampling rate from possible component choice list.

For decimal digits, we tested choice integer a. The best accuracy it can get is through combination xyz + 50 Hz + a, and the test result shows $P_{Accuracy} = 0.87$. This means only save the integer data is not accurate enough for this application.

Performance vector for all the possible combinations

After the above elimination, there are only 6 possible combinations left, we name them CC_1 to CC_6 and list all of them in Table 4.1.

Table 4.1: Possible Name and Component Combinations.

Name	Component combination
CC_1	xyz + 15 Hz +a.b
CC_2	xyz + 15 Hz +a.bc
CC_3	xyz + 15 Hz +a.bcd
CC_4	xyz + 50 Hz +a.b
CC_5	xyz + 50 Hz +a.bc
CC_6	xyz + 50 Hz +a.bcd

We then measure the performance of these 6 component combinations from accuracy, battery life and data quantity.

Accuracy:

In this case study, we calculate the F_{score} and show the result in Figure 4.4.

www.manaraa.com

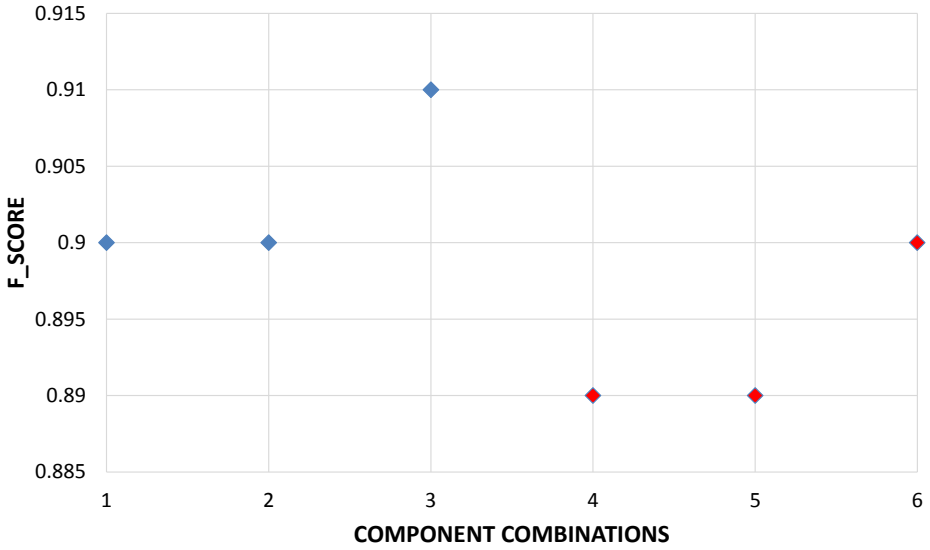

Figure 4.4: F_{score} of component combinations.

By applying Equation 4.1, we get the $S_{i,Accuracy}$ for all the six component combinations and list them in Table 4.2.

Table 4.2: Accuracy score for all the six component combinations.

$S_{1,Accuracy}$	0.9
$S_{2,Accuracy}$	0.9
$S_{3,Accuracy}$	0.91
$S_{4,Accuracy}$	Null
$S_{5,Accuracy}$	Null
$S_{6,Accuracy}$	0.9

From the table above we can see that both $S_{4,Accuracy}$ and $S_{5,Accuracy}$ are $Null$, which means component combinations CC_4 and CC_5 cannot be used in the application, so in the following measurement, we only consider component combinations CC_1, CC_2, CC_3, and CC_6.

Data quantity:

In this case study, for each data entry, we need to have one Byte to store sign bit and one more for each number. In Table 4.3 below we list the format of one data entry for

www.manaraa.com

component combinations CC_1, CC_2, CC_3, and CC_6.

Table 4.3: The format of one data entry for component combinations CC1, CC2, CC3, and CC6.

CC_i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	$\pm$	x	b_x	$\pm$	y	b_y	$\pm$	z	b_z						
2	$\pm$	x	b_x	c_x	$\pm$	y	b_y	c_y	$\pm$	z	b_z	c_z			
3	$\pm$	x	b_x	c_x	d_x	$\pm$	y	b_y	c_y	d_y	$\pm$	z	b_z	c_z	d_z
6	$\pm$	x	b_x	c_x	d_x	$\pm$	y	b_y	c_y	d_y	$\pm$	z	b_z	c_z	d_z

As we can see from the table, the length of one data entry for component combinations CC_1, CC_2, CC_3, and CC_6 respectively are 9 B, 12 B, 15 B, and 15 B. By applying the following equation:

$$\begin{aligned} P_{Dataquantity} = &\text{Sampling rate} \times \text{Battery life} \\ &\times \text{Length of one data entry} \end{aligned} \tag{4.9}$$

We can have the $S_{Dataquantity}$ for each component combination, as listed in Table 4.4:

Table 4.4: Data quantity for each component combination.

$S_{1,DataQuantity}$	81.684
$S_{2,DataQuantity}$	108.864
$S_{3,DataQuantityy}$	136.08
$S_{6,DataQuantity}$	453.6

All of these four component combinations meets the requirement for data quantity, which is $R_{DataQuantity} = 512$.

Battery Life:

www.manaraa.com

As we have discussed before, only sampling rate has influence on battery life, and since we have collected $P_{BatteryLife}$ for 6 Hz and 200 Hz in the previous tests, here we present all of the results in one figure. Firstly, we measured the power dissipation for all of the four sampling rates, as shown in Figure 4.5.

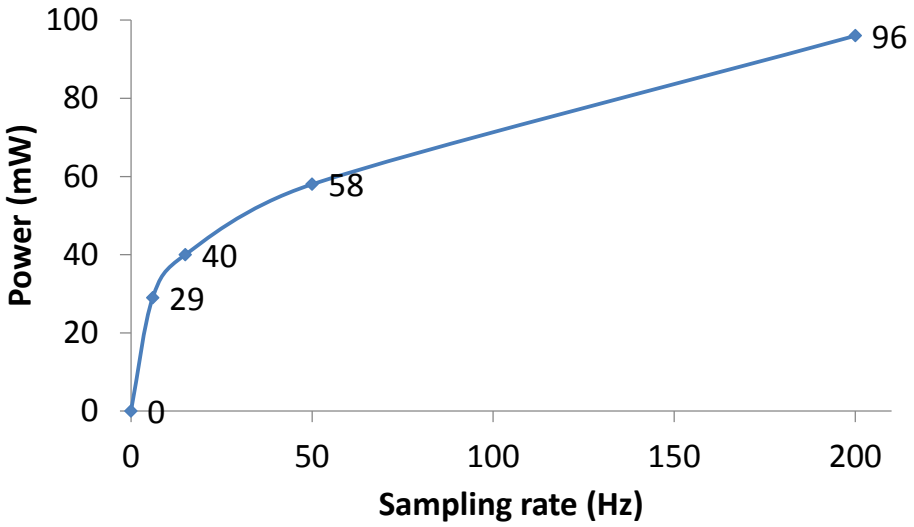

Figure 4.5: Effect of sampling rate on power dissipation.

By applying Equation 4.8 to the data in Figure 4.5, we get the $P_{BatteryLife}$ for all the four sampling rates, and the result is demonstrated in Figure 4.6. We continue calculating the $S_{BatteryLife}$ for all the four component combinations, and list the result in Table 4.5.

Table 4.5: BatteryLife score for all the four component combinations.

$S_{1,BatteryLife}$	210
$S_{2,BatteryLife}$	210
$S_{3,BatteryLife}$	210
$S_{6,BatteryLife}$	Null

Since $S_{6,BatteryLife} = Null$, we need to exclude CC_6 from the possible combinations.

Now we have three component combinations left, and all of them meet the requirements from accuracy, battery life and data quantity. We summarize their performance and get their performance vector in Table 4.6:

www.manaraa.com

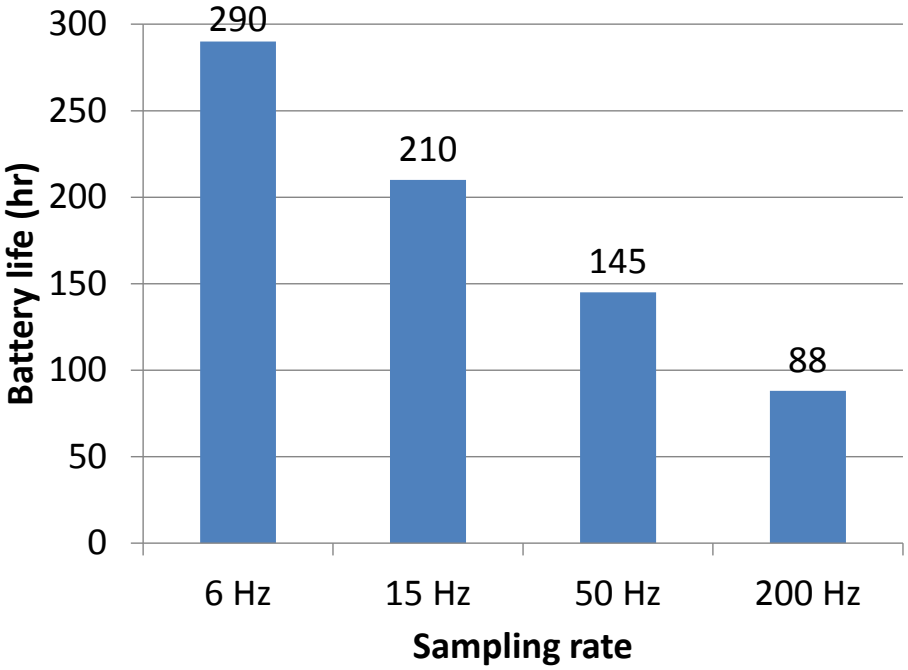

Figure 4.6: Effect of sampling rate on $P_{BatteryLife}$.

Table 4.6: Performance vector for all the three component combinations.

Combination	$S_{i,Accuracy}$	$S_{i,BatteryLife}$	$S_{i,DataQuantity}$
CC_1	0.9	210	81.684
CC_2	0.9	210	108.864
CC_3	0.91	210	136.08

www.manaraa.com

4.4 Find the optimal combination

From Table 4.6 we can see that all of the component combinations have the same performance in battery life, and preform very similarly in accuracy. However, since CC_1 uses much less storage space compared with CC_2 and CC_3, in this application, we select CC_1 as the best combination and use it in the real application developent.

4.5 Two interesting findings in the case study

When we are testing our methodology for using performance vector in our case study, we fully studied the relationship between data quality, battery life and data quantity for the application. We observed two interesting phenomenons during in case study. The first one is the effect of decision making algorithm on data quality, and the second one is different operation periods for battery.

4.5 Effect of decision making algorithm on data quality

In our case study, we first collected data using these four sampling rates and ran the classification algorithm on them, respectively. The threshold used in the algorithm was firstly set to 4.8 empirically, and the F_{score} was then calculated and shown in Figure 4.7. As expected, the F_{score} of the datasets has a positive correlation with the sampling rate, which suggests that with a larger quantity of data as input, decision makers can more easily make the correct diagnosis.

By analyzing the raw data collected at different sampling rates, we found that the information that is useful in diagnosis is presented in a more detailed fashion with a larger sampling rate. However, a problem was also introduced when increasing the sampling rate. The difference between the walking segment and the stair segment was reduced since both of them contain more details at a larger sampling rate, which could lead to a worse performance of classification even if the sampling rate is enlarged. Based on this observation, we applied a series of thresholds from 4.6 to 6.2 on the four datasets and tested their accuracy separately.

www.manaraa.com

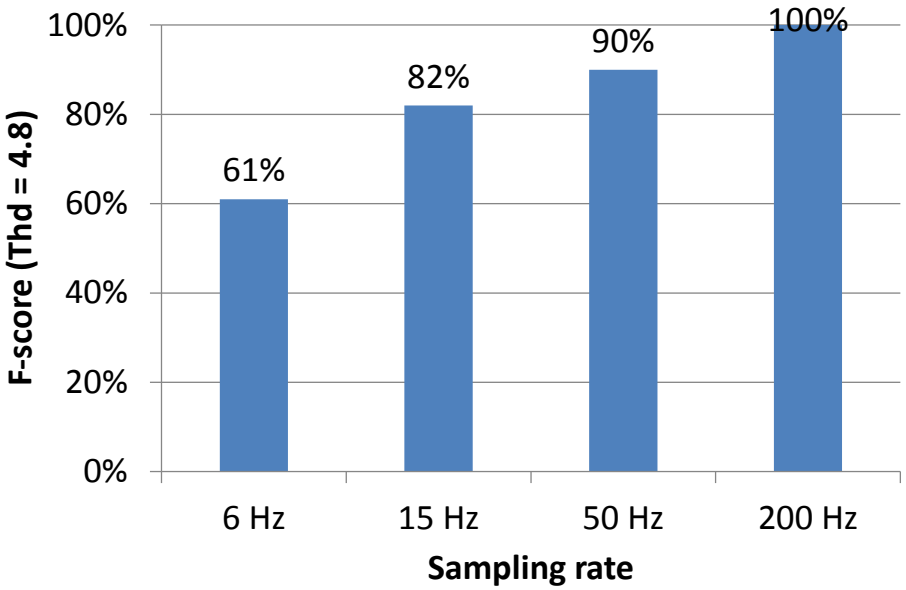

Figure 4.7: Effect of sampling rate on F_{score}.

Figure 4.8 and Figure 4.9 show the recall and precision, and the F_{score} is presented in Figure 4.10.

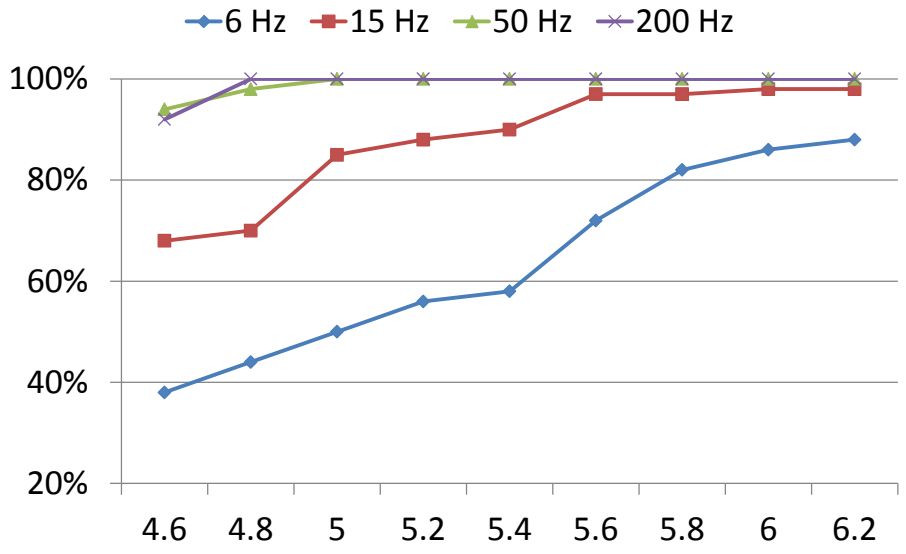

Figure 4.8: Effect of threshold on Recall.

We can see from Figure 4.8 that no matter which threshold is used, the recall will not decrease when enlarging the sampling rate, or in other words, it is easier to identify the step movements from the non-step ones such as standing still. Nevertheless, Figure 4.9 tells

www.manaraa.com

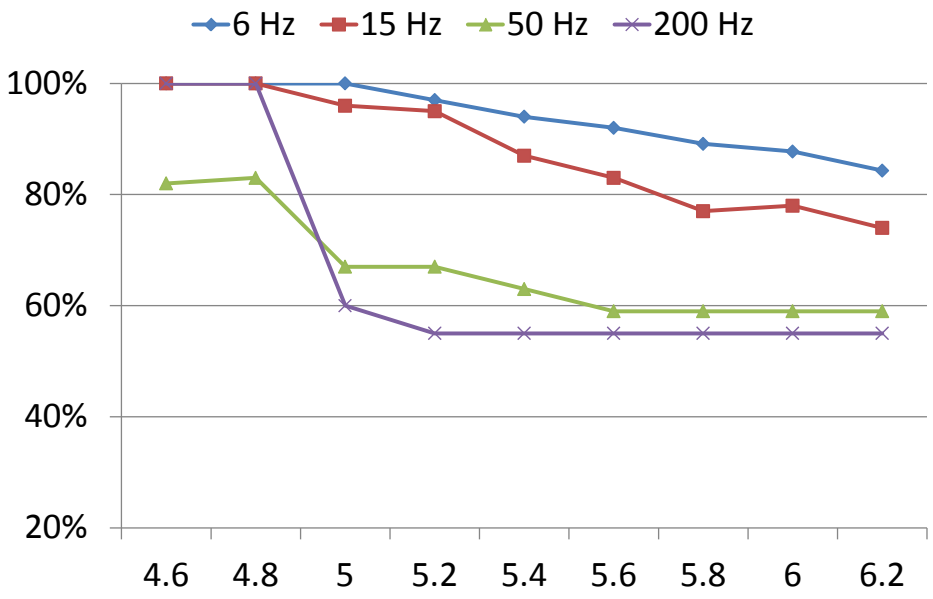

Figure 4.9: Effect of threshold on Precision.

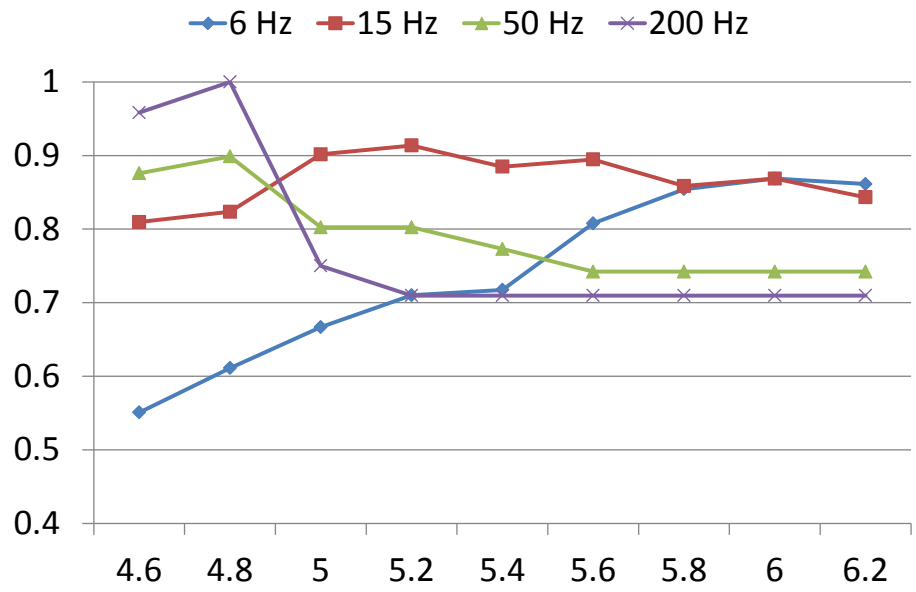

Figure 4.10: Effect of threshold on F_{score}.

us that in most cases, the precision decreases while the sampling rate increases from 6 Hz to 200 Hz, which verifies our deduction that it is more difficult to identify the stair segments from the walking segments when the data is collected at a larger sampling rate. Then we took these two factors into consideration and got Figure 4.10, from which we can see that

www.manaraa.com

when the threshold is below 5, the performance increases with the sampling rate; however, if we take a look at the overall figure, the F_{score} of each sampling rate varies. For 50 Hz and 200 Hz, the accuracy of the algorithm decreases when enlarging the threshold, and for 15 Hz the F_{score} does not vibrate too much; however, for 6 Hz the performance becomes better if we use a larger threshold. This means each sampling rate reaches its highest F_{score} at different thresholds and the algorithm should be adapted when the sampling rate is changed to get the best accuracy.

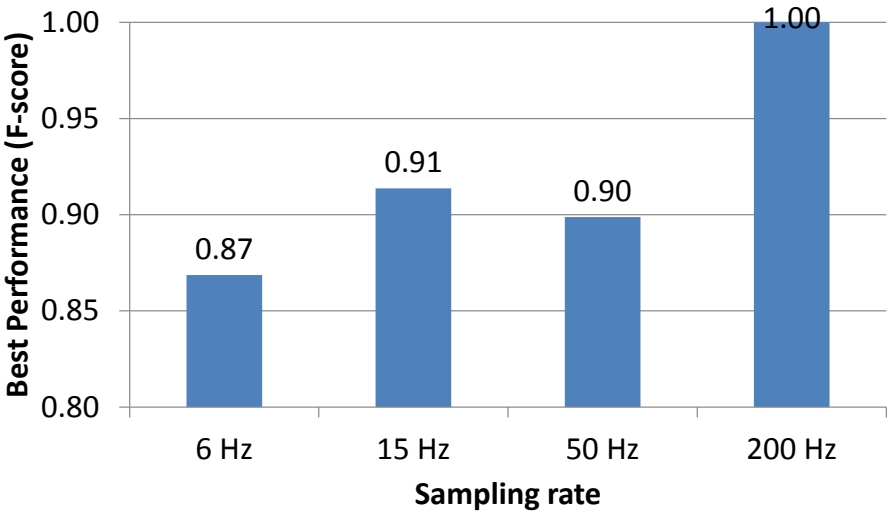

Figure 4.11: Highest F_{score} each sampling rate can get.

Based on the above observation, we extracted the best performance that each sampling rate can get from adaptive thresholds and showed the results in Figure 4.11. Comparing it with Figure 4.7 we can make the following two conclusions. First is that enlarging the data quantity in this case study will not increase the data quality too much. As shown in Figure 4.11, both 6 Hz and 15 Hz can get the F_{score} around 0.9, which is the highest F_{score} that 50 Hz can get. Second, if the practitioners have to exchange some data quantity for a longer battery life, they can modify the algorithm to shrink the gap of data quality introduced by this trade off. In some cases, the practitioners can attempted to use a cheaper or more energy saving sensor but still achieve a good performance if the corresponding decision making algorithm is properly modified [84, 122, 62]. We believe the second finding here

www.manaraa.com

can be applied to other wireless health applications and benefit their practitioners in similar situations [127].

4.5 Three operation period for battery

In the case study, we have observed that when the battery is about to running out, Asgard sensor works abnormally by logging incorrect data. To better understand the reason of this abnormality, the operating characteristic of batteries was analyzed first to give us a better understanding of a battery's discharge process [96]. We first fully charged the battery of Asgard sensor, then configured the sensor to continually log acceleration data at 200 Hz sampling rate until the battery runs out. Considering 88 hours of battery life is too long for us to monitor, we changed to a 400 mAh battery. During the whole process we kept measuring the working voltage of the sensor to get the discharge curve of the battery, as shown in Figure 4.12.

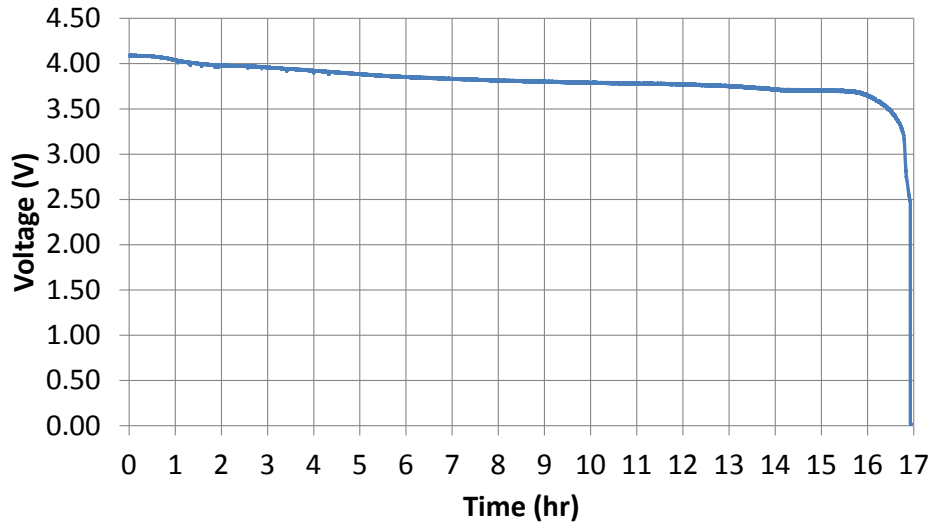

Figure 4.12: Discharge curve of Asgard sensor battery.

Figure 4.12 shows that the Asgard sensor can continually work for 17 hours for one charge, and the working voltage of the sensor drops from $4.1V$ to $2.5V$ until the battery runs out. Also, we can see from Figure 4.12 that during the first 16 hours of 17 hours

www.manaraa.com

working time, the working voltage is only changed subtly from $4.1V$ to $3.7V$. In the last hour, however, the working voltage decreases significantly until the sensor shuts down.

Working voltage has a significant impact on the performance stability of electronic devices. While the battery is running out, there is usually a certain stage in which the device can still be powered by the battery, yet the performance is not stable due to the low working voltage level. This phenomenon is more likely observed on a low-power device. For example, a flashlight could be dim or work intermittently when the battery runs out. Since wireless health applications generally use low-power sensors to guarantee a long battery life, such as in this case study the power dispassion of the Asgard sensor is lower than $100mW$, we suspect that this phenomenon could also appear in wireless health applications that involve low-power sensors. To verify this hypothesis, we performed an off-body analysis on an Asgard sensor. In the off-body analysis, one Asgard sensor powered by a battery was placed on a flat table as the treatment group, and we also set another Asgard sensor powered by a wire on the same table as the control group. We kept recording the acceleration reported by both sensors until the battery of the first sensor ran out. The results show that the wire powered sensor reports $9.8m/s^2$ throughout the process, and the data reported by the battery powered sensor is shown in Figure 4.13.

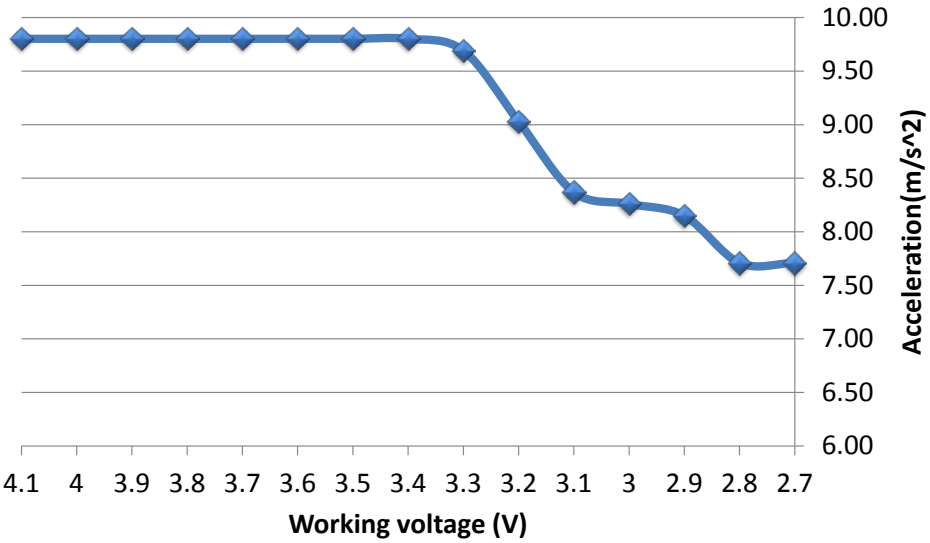

Figure 4.13: Effect of working voltage on Asgard sensor.

www.manaraa.com

Figure 4.13 shows that when the working voltage drops from $4.1V$ to $3.4V$, the sensor works normally and reports the correct acceleration. However, when the working voltage is below $3.4V$, the data reported by the sensor drops with the working voltage. Finally, when the working voltage is lower than $2.7V$, the sensor does not work. Based on this observation, we define the following three sensor operation periods:

Working Period (WP): when the battery capacity is sufficient so that the sensor can work normally.

Transfer Period (TP): when the working voltage is low and the sensor's performance is affected.

Non Working Period (NWP): when the working voltage is too low to support the sensor.

In this case study, the sensor first worked in the working period, which means the stage when the working voltage is between $4.1V$ and $3.4V$. Then the working voltage dropped from $3.4V$ to $2.7V$, which means the sensor worked in the transfer period. Last, the sensor went to the non working period when the working voltage was below $2.7V$. Obviously, The deviation between the values reported by these two sensors powered by the wire and battery affected the accuracy of the fall detection algorithm in the Asgard system. However, it is usually an arduous task to catch the transfer period during real usage of the sensor. In this case study, in order to evaluate the influence of the error introduced by the low working voltage on data quality, we used the normalized root-mean-square deviation (NRMSD) to show the difference between the values reported by the wire powered Asgard sensor and the values reported by the battery powered one, where lower values indicate less residual variance.

$$RMSD = \sqrt{\frac{\sum_{i=1}^{n}(x_{b,i} - x_{w,i})^2}{n}} \tag{4.10}$$

www.manaraa.com

$$NRMSD = \frac{RMSD}{x_{max} - x_{min}} \tag{4.11}$$

In Equation 4.10, x_b denotes the values collected by the battery powered sensor, and x_w denotes the data from wire powered sensor. Variable n was set to 1000, which means for every $0.1V$ from $2.7V$ to $3.4V$. We collected 1000 samples and used them to calculate the result. NRMSD of these two datasets is shown in Figure 4.14.

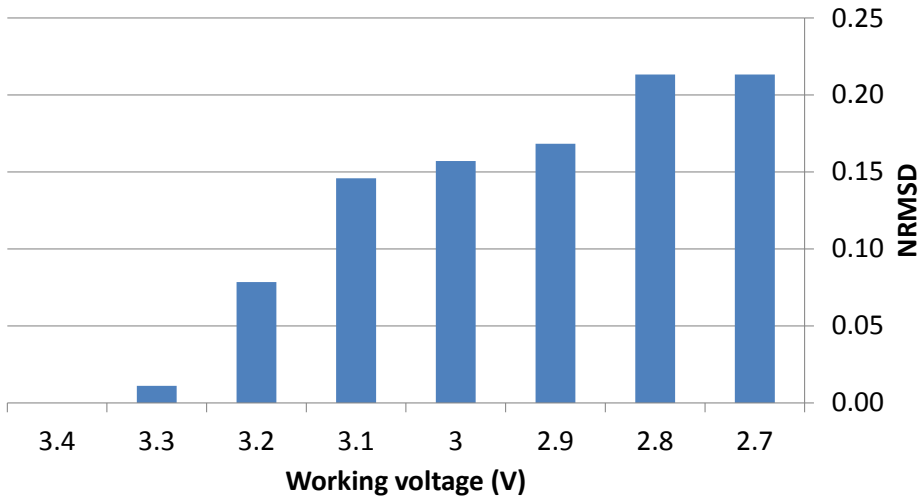

Figure 4.14: Data quality recession in the transfer period.

Figure 4.14 shows that data quality in the transfer period is not as good as when battery capacity is sufficient, and it becomes worse when working voltage keeps decreasing. In order to identify how long the transfer period lasts, we zoomed in on Figure 4.12 and located the points corresponding to $3.4V$ and $2.7V$. The result is presented in Figure 4.15. From Figure 4.15 we can see that the transfer period took about 15 minutes in the 17-hour working period. This may not be a large number, but if we extend the battery life to days or even months, the transfer period could also be prolonged to hours or days. Since the data collected in this stage is highly unreliable, the practitioners must be aware of this stage and take appropriate measures to avoid making incorrect decisions from it.

www.manaraa.com

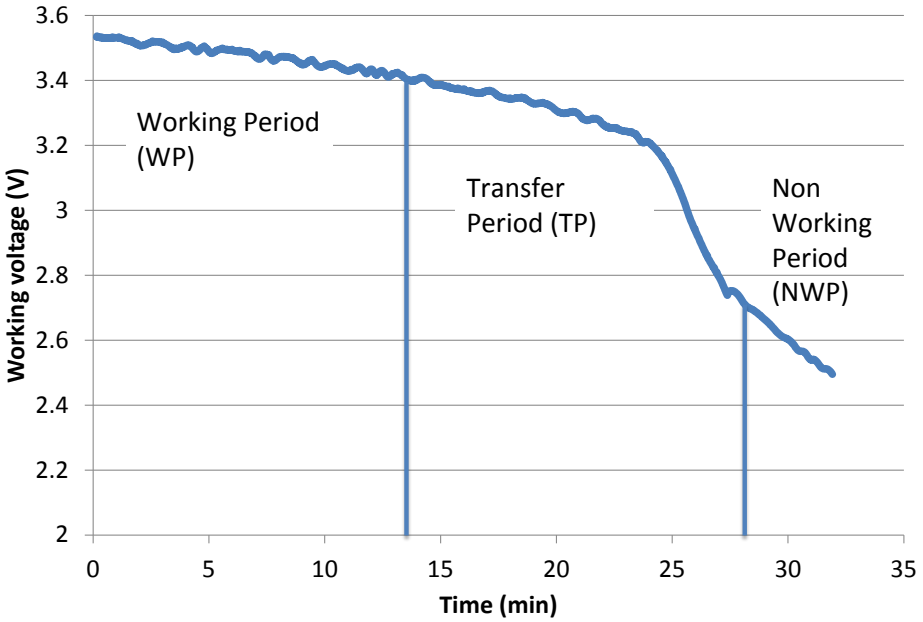

Figure 4.15: Three operation periods of battery.

4.6 Summary

In the design and development of Edge computing applications, practitioners usually face several performance requirements from service providers, end users, and/or hardware constrains. Moreover, there could be a large number of available choices for different components in the application. To help the practitioners to select the optimal component combinations that can meet the performance requirements and reduce cost as much as possible at the same time, we proposed performance evaluation metrics and a methodology for using performance vector. We tested our methodology through a case study, and the result showed that our methodology is very efficient in finding the optimal component combination. Finally, we discussed two interesting findings we have observed in the case study, one is the effect of decision making algorithm on data quality, and the other is different operation periods for battery.

www.manaraa.com

Chapter 5: Enabling Semantics in oneM2M Service Delivery Platform

With the burgeoning of Edge computing, data from Internet of Things (IoT) and Machine-to-Machine (M2M) systems are shared and consumed at the edge of the network. How to understand and leverage the data from other systems is a key challenge for information exchange and communication among IoT/M2M. Semantic paved the road for the knowledge-driven IoT/M2M systems. Context awareness becomes possible with the help of semantic information. In this chapter, we proposed the methods of support semantic service in a mature IoT/M2M system with reasonable overhead. Moreover, we also discussed the approaches of support secure mechanism such as access control directly in within the semantic information to boost the performance of the system. Our experiment shows that the proposed method could effectively and securely provide semantic service to the IoT/M2M system.

5.1 Introduction

New streams of data that previous unimaginable are generated by the Internet of Things (IoT) and Machine-to-Machine (M2M) communications, both in quantity and variety. Cisco Global Cloud Index estimated that 500 Zettabytes data will be generated by people, machines, and things by 2019 [65]. The sensors on a Boeing 787 will produce about 5 Gigabytes data every second [43]. The current vehicle will also generate one Gigabytes data every second [121]. Despite the proliferation of IoT/M2M, IoT/M2M is a fast-growing area and still in its infancy. New generation of things introduce new sensors to the community every day, and the new sensors generate new data and new functional requirements. However, it is very difficult for the IoT/M2M systems to interoperable with each other due to the nature that they are usually designed in a silo-based fashion. In IoT/M2M systems, data are collected from various resource, modeled in various data formats, and stored in various databases. This heterogeneous nature makes it extremely challenging for the IoT/M2M systems to exchange data and communicate with each other.

www.manaraa.com

In [109] the researchers came up with Edge computing as the new computing paradigm for the IoT/M2M applications and systems. Although Edge computing could improve the efficiency and reduce the cost in data collection and processing for IoT/M2M systems, the implementation of connecting multiple IoT/M2M systems on the Edge of the network is still very challenging due to the difficulty in understanding the data from each other. Built on top of the fundamental lightweight communication protocols such as HyperText Transfer Protocol (HTTP) and Constrained Application Protocol (CoAP), interfaces are often provided by modern constrained IoT systems for resource access, request, and discovery. Although the common communication protocols solved the connectivity issue, data from IoT/M2M systems are still difficult to be used by others due to the lack of common semantic information. How the data is encoded? How the access is controlled? What is the format, unit, and precision of the data? Similar problems bring huge challenges to the interoperability of IoT/M2M systems.

Semantic approaches discussed in [92] paved the road for the knowledge-driven IoT/M2M systems with context awareness. The effort to access, discover, and extract knowledge from the data would be significantly reduced by employing semantic technologies to the IoT/M2M platforms. Unlike semantic web, the data is stored and managed differently in the IoT/M2M systems where the devices are resource constrained, thus the semantic service should be enabled in a different mechanisms. There are several challenges in applying semantic techniques to the IoT/M2M systems. An IoT/M2M system usually employs its own database such as MySQL, SQLite, or MongoDB, etc. However, semantic technology such as the Resource Description Framework (RDF) usually require graph data-store. How to support semantic service for an IoT/M2M system in an affordable and secure manner meanwhile with reasonable overhead in storage size and resource retrieve time is remaining an open issue for IoT/M2M practitioners. In this chapter, we take the challenge and present our implementation of semantic service in oneM2M Service Delivery Platform (SDP) [117, 80], which is

www.manaraa.com

a scalable horizontal IoT/M2M system that conforms to the global oneM2M and ETSI TC M2M standard.

The rest of the chapter is organized as follows. Section 5.2 presents the background information of semantic technologies that can be applied in IoT/M2M systems and the brief description of oneM2M SDP. Section 5.3 explains the design of our solution which employs Jena semantic web framework. In Section 5.4 we propose two different approaches of supporting access control in semantic graph store alongside the SDP. The performance of our approach is evaluated in Section 5.5. At last, this chapter concludes in Section 5.6.

5.2 Background

In this section, we will first review the semantic technologies that could be employed by the IoT/M2M systems. Then we will introduce the design of oneM2M SDP to give the readers a brief overview background.

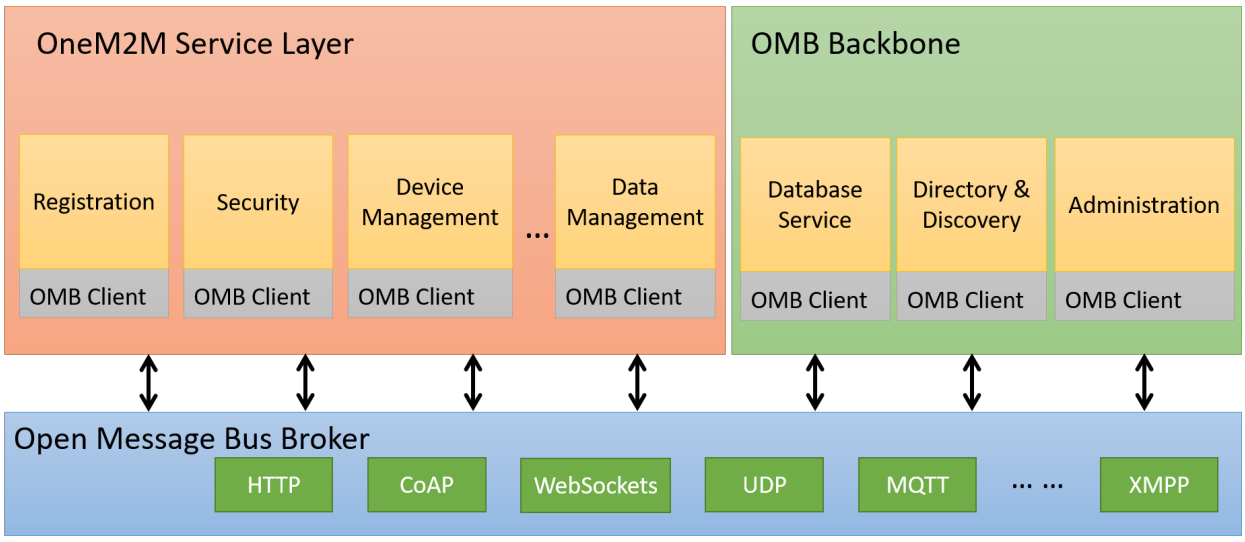

Figure 5.1: oneM2M SDP Architecture.

5.2 Semantic Technologies

In [75], the semantic technologies that could be applied to IoT/M2M systems are reviewed. Here we present the background knowledge of what we used in our project.

www.manaraa.com

- ***Resource Description Framework (RDF)*** RDF is a metadata model provided by World Wide Web Consortium (W3C) specifications [73]. It is proposed as a framework to represent information on the Web. RDF can connect the concrete syntax with the formal semantics based on the abstract syntax. Applying the entity-relationship conceptual modeling approaches [30], RDF data model describes the resource using statements expressed in the form of *subjectâĂŞpredicateâĂŞobject*. This form is also refereed as *triples*. In a triple, the subject refers to the resource itself, and the predicate expresses the relationship between the subject and the object.

  In RDF, the triples can be represented in a machine-readable manner. The subject, predicate, and object can all be addressed as unique URIs. For example, the statement "sensor1-measures-temperature" could be represented in RDF as

  "http://example.subject#**sensor1**

  http://example.predicate#**measures**

  http://example.object#**temperature**".

  The database that stores and retrievals the collection of triples via the query language is named Triplestore, and in this project we use Apache Jena [70] as our triplestore. Details background of Apache Jena will be presented in the next Section.

- ***RDF Schema (RDFS)*** RDF ontologies, or RDF vocabularies can be used to build resource structure in the triplestore, and the set of classes which enables this model is called RDFS [25]. Web Ontology Language (OWL) [21], which is the family of ontologies authoring languages, contains most of the RDFS.

- ***SPARQL Protocol and RDF Query Language (SPARQL)*** SPARQL [93] is the RDF query language for semantic triples store and retrieve. A couple of semantic operations such as discovery, query, and reasoning are supported by SPARQL. Apache Jena offers an interface to receive and execute SPARQL queries.

www.manaraa.com

5.2 oneM2M SDP

This work is finished during the Author's internship at Interdigital, Inc. The IoT/M2M system used in this chapter is Interdigital's oneM2M SDP, which is a scalable service delivery platform. oneM2M SDP is a standard-compliant M2M/IoT service platform that provides standardized End-to-End solutions. All the entities in M2M/IoT such as devices, gateways, servers, services, and applications. As a scalable service platform, oneM2M SDP could not only be deployed on the cloud using an universe architecture, but also run on resource constrained M2M/IoT devices and gateways. The service oriented design of oneM2M SDP leveraging RESTful architecture enables the efficient deployment of the applications and supports a various choice of IoT/M2M devices.

Figure 5.1 shows the architecture of oneM2M SDP. All the services in the oneM2M service layer connect to the Open Message Bus (OMB) through the infrastructure of OMB backbone. OMB clients play as the interface to abstract the connection between the underlying transport communication protocols (such as HTTP, CoAP, WebSockets, UDP, MQTT, XMPP, etc.) and the oneM2M service layer services. The API provided by the OMB clients will allow the services exchange messages with other services through the connection to the OMB broker. All the clients connected to the OMB are administered and monitored by the OMB Administration service.

This architecture specifies a process by which a primitive is received at the SDP and then it proceeds through a sequence of âĂIJoperationsâĂİ on the primitive and then pass the primitive to the âĂIJnext operationâĂİ in the lifecycle. The operations are captured in terms of a âĂIJserviceâĂİ performed on the primitive. The transition to the next operation or âĂIJserviceâĂİ is done using OMB messages.

All of the services will store data and query information from the database service in the OMB backbone. The database service will also provide a Generic Database Interface (GDI) to support various databases such as SQL and MongoDB without the awareness of

www.manaraa.com

the OMB. For the operations related to the database, **Create, Retrieve, Update, and Delete (CRUD)** are supported by the SDP.

5.3 Solution Design

In the previous section, we presented the background information of semantic technologies used in our work, and introduced oneM2M SDP, which is the IoT/M2M system that we are targeting. In this section, we will move on illustrate the design of our solution which enables semantic service in oneM2M SDP. First we will start with the basic semantic features that we want to support.

5.3 Basic semantic operations

In order to enable semantic service on oneM2M SDP, several basic operations should be supported.

- *Semantic annotation:* Semantic annotation refers to the operation of triple creation for the IoT/M2M devices. The annotated semantic information could later on be used for semantic discovery and query by other services and devices.
- *Semantic discovery:* Semantic discovery means the resource discover operation based on the semantic criteria. Giving the semantic filter, the semantic service should be able to find all the matched resources and send back the discovered list.
- *Semantic query:* Unlike semantic discovery which restrict the query result to be the resource (usually the unique ID or the URI of IoT/M2M devices), semantic query focus more on the knowledge that could be learned based on the semantic information of the system. For example, the overview statistics of the database or the detailed description of a resource.

5.3 Semantic service design

In order to support semantic annotation, discovery, and query in SDP, we add ⟨*semanticDescriptor*⟩ as the virtual child resource of the devices in SDP. By doing this,

www.manaraa.com

the semantic description of a device will be stored in ⟨*semanticDescriptor*⟩, and we can use the CRUD operation from the SDP to implement semantic annotation, discovery, and query.

Now we can store the semantic information in the SDP's native databse, which is MongoDB. However, MongoDB doesn't support RDF data model and SPARQL query language. To solve this issue and support semantic discovery and query, we introduce Apache Jena [70] in our solution as the RDF triple store and the SPARQL query engine.

To build the communication bridge between MongoDB and Jena, we introduced Semantic service to the SDP's oneM2M service layer, as shown in Figure 5.2.

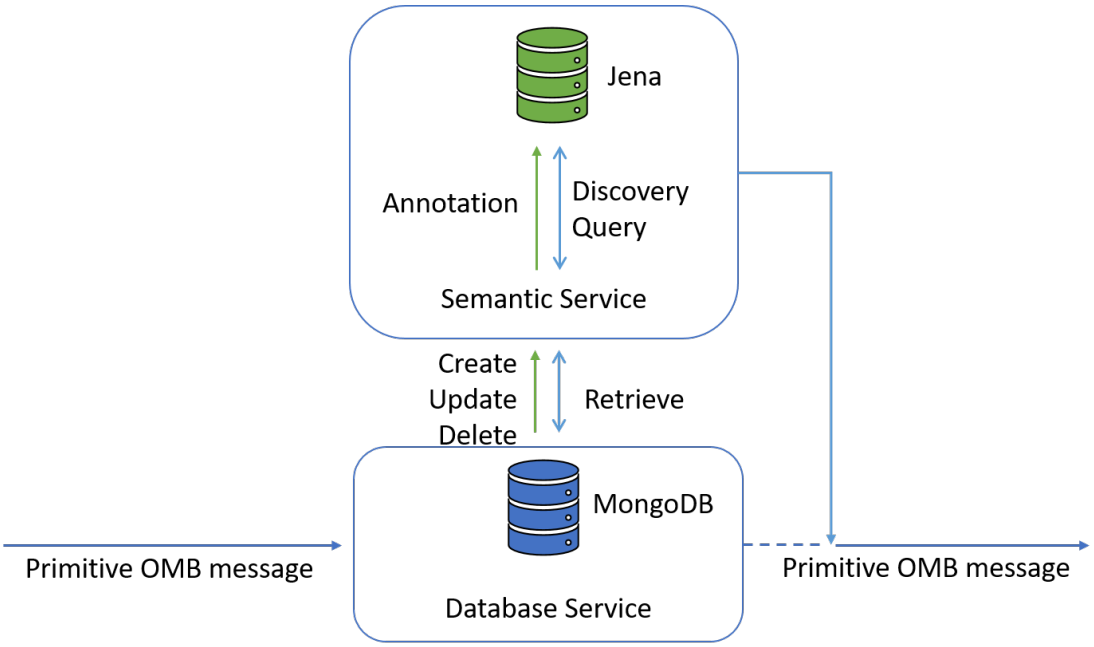

Figure 5.2: oneM2M SDP semantic service.

There are two main functions for the semantic service, which are semantic annotation for the resources with semantic information, and handle semantic based retrieve request as either semantic query or semantic discovery. When the database service of the SDP received a CRUD event message that contains semantic information, semantic service will perform the functions based on the event type. The detailed CRUD event processing procedure in semantic service is presented in Figure 5.3.

www.manaraa.com

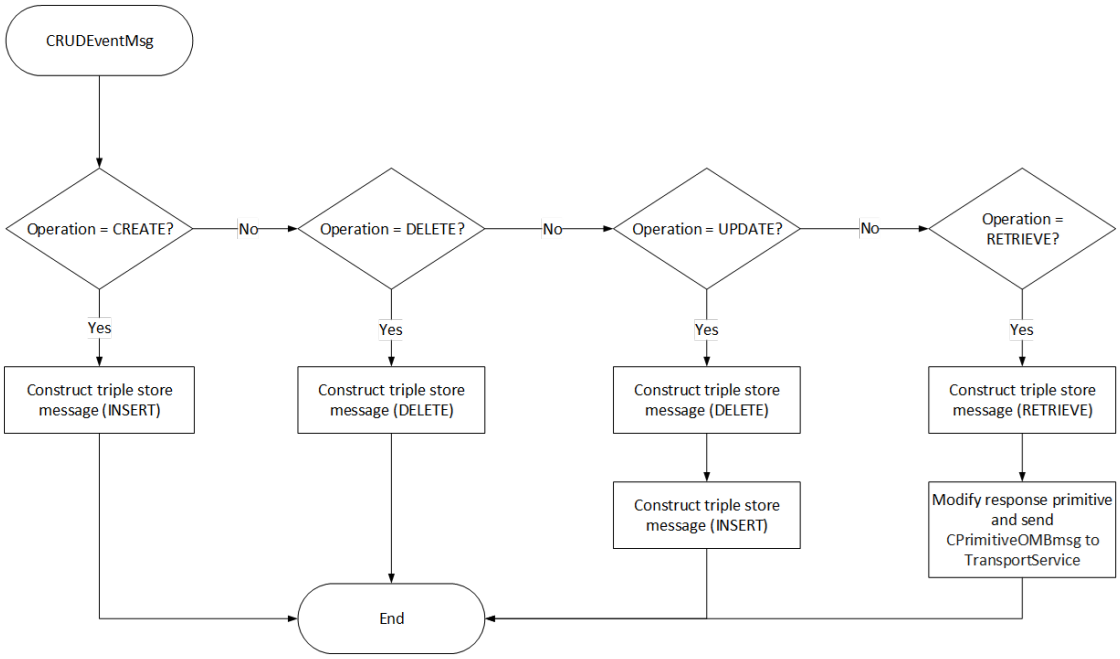

Figure 5.3: CRUD event processing in semantic service.

If the event type is Create, Update, or Delete event, the semantic service will automatically generate corresponding SPARQL request and perform the annotation operation in Jena, as shown in Figure 5.4. If the event is Retrieve, the semantic service will first decide whether the request is for semantic query or discovery based on the Semantic Query Indicator (SQI) in the primitive OMB message, and send SPARQL query to Jena for the operation. The query result, either resource ID, which is a URI in SDP, or the result string, which is a string, will be added to the primitive OMB message by the semantic service and send to the destination services in the SDP life-cycle, as shown in Figure 5.5 and Figure 5.6.

By this design of semantic service, we successfully implemented message convert and communication between the native database of SDP, which is the MongoDB, and our semantic RDF triple store, which is the Jena. However, there are still challenges in the syncronization between these two databases. In the semantic annotation lifecycle in Figure 5.4, Jena will perform the SPARQL request generated by the semantic service, however, the result is not send back to the original primitive OMB message. In this way, if there are failures during the Jena operation due to the incorrect semantic information in format or content,

www.manaraa.com

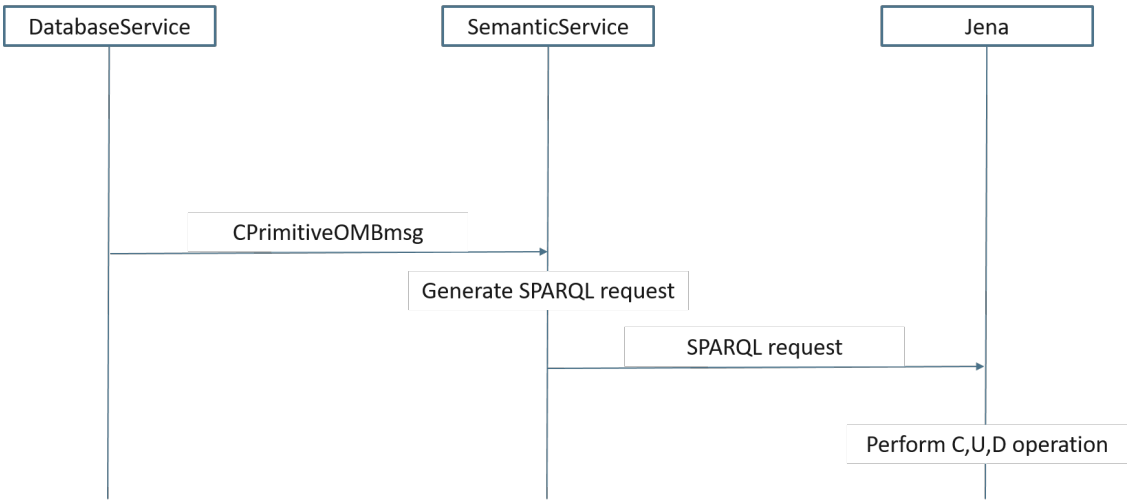

Figure 5.4: Semantic annotation lifecycle.

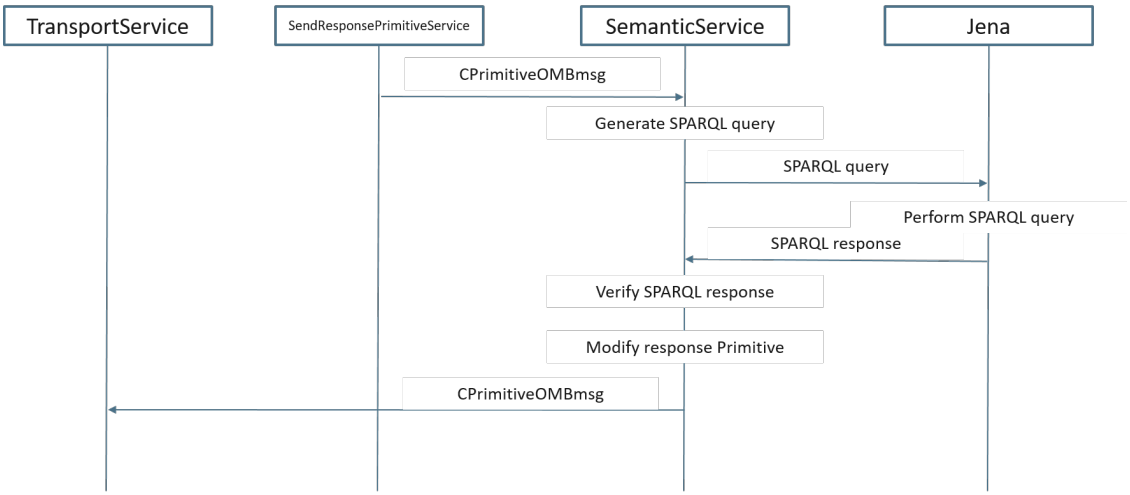

Figure 5.5: Semantic query/discovery lifecycle.

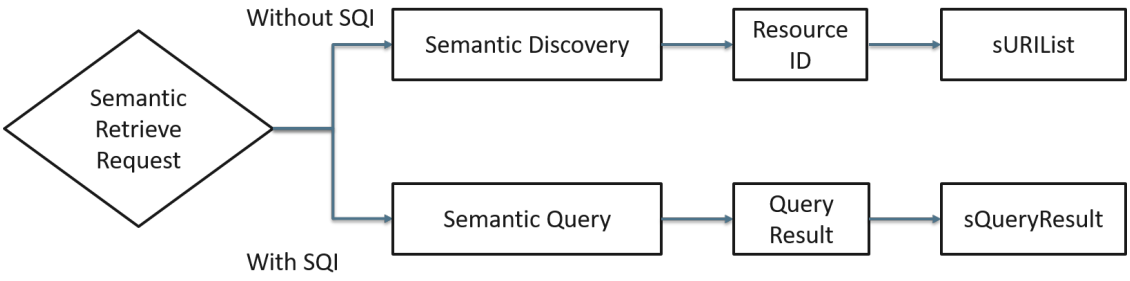

Figure 5.6: Semantic query indicator.

the SDP can not be informed and handle the errors correspondingly. To solve this issue, we implemented the triple validation feature in the semantic service, as presented in Figure 5.7.

www.manaraa.com

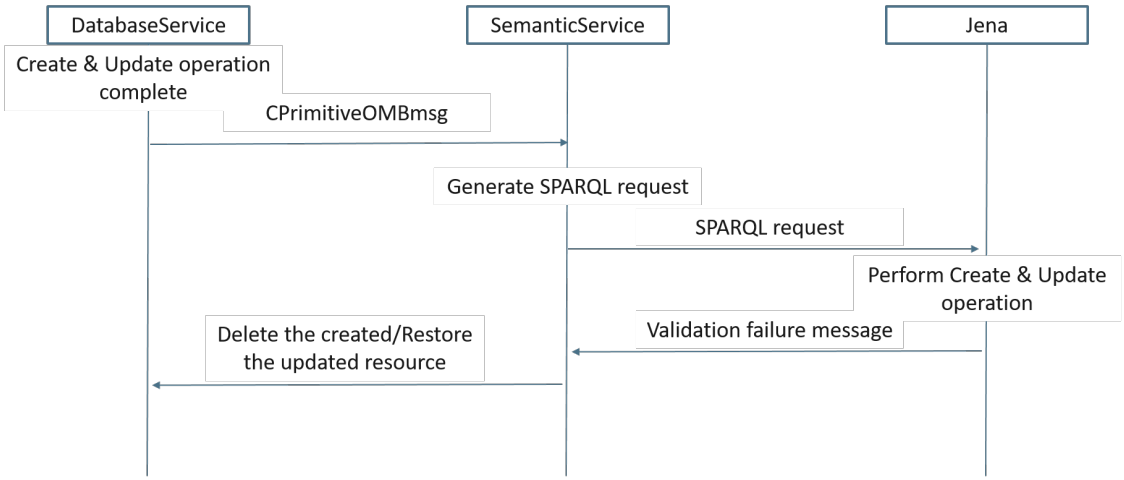

Figure 5.7: Triple validation in semantic service.

In our triple validation mechanism, the database service will send primitive OMB message to the semantic service after the create and update operation is completed in the MongoDB. Then the semantic service will wait for the response from Jena for the SPARQL request, and if the annotation operation in Jena is failed due to the incorrect semantic information, semantic service will embed the failure message in the primitive OMB message and notify the database service to delete the resource that are created, or restore the resource that are just updated. Finally, the message originator will be notified the failure and check the semantic information.

5.3 Case study

To better illustrate our solution in a more detailed manner and make it easier to understand, we will present a case study of the temperature sensor in a smart home in this chapter.

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.onem2m.org/ontology/houses_temperature_example#"
    xml:base="http://www.onem2m.org/ontology/houses_temperature_example"
    xmlns:temperature_example="http://www.onem2m.org/ontology/temperature_example#">

```

www.manaraa.com

```
    <owl:NamedIndividual rdf:about="http://www.onem2m.org/ontology/
     houses_temperature_example#House1">
       <rdf:type rdf:resource="http://www.onem2m.org/ontology/temperature_example#House
     "/>
       <temperature_example:hasIndoorTemperature rdf:resource="http://www.onem2m.org/
     ontology/houses_temperature_example#IndoorTempProperty1"/>
    </owl:NamedIndividual>

    <owl:NamedIndividual rdf:about="http://www.onem2m.org/ontology/
     houses_temperature_example#IndoorTempProperty1">
       <rdf:type rdf:resource="http://www.onem2m.org/ontology/temperature_example#
     TemperatureProperty"/>
       <temperature_example:hasDatatype>xsd:int</temperature_example:hasDatatype>
       <temperature_example:hasUnit rdf:datatype="http://www.w3.org/2001/XMLSchema#
     string">Fahrenheit</temperature_example:hasUnit>
    </owl:NamedIndividual>

    <owl:NamedIndividual rdf:about="http://www.onem2m.org/ontology/
     houses_temperature_example#IndoorTempSensor1">
       <rdf:type rdf:resource="http://www.onem2m.org/ontology/temperature_example#
     TemperatureSensor"/>
       <temperature_example:hasTemperatureProperty rdf:resource="http://www.onem2m.org/
     ontology/houses_temperature_example#IndoorTempProperty1"/>
       <temperature_example:hasResourceID>oneMPOWER-IN-CSE/tempsensorae1</
     temperature_example:hasResourceID>
    </owl:NamedIndividual>

</rdf:RDF>
```

www.manaraa.com

The code above shows how the semantic descriptor look like in oneM2M SDP. In this case, we have an example house named *House1*, and there is a temperature sensor *IndoorTempSensor1*, which has resource ID as *oneMPOWER-IN-CSE/tempsensorae1* and measures temperature in Fahrenheit.

The semantic query example is presented in the following code.

```
1
2 PREFIX temp: <http://www.onem2m.org/ontology/temperature_example#>
3 SELECT ?sensor WHERE {
4           ?sensor temp:hasTemperatureMeasuringFunction ?tempFunction .
5             ?tempFunction temp:measuresTemperature ?property .
6    ?property temp:hasUnit "Celsius"^^<http://www.w3.org/2001/XMLSchema#string>
7 }
```

This query will return the resource ID of the sensor which measures the temperature in celsius unit.

In this section, we present the design of our solution for semantic support of oneM2M SDP. In the next section, we will discuss different methods that we tested for supporting access control with semantic information.

5.4 Access Control

Security is one of the most important feature in any IoT/M2M system, in oneM2M SDP, access control policy is applied for each resource to ensure the secure data access. Since semantic data could contain private and security sensitive information, access control should also be supported in the semantic database.

In order to find the most efficient method of supporting access control with semantic data, we proposed and implemented three different solutions regrading store the semantic information in single or multiple graphs, and evaluated the performance of them correspondingly. In this section, we will introduce the design and implementation of these methods.

www.manaraa.com

Graph is the basic unit for RDF triple storage in Jena with the concept similar with table or form in other databases. Multiple graphs with unique names are supported in one Jena database.

5.4 Store semantic descriptor in one graph without access control

Since access control has already been implemented in oneM2M SDP, the most straightforward method would be leverage the access control policy in oneM2M SDP. In this method, oneM2M SDP will generate two query messages when the user is trying to find the resource with specific semantic criteria. First the SDP will query its own MongoDB to get a list with all the resource IDs which the user has access permission. Meanwhile the SDP will also generate a query message to Jena with the semantic criteria and find another list which holds all the resource IDs that matches the criteria. Finally the SDP will find the intersection of these two lists and return the result to the user.

Although this method is very easy to implement and all the semantic information could be stored in a single graph, the disadvantage of doing this is obvious. Both of the lists mentioned above could be very long and this will increase the query time significantly. We will look at this issue in details in the next section.

5.4 Store semantic descriptor in one graph with access control

Instead of only store semantic information in the RDF database, we could also save access control policy as RDF triples and use Jena to manage access permission, the detailed implementation of this method is also described in. To support this, we need to modify the original triples during the semantic annotation operation.

The following example shows how the original triples are modified in order to support access control in Jena.

```
1
2 HomeA     rdf:type            ex:Home .
3 HomeA     ex:hasLocation       LocationA .
```

www.manaraa.com

```
LocationA    ex:hasLatitude          âĂIJ300âĂİ .
LocationA    ex:hasLongitude         âĂIJ200âĂİ .
```

The code above shows a simple example of a semantic descriptor named $\langle SD-1 \rangle$ with only 4 original triples. To add access control policy as triples in Jena, the original triples need to be modified as the following format:

```

atomDescription1  rdf:type      sd:atomDescription .
<SD−1>            rdf:type      sd:semanticDescriptor .
atomDescription1  sd:hasSubject   HomeA .
atomDescription1  sd:hasObject      ex:Home .
atomDescription1  sdhasProperty   rdf:type .
atomDescription1  sd:describedIn  <SD−1> .
atomDescription2  sd:hasSubject   HomeA .
atomDescription2  sd:hasObject      LocationA .
atomDescription2  sd:hasProperty  ex:hasLocation .
atomDescription2  sd:describedIn  <SD−1> .
atomDescription3  sd:hasSubject   LocationA .
atomDescription3  sd:hasObject      âĂIJ300âĂİ .
atomDescription3  sd:hasProperty  ex:Latitude .
atomDescription3  sd:describedIn  <SD−1> .
atomDescription4  sd:hasSubject   LocationA .
atomDescription4  sd:hasObject      âĂIJ200âĂİ .
atomDescription4  sd:hasProperty  ex:hasLongtitude .
atomDescription4  sd:describedIn  <SD−1> .
```

And the following access control policy could also be applied in the triple store:

```

acp:ACP1      acp:applidedTo          ex:<SD−1> .
acp:rule1_1   rdf:type                acp:accessControlRule .
```

www.manaraa.com

```
acp:ACP1    acp:hasACPRule      acp:rule1_1 .
acp:rule1_1  acp:hasACOriginator  "User1" .
acp:rule1_1  acp:hasACOperations  "CREATE", "RETRIEVE" .
```

In this example, we add the access control policy named "ACP1" to the triple store, "ACP1" applied to semantic descriptor $\langle SD-1 \rangle$ and has one access control rule "rule1–1", which gives user "User1" the permission to perform create and retrieve operations.

The query message also needs to be modified in this method. Consider the original query like this:

```

select distinct ?home
where
{
  ?home   rdf:type     ex:Home .
  ?home   ex:hasLocation     ?location .
  ?location   ex:hasLatitude   "300" .
}
```

In order to query the modified triple store, the query message will be adjusted correspondingly as shown in the following code:

```

select distinct ?home
where
{
  ?accessControlRule      acp:hasACOriginator     ?Originator
  FILTER(?Originator = "User1")
  ?accessControlRule    acp:hasACOperations    "RETRIEVE" .
  ?accessControlPolicy  acp:hasACPRule     ?accessControlRule .
  ?accessControlPolicy  acp:appliedTo      ?semanticDescriptor1 .
```

www.manaraa.com

```
10  ?accessControlPolicy   acp:appliedTo      ?semanticDescriptor2 .
11  ?accessControlPolicy   acp:appliedTo      ?semanticDescriptor3 .
12
13  ?atomDescription1   sd:describedIn     ?semanticDescriptor1 .
14  ?atomDescription1   sd:hasSubject      ?home .
15  ?atomDescription1   sd:hasObject       ex:Home .
16  ?atomDescription1   sd:hasProperty     rdf:type .
17  ?atomDescription2   sd:describedIn     ?semanticDescriptor2 .
18  ?atomDescription2   sd:hasSubject      ?home .
19  ?atomDescription2   sd:hasObject       ?location .
20  ?atomDescription2   sd:hasProperty     ex:hasLocation .
21  ?atomDescription3   sd:describedIn     ?semanticDescriptor3 .
22  ?atomDescription3   sd:hasSubject      ?location .
23  ?atomDescription3   sd:hasObject       "300" .
24  ?atomDescription3   sd:hasProperty     ex:hasLatitude .
25
26 }
```

This method, although directly support access control in the triple store, the disadvantage is very obvious. The triple store needs to handle triples with four times more than the number of the original triples, which will cause extra cost in both storage size and the query time.

5.4 Store semantic descriptor in multiple graphs with access control

In order to solve the issue in the previous method, we designed another solution to support access control in the triple store, which leverages the multiple graphs feature in Jena.

In this method, all the access control policies are stored in one graph, and the triples from each semantic descriptor will be stored in an individual graph, the resource ID of the

www.manaraa.com

semantic descriptor will be used as the name of the graph, and the key to search among the triple store.

The triples in the ACP graph looks very similar with the ones in the previous method, and the query message will looks like

```
1
2  SELECT DISTINCT ?pi WHERE {
3   ?accessControlRule acp:hasACOriginator ?Originator
4   FILTER(?Originator = âĂIJUser1")
5   ?accessControlRule acp:hasACOperations "DISCOVERY" .
6   ?accessControlPolicy acp:hasACPRule ?accessControlRule .
7   ?accessControlPolicy acp:appliedTo ?graph .
8   GRAPH ?graph{
9     ?home     rdf:type      ex:Home .
10    ?home    ex:hasLocation      ?location .
11    ?location    ex:hasLatitude    "300" .
12     }
13 }
```

Please note that in this method, Jena will first search in the ACP graph when handling a query message, and then go through the graphs returned by the ACP graph, this is a tree based search and will significantly reduce the searching time compared with the previous methods, which need to go through the whole triple store.

In this section, we introduced the design of three potential solutions that we designed to support access control in the triple store. In the next section, we will evaluate the performance of these methods.

5.5 Performance Evaluation

In this section, we will move on evaluate the performance of our proposed solutions. First we would like to start with the stress test of RDF triple store.

www.manaraa.com

5.5 RDF triple store limitation

In the previous research [72], the researchers trying to use RDF triplestore to store and query the data from all the domestic flights to and from Atlanta airport in one day, which is about 2.4M triples. The triple store successfully handled the storage and query, while the query time could be as long as 27 mins.

To verify the capability of Jena, we also download two large RDF databases from data.gov [120], which are city of Detroit crime data from 2009 to 2016, and city of Chicago crime data from 2001 to 2016. We store these two databases in Jena and want to test if Jena can handle the saving and querying without crash. The test result is shown in the following table.

Table 5.7: Stress test result of Jena using public crime data

	Detroit	Chicago
Row#	1,311,744	6,426,882
Col#	16	23
Disk(MB)	3,563	27,610
Triple#	20,987,904	146,125,721
Query Time(ms)	27,669	71,386

In our test, Jena successfully handled both of the databases, one with 20 million triples, the other with 146 million, without crash. Although the query time reaches minute level, this stress test gave us the confidence of using Jena in oneM2M SDP to support million level sensors.

5.5 Performance of the proposed methods

To evaluate the performance of the proposed methods, 6 test cases were designed with different number of semantic descriptors and number of triples in each semantic descriptor, as shown in Table II. To test the scalability of the proposed system, the test cases is designed with 100,000 sensors and 100,000 semantic descriptors every sensor. The experiment is set up using two Lenovo T580 laptop, with Intel Core i7-8650U CUP, 32GB DDR4 2400 MHz memory and 16 GB SSD hard-drive. The two machines are connect to the same router as

www.manaraa.com

network configuration. One laptop is configured as the standalone Jena server and another simulating all semantic descriptors and send them to Jena server using HTTP request.

Table 5.8: Test cases used in the evaluation

Test case	# of semantic descriptors	triples#/semantic descriptor
1	100,000	1
2	10,000	10
3	1,000	100
4	100	1000
5	10	10,000
6	1	100,000

In these 6 cases, the total number of triples are controlled to be 100,000. We tried to test with million level triples, but it would take several days to simulate the semantic descriptors and push them to Jena. Although Jena has been proved to be capable of handling those amount of data, the laptop we used in the experiment will crash due to the lack of memory during the generating of triples.

We evaluated the performance of the proposed methods from three aspects as query time, storage size and triple numbers, as shown in Figure 5.8, Figure 5.9, and Figure 5.10.

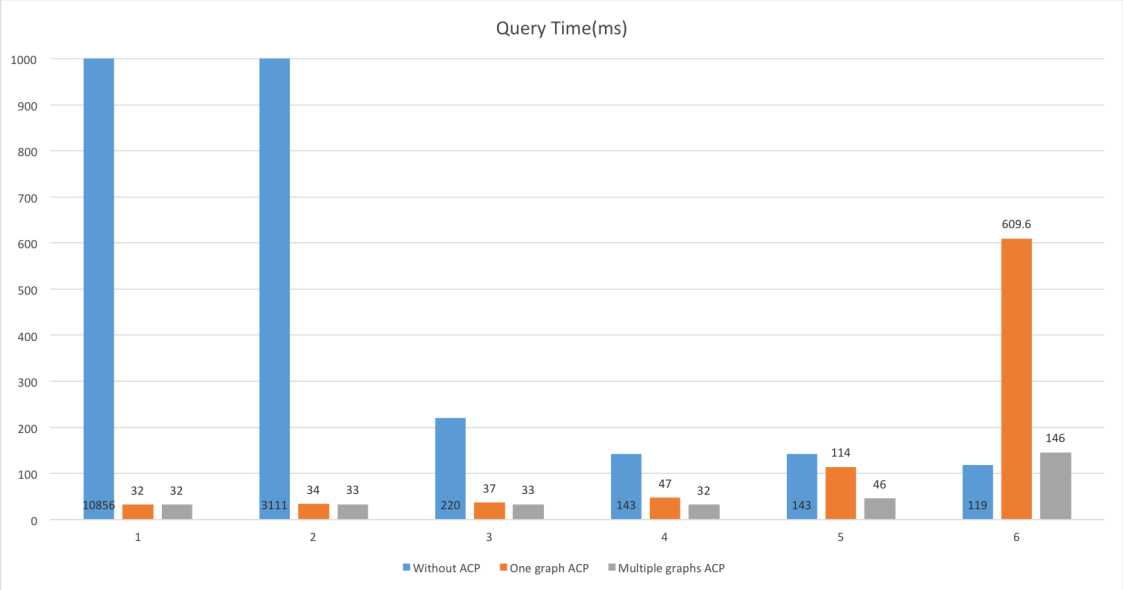

Figure 5.8: Query time of three proposed methods.

www.manaraa.com

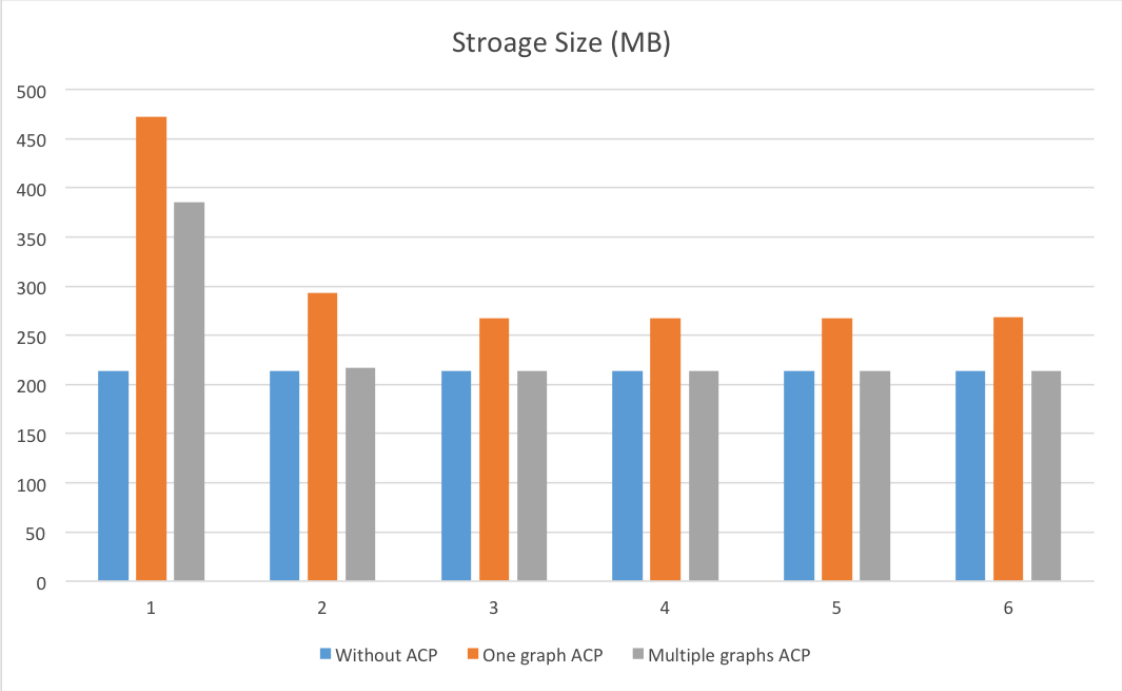

Figure 5.9: Storage size of three proposed methods..

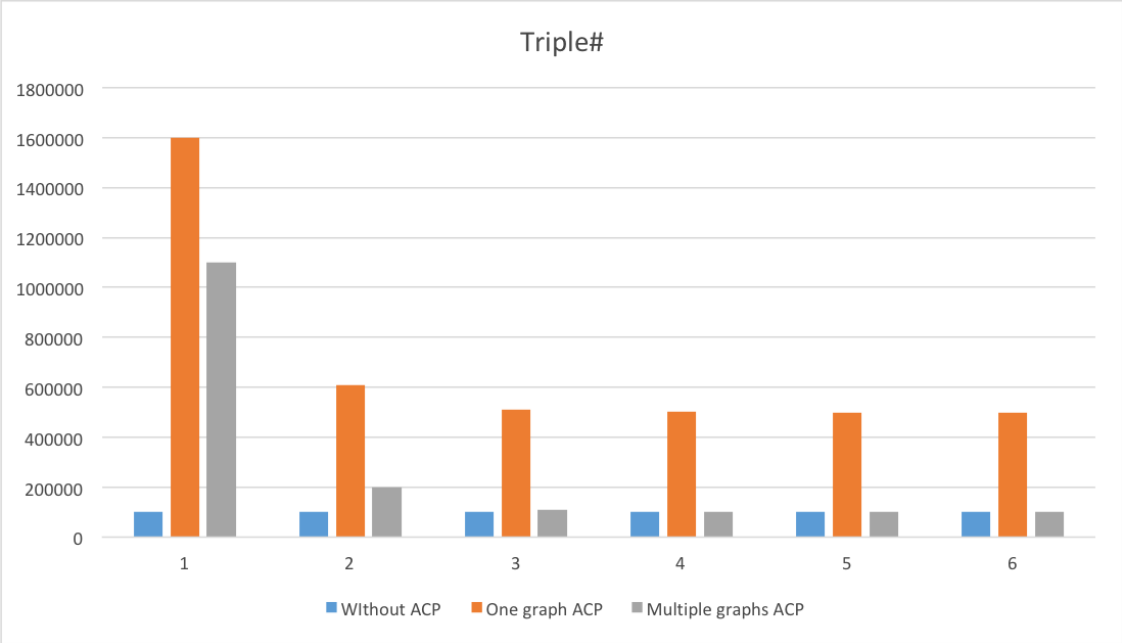

Figure 5.10: Triple number of three proposed methods.

Figure 5.8 shows the query time of the three proposed methods. Here we collected the total time from the user sent the query to the user received the result. Since the experiment is employed on a single machine, the travel time through the network is always fixed and

www.manaraa.com

measured to be 17 ms. In this case, the actual query time in Jena and oneM2M SDP could be clearly observed from Figure 5.8.

From Figure 5.8 we could tell that when the number of semantic descriptors are small, the query time will be small if the ACP is controlled by the oneM2M SPD itself. However, as the number of semantic descriptors increase, the query time of our first method becomes intolerable, since it will take to much time to search both Jena and the MongoDB then find the intersection list. The multiple graphs method is always faster compared with the other two.

From Figure 5.9 and Figure 5.10 we could tell the overhead of implementing ACP in Jena in either single and multiple graphs. Without ACP, Jena only need to store the original triples. If the ACP is stored in the single graph method, the number of triples will be increased significantly when the number of semantic descriptors are large. Reflecting to the storage size used by the disk, we can tell that the same trend from Figure 5.9.

Overall, the method which store access control policy in a separate graph and store the triples from one semantic descriptor in an individual graph only brings a little overhead in the storage size, but this method could reduce the query time significantly compared with the one that leverage the access control mechanism of oneM2M SDP. Based on this conclusion, we will add the proposed method in the future release of the oneM2M SDP.

To further test the scalability of the proposed method, we enlarge both the number of the semantic descriptors and the number of triples per descriptor to ten times of their original scale, and compared the result of both query time and storage size in the following tables.

From Table 5.9 and 5.10 we can tell that when the number of Semantic descriptors and number of triples increased to ten times of its original size, the proposed method still performs very stable with reasonable query time and storage size. In this experiment, we successfully verified the scalability of our system to support million level triples in one stand alone machine.

www.manaraa.com

Table 5.9: Compared query time (ms) for scalability test

SD \Triple#	1	10	100	1000	10000	100000
100000	146					
10000	15	46				
1000		18	33			
100			17	32		
10				18	33	
1					17	32

Table 5.10: Compared storage size (MB) for scalability test

SD \Triple#	1	10	100	1000	10000	100000
100000	217					
10000	1.49	214				
1000		1.47	214			
100			1.49	214		
10				1.28	217	
1					4.15	358

5.6 Summary

More and more IoT/M2M systems are deployed at the edge of the network and connected with each other for collaboration. Context awareness becomes a huge challenge for data exchange and communication under this background. Semantic paved the road for the knowledge-driven IoT/M2M systems. With semantic annotation, data from one IoT/M2M system could be easily understood and used by others. However, provide semantic service to an IoT/M2M system is a challenging task since the original database used in the system may not be capable of semantic annotation. In this chapter, we took the challenge and proposed a design to support semantic service to oneM2M SDP, which is a mature M2M system with its own database. Moreover, we also discussed the approaches of support secure mechanism such as access control directly in within the semantic information to boost the performance of the system. Our experiment shows that the proposed method could effectively and securely provide semantic service to the IoT/M2M system.

www.manaraa.com

Chapter 6: Conclusions

The proliferation of Internet of Things and the success of rich cloud services have pushed the horizon of a new computing paradigm, Edge computing, which calls for processing the data at the edge of the network. Applications such as cloud offloading, smart home, and smart city are idea area for Edge computing to achieve better performance than cloud computing. Edge computing has the potential to address the concerns of response time requirement, battery life constraint, bandwidth cost saving, as well as data safety and privacy.

However, there are still some challenges for applying Edge computing in our daily life. The missing of the specialized operating system for Edge computing is holding back the flourish of Edge computing applications.

We illustrated our vision and understanding of Edge computing. We also give several case studies to further explain how Edge computing could be adapted to the current computing paradigm. Then we summarize the challenges in detail and bring forward some potential solutions and opportunities worth further research, including programmability, naming, data abstraction, service management, privacy and security and optimization metrics. we also introduced the design of SOFIE and talk about how we want to address these challenges using SOFIE.

To address the challenges for Edge computing systems and applications in these aspects, we have planned a series of empirical and theoretical research. We proposed SOFIE: Smart Operating System For Internet Of Everything. SOFIE is the operating system specialized for Edge computing running on the Edge gateway. SOFIE could establish and maintain a reliable connection between cloud and Edge device to handle the data transportation between gateway and Edge devices; to provide service management and data management for Edge applications; to protect data privacy and security for Edge users; to guarantee the wellness of the Edge devices. Moreover, SOFIE also provide a naming mechanism to connect Edge device more efficiently. To solve the component selection problem in Edge

www.manaraa.com

computing paradigm, SOFIE also include our previous work, SURF, as a model to optimize the performance of the system.

In the design and development of Edge computing applications, practitioners usually face several performance requirements from service providers, end users, and/or hardware constrains. Moreover, there could be a large number of available choices for different components in the application. To help the practitioners to select the optimal component combinations that can meet the performance requirements and reduce cost as much as possible at the same time, we proposed performance evaluation metrics and a methodology for using performance vector. We tested our methodology through a case study, and the result showed that our methodology is very efficient in finding the optimal component combination. Moreover, we discussed two interesting findings we have observed in the case study, one is the effect of decision making algorithm on data quality, and the other is different operation periods for battery.

As more and more IoT/M2M systems are deployed at the edge of the network and connected with each other for collaboration. Context awareness becomes a huge challenge for data exchange and communication under this background. Semantic paved the road for the knowledge-driven IoT/M2M systems. With semantic annotation, data from one IoT/M2M system could be easily understood and used by others. However, provide semantic service to an IoT/M2M system is a challenging task since the original database used in the system may not be capable of semantic annotation. In this chapter, we took the challenge and proposed a design to support semantic service to oneM2M SDP, which is a mature M2M system with its own database. Moreover, we also discussed the approaches of support secure mechanism such as access control directly in within the semantic information to boost the performance of the system. Our experiment shows that the proposed method could effectively and securely provide semantic service to the IoT/M2M system.

www.manaraa.com

Chapter 7: Future Work

In the future, I plan to extend my research in the following four areas.

7.1 Connected Health

Connected health can be defined as the use of Internet, sensing, communications and intelligent techniques in support of health related applications, systems and engineering. Connected health brings together multidisciplinary technologies to provide preventive or remote treatments by utilizing digital health information structure such as body sensor networks and intelligent techniques such as from Data to Knowledge to Decisions, while at the same time connecting patient and caregivers seamlessly in the loop of the healthcare ecosystem. Future Connected health will be realized by providing rich medical information to each individual through replacing infrequent, clinic-based measurements with unobtrusive, continuous sensing, monitoring and assessment. Connected health technologies will enable preventive health and personalized medicine and may significantly reduce healthcare costs.

I have learned some experience from the design and development of several connected health systems, and plan to collaborate with health care providers, patients, insurance, pharmaceutical and medical technology companies to further reduce the healthcare cost and improve the healthcare efficiency.

7.2 Data Quality Management in Edge Computing

In an Edge Computing system, data driven applications provide information to users in order to make decisions. It is important that those decisions are based on data that is accurate, complete, and in or close to real-time. Data accuracy, completeness, and delay play a critical role, particularly in a smart home environment. In order to detect sensing error, we think in Edge Computing systems data quality could be evaluated by two aspects: history pattern and reference data.

In an Edge Computing system, data could easily fall into a certain pattern due to the periodical user behavior. To provide better service to applications and devices, new

www.manaraa.com

data mining and machine learning algorithm should be introduced to data quality detection model. This model should automatically detect abnormal data pattern from the historical data record, and further analyze the reason for the abnormal pattern, which could be user behavior changing, device failure, communication interfacing, or attack from outside.

In the future, I plan to design a data quality management infrastructure for Edge Computing. This infrastructure could benefit Edge Computing systems from two aspects: error/low-quality data detection; and provide suitable data to the application based on quality requirement.

7.3 Knowledge Abstraction in Edge Computing

Various applications can run on an Edge Computing system consuming data or providing service. We envision that human involvement in Edge Computing should be minimized and the Edge node should consume/process all the data and interact with users in a proactive fashion. In this case, data should be preprocessed at the gateway level, such as noise/low-quality removal, event detection, and privacy protection, and so on. Processed data will be sent for future service providing.

My internship work at InterDigital includes using RDF triple store to annotate the semantic information of IoT data, and I plan to expand this work in the future to abstract valuable information from the raw data collected from Edge nodes, and provide the knowledge to other applications and systems.

7.4 Security and Privacy in Edge Computing

Limited resources and computational capabilities, as well as multiple communication protocols makes it very difficult to protect Edge Computing systems from attackers. WhatâĂŹs more, security and privacy problems at Edge such as smart home can always raise peopleâĂŹs concerns. Resource constrains and heterogeneous communication brings new challenges to security and privacy in Edge Computing.

www.manaraa.com

In the future, I plan to work on Edge Computing security and privacy protection from two aspects. One is usage privacy protection between various applications running on the Edge of the network. The other one is secure data storage and computation in different Edge Computing layers.

www.manaraa.com

REFERENCES

[1] Baby monitor hacker still terrorizing babies and their parents. [Online]. Available: http://www.forbes.com/sites/kashmirhill/2014/04/29/baby-monitor-hacker-still-terrorizing-babies-and-their-parents/#5b91ff7717e2

[2] "Boeing 787s to create half a terabyte of data per flight, says virgin atlantic," https://datafloq.com/read/self-driving-cars-create-2-petabytes-data-annually/172.

[3] "Data never sleeps 2.0," https://www.domo.com/blog/2014/04/data-never-sleeps-2-0/.

[4] "Official: Two men sought as possible suspects in boston bombing," http://www.cnn.com/2013/04/17/us/boston-blasts/.

[5] "Open mhealth platform," http://www.openmhealth.org/.

[6] "Self-driving cars will create 2 petabytes of data, what are the big data opportunities for the car industry?" http://www.computerworlduk.com/news/data/.

[7] "The smart home industry: Trends and innovations," http://dishwashersguide.com/knowledgebase/smart-home-trends-innovations/.

[8] "Software-defined networking: How it affects network security." [Online]. Available: http://www.techrepublic.com/blog/it-security/software-defined-networking-how-it-affects-network-security/

[9] "Wi-fi networks security." [Online]. Available: http://graphs.net/wifi-stats.html

[10] "Wifi network security statistics/graph," http://graphs.net/wifi-stats.html/.

[11] "Your hackable house." [Online]. Available: http://money.cnn.com/interactive/technology/hackable-house/

[12] "Cisco global cloud index: Forecast and methodology, 2014-2019 white paper," 2014.

[13] "IDC futurescape: Worldwide internet of things 2016 predictions," 2015.

[14] "Openfog architecture overview," *OpenFog Consortium Architecture Working Group*, 2016.

www.manaraa.com

[15] G. Ananthanarayanan, P. Bahl, P. Bodik, K. Chintalapudi, M. Philipose, L. Ravindranath, and S. Sinha, "Real-time video analytics: The killer app for edge computing," *Computer*, vol. 50, no. 10, pp. 58–67, 2017.

[16] 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," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010. [Online]. Available: http://doi.acm.org/10.1145/1721654.1721672

[17] I. Armenti, P. Asare, J. Su, and J. Lach, "A methodology for developing quality of information metrics for body sensor design," *Wireless Health 2012*, 2012.

[18] K. Ashton, "That âĂŸinternet of thingsâĂŹ thing," *RFiD Journal*, vol. 22, no. 7, pp. 97–114, 2009.

[19] R. Balan, J. Flinn, M. Satyanarayanan, S. Sinnamohideen, and H.-I. Yang, "The case for cyber foraging," in *Proceedings of the 10th workshop on ACM SIGOPS European workshop*. ACM, 2002, pp. 87–92.

[20] K. Balasubramanian and A. Cellatoglu, "Improvements in home automation strategies for designing apparatus for efficient smart home," *IEEE Transactions on Consumer Electronics*, vol. 54, no. 4, pp. 1681–1687, 2008.

[21] S. Bechhofer, "Owl: Web ontology language," in *Encyclopedia of database systems*. Springer, 2009, pp. 2008–2009.

[22] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. ACM, 2012, pp. 13–16.

[23] ——, "Fog computing and its role in the internet of things," in *Proceedings of the Mobile Cloud Computing*. ACM, 2012, pp. 13–16.

[24] J. S. Brantley, A. T. Barth, and J. Lach, "Optimizing battery lifetime-fidelity trade-offs in bsns using personal activity profiles," in *Proceedings of the 7th International Conference on Body Area Networks*, ser. BodyNets '12.

www.manaraa.com

[25] D. Brickley, "Rdf vocabulary description language 1.0: Rdf schema," *http://www. w3. org/TR/rdf-schema/*, 2004.

[26] J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava, "Participatory sensing," 2006.

[27] M. Cannataro and D. Talia, "Semantics and knowledge grids: building the next-generation grid," *IEEE Intelligent Systems*, vol. 19, no. 1, pp. 56–63, 2004.

[28] J. Cao, L. Ren, Z. Yu, and W. Shi, "A framework for component selection in collaborative sensing application development," in *10th IEEE Conference on Collaborative Computing: Networking, Applications and Worksharing*. IEEE.

[29] S. Chakrabarti and A. Mishra, "Qos issues in ad hoc wireless networks," *Communications Magazine, IEEE*, vol. 39, no. 2, pp. 142–148, 2001.

[30] P. P.-S. Chen, "The entity-relationship modelâĂŤtoward a unified view of data," in *Readings in artificial intelligence and databases*. Elsevier, 1988, pp. 98–111.

[31] S. Chen, J. S. Brantley, T. Kim, and J. Lach, "Characterizing and minimizing synchronization and calibration errors in inertial body sensor networks," in *Proceedings of the Fifth International Conference on Body Area Networks*, ser. BodyNets '10. New York, NY, USA: ACM, 2010, pp. 138–144.

[32] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: Elastic execution between mobile device and cloud," in *Proceedings of the Sixth Conference on Computer Systems*, ser. EuroSys '11. New York, NY, USA: ACM, 2011, pp. 301–314. [Online]. Available: http://doi.acm.org/10.1145/1966445.1966473

[33] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: making smartphones last longer with code offload," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 2010, pp. 49–62.

[34] F. DaCosta, *Rethinking the Internet of Things: a scalable approach to connecting everything*. Apress, 2013.

www.manaraa.com

[35] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[36] Z.-Y. Demetrios, "A glance at quality of services in mobile ad-hoc networks," in *Seminar in Mobile Ad Hoc Networks*, 2001.

[37] N. Ding, D. Wagner, X. Chen, A. Pathak, Y. C. Hu, and A. Rice, "Characterizing and modeling the impact of wireless signal strength on smartphone battery drain," *SIGMETRICS Perform. Eval. Rev.*, vol. 41, no. 1, pp. 29–40, Jun. 2013. [Online]. Available: http://doi.acm.org/10.1145/2494232.2466586

[38] D. Evans, "The internet of things: How the next evolution of the internet is changing everything," *CISCO white paper*, vol. 1, p. 14, 2011.

[39] A. Fahim, A. Mtibaa, and K. A. Harras, "Making the case for computational offloading in mobile device clouds," in *Proceedings of the Conference on Mobile Computing and Networking*. ACM, 2013, pp. 203–205.

[40] J. Feld, "PROFINET-scalable factory communication for all applications," in *Factory Communication Systems, 2004. Proceedings. 2004 IEEE International Workshop on*. IEEE, 2004, pp. 33–38.

[41] E. Fernandes, J. Jung, and A. Prakash, "Security analysis of emerging smart home applications," 2016.

[42] N. Fernando, S. W. Loke, and W. Rahayu, "Computing with nearby mobile devices: a work sharing algorithm for mobile edge-clouds," *IEEE Transaction on Cloud Computing*, 2016.

[43] M. Finnegan, "Boeing 787s to create half a terabyte of data per flight, says virgin atlantic," *Computerworld UK*, vol. 6, 2013.

[44] I. Foster, "The globus toolkit for grid computing," in *ccgrid*. IEEE, 2001, p. 2.

[45] I. Foster and C. Kesselman, *The Grid 2: Blueprint for a new computing infrastructure*. Elsevier, 2003.

www.manaraa.com

[46] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke, "The physiology of the grid," *Grid computing: making the global infrastructure a reality*, pp. 217–249, 2003.

[47] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations," *The International Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 200–222, 2001.

[48] P. J. Frantz and G. O. Thompson, "Vlan frame format," Sep. 28 1999, uS Patent 5,959,990.

[49] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Annual Cryptology Conference*. Springer, 2010, pp. 465–482.

[50] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in *ACM SIGOPS operating systems review*, vol. 37, no. 5. ACM, 2003, pp. 29–43.

[51] S. Goyal and J. Carter, "A lightweight secure cyber foraging infrastructure for resource-constrained devices," in *Mobile Computing Systems and Applications, 2004. WMCSA 2004. Sixth IEEE Workshop on*. IEEE, 2004, pp. 186–195.

[52] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM computer communication review*, vol. 39, no. 1, pp. 68–73, 2008.

[53] ——, "The cost of a cloud: Research problems in data center networks," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 68–73, Dec. 2008. [Online]. Available: http://doi.acm.org/10.1145/1496091.1496103

[54] T. Guan, E. Zaluska, and D. De Roure, "A grid service infrastructure for mobile devices," in *Semantics, Knowledge and Grid, 2005. SKG'05. First International Conference on*. IEEE, 2005, pp. 42–42.

[55] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

www.manaraa.com

[56] C. Gui and P. Mohapatra, "Power conservation and quality of surveillance in target tracking sensor networks," in *Proceedings of the 10th annual international conference on Mobile computing and networking*. ACM, 2004, pp. 129–143.

[57] V. C. Gungor, D. Sahin, T. Kocak, S. Ergut, C. Buccella, C. Cecati, and G. P. Hancke, "Smart grid and smart homes: key players and pilot projects," *IEEE Industrial Electronics Magazine*, vol. 6, no. 4, pp. 18–34, 2012.

[58] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan, "Towards wearable cognitive assistance," in *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*. ACM, 2014, pp. 68–81.

[59] K. Habak, M. Ammar, K. A. Harras, and E. Zegura, "Femto clouds: Leveraging mobile devices to provide cloud service at the edge," in *Proceedings of the International Conference on Cloud Computing*. IEEE, 2015, pp. 9–16.

[60] J. Han, C.-S. Choi, and I. Lee, "More efficient home energy management system based on zigbee communication and infrared remote controls," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 1, pp. 85–89, 2011.

[61] M. Hanson, H. Powell, A. Barth, K. Ringgenberg, B. Calhoun, J. Aylor, and J. Lach, "Body area sensor networks: Challenges and opportunities," *Computer*, vol. 42, no. 1, pp. 58–65, 2009.

[62] M. A. Hanson, H. C. Powell, Jr., A. T. Barth, and J. Lach, "Application-focused energy-fidelity scalability for wireless motion-based health assessment," *ACM Trans. Embed. Comput. Syst.*, vol. 11, no. S2, pp. 50:1–50:21, Aug. 2012.

[63] G. C. Hunt, M. L. Scott *et al.*, "The coign automatic distributed partitioning system," in *OSDI*, vol. 99, 1999, pp. 187–200.

[64] ——, "The coign automatic distributed partitioning system," in *OSDI*, vol. 99, 1999, pp. 187–200.

[65] C. G. C. Index, "Forecast and methodology, 2015-2020 white paper," 2016.

www.manaraa.com

[66] R. Istepanian, E. Jovanov, and Y. Zhang, "Guest editorial introduction to the special section on m-health: Beyond seamless mobility and global wireless health-care connectivity," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 8, no. 4, pp. 405–414, 2004.

[67] R. Iyer and L. Kleinrock, "Qos control for sensor networks," in *Communications, 2003. ICC'03. IEEE International Conference on*, vol. 1. IEEE, 2003, pp. 517–521.

[68] K. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. J. Wasserman, and N. Wright, "Performance analysis of high performance computing applications on the amazon web services cloud," in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, Nov 2010, pp. 159–168.

[69] S. Jain, V. Nguyen, M. Gruteser, and P. Bahl, "Panoptes: servicing multiple applications simultaneously using steerable cameras," in *Proceedings of the International Conference on Information Processing in Sensor Networks*, 2017, pp. 119–130.

[70] A. Jena, "Apache jena," *jena. apache. org [Online]. Available: http://jena. apache. org [Accessed: Mar. 20, 2018]*, p. 14, 2018.

[71] S. H. Ju, Y. H. Lim, M. S. Choi, J.-M. Baek, and S.-Y. Lee, "An efficient home energy management system based on automatic meter reading," in *Power Line Communications and Its Applications (ISPLC), 2011 IEEE International Symposium on*. IEEE, 2011, pp. 479–484.

[72] R. M. Keller, S. Ranjan, M. Y. Wei, and M. M. Eshow, "Semantic representation and scale-up of integrated air traffic management data," in *Proceedings of the International Workshop on Semantic Big Data*. ACM, 2016, p. 4.

[73] G. Klyne and J. J. Carroll, "Resource description framework (rdf): Concepts and abstract syntax," 2006.

[74] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 945–953.

www.manaraa.com

[75] E. Kovacs, M. Bauer, J. Kim, J. Yun, F. Le Gall, and M. Zhao, "Standards-based worldwide semantic interoperability for iot," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 40–46, 2016.

[76] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, no. 4, pp. 51–56, 2010.

[77] S. Kumar, W. Nilsen, M. Pavel, and M. Srivastava, "Mobile health: Revolutionizing healthcare through transdisciplinary research," *Computer*, vol. 46, no. 1, pp. 28–35, 2013.

[78] J. Lertlakkhanakul, J. W. Choi, and M. Y. Kim, "Building data model and simulation platform for spatial interaction management in smart home," *Automation in Construction*, vol. 17, no. 8, pp. 948–957, 2008.

[79] A. Li, X. Yang, S. Kandula, and M. Zhang, "Cloudcmp: Comparing public cloud providers," in *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '10. New York, NY, USA: ACM, 2010, pp. 1–14. [Online]. Available: http://doi.acm.org/10.1145/1879141.1879143

[80] H. Li, D. Seed, B. Flynn, C. Mladin, and R. Di Girolamo, "Enabling semantics in an m2m/iot service delivery platform," in *Semantic Computing (ICSC), 2016 IEEE Tenth International Conference on*. IEEE, 2016, pp. 206–213.

[81] X.-Y. Li, P.-J. Wan, and O. Frieder, "Coverage in wireless ad hoc sensor networks," *Computers, IEEE Transactions on*, vol. 52, no. 6, pp. 753–763, 2003.

[82] B. Lo, S. Thiemjarus, R. King, and G.-Z. Yang, "Body sensor network-a wireless sensor platform for pervasive healthcare monitoring," in *The 3rd International Conference on Pervasive Computing*, vol. 13, 2005, pp. 77–80.

[83] R. Lu, X. Liang, X. Li, X. Lin, and X. Shen, "Eppa: An efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1621–1631, 2012.

www.manaraa.com

[84] J. P. Lynch, K. H. Law, A. S. Kiremidjian, T. W. Kenny, E. Carryer, and A. Partridge, "The design of a wireless sensing unit for structural health monitoring," in *Proceedings of the 3rd International Workshop on Structural Health Monitoring*, 2001, pp. 12–14.

[85] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak, "Exposure in wireless ad-hoc sensor networks," in *Proceedings of the 7th annual international conference on Mobile computing and networking*. ACM, 2001, pp. 139–150.

[86] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing*, ser. HotCloud'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 4–4. [Online]. Available: http://dl.acm.org/citation.cfm?id=1863103.1863107

[87] A. Milenkovic, C. Otto, and E. Jovanov, "Wireless sensor networks for personal health monitoring: Issues and an implementation," *Computer Communications*, vol. 29, no. 13-14, pp. 2521 – 2533, 2006.

[88] S. Möller, J. Krebber, A. Raake, P. Smeele, M. Rajman, M. Melichar, V. Pallotta, G. Tsakou, B. Kladis, A. Vovos *et al.*, "Inspire: Evaluation of a smart-home system for infotainment management and device control," *arXiv preprint cs/0410063*, 2004.

[89] A. Natrajan, A. Nguyen-Tuong, M. A. Humphrey, M. Herrick, B. P. Clarke, and A. S. Grimshaw, "The legion grid portal," *Concurrency and Computation: Practice and Experience*, vol. 14, no. 13-15, pp. 1365–1394, 2002.

[90] T. Oluwafemi, T. Kohno, S. Gupta, and S. Patel, "Experimental security analyses of non-networked compact fluorescent lamps: A case study of home automation security," in *Proceedings of the LASER 2013 (LASER 2013)*, 2013, pp. 13–24.

[91] C. Otto, A. Milenkovic, C. Sanders, and E. Jovanov, "System architecture of a wireless body area sensor network for ubiquitous health monitoring," *Journal of Mobile Multimedia*, vol. 1, no. 4, pp. 307–326, 2006.

www.manaraa.com

[92] A. Palavalli, D. Karri, and S. Pasupuleti, "Semantic internet of things," in *Semantic Computing (ICSC), 2016 IEEE Tenth International Conference on*. IEEE, 2016, pp. 91–95.

[93] E. Prud, A. Seaborne *et al.*, "Sparql query language for rdf," 2006.

[94] B. Qela and H. T. Mouftah, "Observe, learn, and adapt (ola)âĂŤan algorithm for energy management in smart homes using wireless sensors and artificial intelligence," *IEEE Transactions on Smart Grid*, vol. 3, no. 4, pp. 2262–2272, 2012.

[95] D. Raychaudhuri, K. Nagaraja, and A. Venkataramani, "Mobilityfirst: a robust and trustworthy mobility-centric architecture for the future internet," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 16, no. 3, pp. 2–13, 2012.

[96] D. Raychev, Y. Li, and W. Shi, "The seventh cell of a six-cell battery," in *Proceedings of the third Workshop on Energy-Efficient Design (WEED 2011)*, 2011.

[97] L. Ren, Q. Zhang, and W. Shi, "Low-power fall detection in home-based environments," in *Proceedings of the 2nd ACM international workshop on Pervasive Wireless Healthcare*. ACM, 2012, pp. 39–44.

[98] H. Rowaihy, S. Eswaran, M. Johnson, D. Verma, A. Bar-Noy, T. Brown, and T. La Porta, "A survey of sensor selection schemes in wireless sensor networks," in *Defense and Security Symposium*. International Society for Optics and Photonics, 2007, pp. 65 621A–65 621A.

[99] A. Rudenko, P. Reiher, G. J. Popek, and G. H. Kuenning, "Saving portable computer battery power through remote process execution," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 2, no. 1, pp. 19–26, 1998.

[100] N. Sarma and S. Nandi, "Qos support in mobile ad hoc networks," in *Wireless and Optical Communications Networks, 2006 IFIP International Conference on*, 2006.

[101] M. Satyanarayanan, "Mobile computing: The next decade," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 15, no. 2, pp. 2–10, Aug. 2011. [Online]. Available: http://doi.acm.org/10.1145/2016598.2016600

www.manaraa.com

[102] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *Pervasive Computing, IEEE*, vol. 8, no. 4, pp. 14–23, 2009.

[103] ——, "The case for vm-based cloudlets in mobile computing," *Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[104] E. Saurez, K. Hong, D. Lillethun, U. Ramachandran, and B. Ottenwälder, "Incremental deployment and migration of geo-distributed situation awareness applications in the fog," in *Proceedings of the International Conference on Distributed and Event-based Systems*. ACM, 2016, pp. 258–269.

[105] K. Scarfone and J. Padgette, "Guide to bluetooth security," *NIST Special Publication*, vol. 800, p. 121, 2008.

[106] K. Sha and W. Shi, "Consistency-driven data quality management of networked sensor systems," *Journal of Parallel and Distributed Computing*, vol. 68, no. 9, pp. 1207 – 1221, 2008.

[107] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[108] ——, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2016.

[109] ——, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[110] Y. Shi, S. Abhilash, and K. Hwang, "Cloudlet mesh for securing mobile clouds from intrusions and network attacks," in *Mobile Cloud Computing, Services, and Engineering (MobileCloud), 2015 3rd IEEE International Conference on*. IEEE, 2015, pp. 109–118.

[111] S. Shin and G. Gu, "Cloudwatcher: Network security monitoring using openflow in dynamic cloud networks (or: How to provide security monitoring as a service in clouds?),"

www.manaraa.com

in *2012 20th IEEE international conference on network protocols (ICNP)*. IEEE, 2012, pp. 1–6.

[112] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*. IEEE, 2010, pp. 1–10.

[113] P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, and M. Satyanarayanan, "Scalable crowd-sourcing of video from mobile devices," in *Proceedings of the International Conference on Mobile systems, Applications, and Services*. ACM, 2013, pp. 139–152.

[114] J. Singh, T. Pasquier, J. Bacon, H. Ko, and D. Eyers, "Twenty security considerations for cloud-supported internet of things," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 269–284, 2016.

[115] Y.-Y. Su and J. Flinn, "Slingshot: Deploying stateful services in wireless hotspots," in *Proceedings of the 3rd international conference on Mobile systems, applications, and services*. ACM, 2005, pp. 79–92.

[116] H. Sundmaeker, P. Guillemin, P. Friess, and S. Woelfflé, "Vision and challenges for realising the internet of things," 2010.

[117] J. Swetina, G. Lu, P. Jacobs, F. Ennesser, and J. Song, "Toward a standardized common m2m service layer platform: Introduction to onem2m," *IEEE Wireless Communications*, vol. 21, no. 3, pp. 20–26, 2014.

[118] D. Thain, T. Tannenbaum, and M. Livny, "Distributed computing in practice: the condor experience," *Concurrency and computation: practice and experience*, vol. 17, no. 2-4, pp. 323–356, 2005.

[119] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman, "A taxonomy of wireless microsensor network models," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 6, no. 2, pp. 28–36, 2002.

[120] USDataGov, "The home of the u.s. governmentâĂŹs open data," *https://www.data.gov/safety*, 2018.

www.manaraa.com

[121] M. Van Rijmenam, "Self-driving cars will create 2 petabytes of data, what are the big data opportunities for the car industry," 2017.

[122] G. Virone, A. Wood, L. Selavo, Q. Cao, L. Fang, T. Doan, Z. He, and J. Stankovic, "An advanced wireless sensor network for health monitoring," in *Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare (D2H2)*, 2006, pp. 2–4.

[123] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *INFOCOM, 2010 Proceedings IEEE*. Ieee, 2010, pp. 1–9.

[124] J. Wang, B. Amos, A. Das, P. Pillai, N. Sadeh, and M. Satyanarayanan, "A scalable and privacy-aware iot service for live video analytics," in *Proceedings of the Multimedia Systems Conference*. ACM, 2017, pp. 38–49.

[125] Y. Wang, J. P. Lynch, and K. H. Law, "A wireless structural health monitoring system with multithreaded sensing devices: design and validation," *Structure and Infrastructure Engineering*, vol. 3, no. 2, pp. 103–120, 2007.

[126] W. H. Wu, A. A. Bui, M. A. Batalin, L. K. Au, J. D. Binney, and W. J. Kaiser, "Medic: Medical embedded device for individualized care," *Artificial Intelligence in Medicine*, vol. 42, no. 2, pp. 137–152, 2008.

[127] Z. Yan, V. Subbaraju, D. Chakraborty, A. Misra, and K. Aberer, "Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach," in *Wearable Computers (ISWC), 2012 16th International Symposium on*. IEEE, 2012, pp. 17–24.

[128] G.-Z. Yang and M. Yacoub, "Body sensor networks," 2006.

[129] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog computing: Platform and applications," in *Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop on*. IEEE, 2015, pp. 73–78.

www.manaraa.com

[130] S. Yi, Z. Qin, and Q. Li, "Security and privacy issues of fog computing: A survey," in *International Conference on Wireless Algorithms, Systems, and Applications*. Springer, 2015, pp. 685–695.

[131] M. A. Yigitel, O. D. Incel, and C. Ersoy, "Qos-aware mac protocols for wireless sensor networks: A survey," *Computer Networks*, vol. 55, no. 8, pp. 1982–2004, 2011.

[132] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *Infocom, 2010 proceedings IEEE*. Ieee, 2010, pp. 1–9.

[133] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: cluster computing with working sets," in *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, vol. 10, 2010, p. 10.

[134] H. Zhang, G. Ananthanarayanan, P. Bodik, M. Philipose, P. Bahl, and M. J. Freedman, "Live video analytics at scale with approximation and delay-tolerance," in *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation*, 2017, pp. 377–392.

[135] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, D. Massey, C. Papadopoulos *et al.*, "Named data networking (ndn) project," *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, 2010.

[136] Q. Zhang, L. Ren, and W. Shi, "Honey: A multimodality fall detection and telecare system," *Telemedicine and e-Health*, 2013.

[137] T. Zhang, A. Chowdhery, P. V. Bahl, K. Jamieson, and S. Banerjee, "The design and implementation of a wireless video surveillance system," in *Proceedings of the International Conference on Mobile Computing and Networking*. ACM, 2015, pp. 426–438.

[138] Y. Zigel, A. Cohen, and A. Katz, "The weighted diagnostic distortion (wdd) measure for ecg signal compression," *Biomedical Engineering, IEEE Transactions on*, vol. 47, no. 11, pp. 1422–1430, 2000.

www.manaraa.com

ABSTRACT

SOFIE: SMART OPERATING SYSTEM FOR INTERNET OF EVERYTHING

by

JIE CAO

December 2018

Advisor: Dr. Weisong Shi

Major: Computer Science

Degree: Doctor of Philosophy

The proliferation of Internet of Things and the success of rich cloud services have pushed the horizon of a new computing paradigm, Edge computing, which calls for processing the data at the edge of the network. Applications such as cloud offloading, smart home, and smart city are idea area for Edge computing to achieve better performance than cloud computing. Edge computing has the potential to address the concerns of response time requirement, battery life constraint, bandwidth cost saving, as well as data safety and privacy.

However, there are still some challenges for applying Edge computing in our daily life. The missing of the specialized operating system for Edge computing is holding back the flourish of Edge computing applications. Service management, device management, component selection as well as data privacy and security is also not well supported yet in the current computing structure.

To address the challenges for Edge computing systems and applications in these aspects, we have planned a series of empirical and theoretical research. We propose SOFIE: Smart Operating System For Internet Of Everything. SOFIE is the operating system specialized for Edge computing running on the Edge gateway. SOFIE could establish and maintain a reliable connection between cloud and Edge device to handle the data transportation between gateway and Edge devices; to provide service management and data management for Edge applications; to protect data privacy and security for Edge users; to guarantee the wellness of the Edge devices. Moreover, SOFIE also provide a naming mechanism to connect

www.manaraa.com

Edge device more efficiently. To solve the component selection problem in Edge computing paradigm, SOFIE also include our previous work, SURF, as a model to optimize the performance of the system. Finally, we deployed the design of SOFIE on an IoT/M2M system and support semantics with access control.

www.manaraa.com

AUTOBIOGRAPHICAL STATEMENT

Jie Cao is a Ph.D. candidate in the Department of Computer Science at Wayne State University. He joined the Ph.D. program in Sep 2012. He received his Masters degree in Computer Science at Wayne State University in Sep 2015 and received his Bachelor degree in Computer Science at XiDian University in Jul 2011. His research area is connected health and Edge computing. Specially, his recent work focuses on data management of IoT systems, including data quality, semantic annotation, data privacy and security. His research has been published in prestigious venues, including IEEE Internet of Things Journal and ICDCS, winning HealthCom Best Student Paper Award and an IEEE Best Paper Award. He has also served as a peer reviewer for journals and conferences.

www.manaraa.com