Fog Computing for Sustainable Smart Cities: A Survey

CHARITH PERERA, The Open University
YONGRUI QIN, University of Huddersfield
JULIO C. ESTRELLA, University of Sao Paulo
STEPHAN REIFF-MARGANIEC, University of Leicester
ATHANASIOS V. VASILAKOS, Lulea University of Technology

The Internet of Things (IoT) aims to connect billions of smart objects to the Internet, which can bring a promising future to smart cities. These objects are expected to generate large amounts of data and send the data to the cloud for further processing, specially for knowledge discovery, in order that appropriate actions can be taken. However, in reality sensing all possible data items captured by a smart object and then sending the complete captured data to the cloud is less useful. Further, such an approach would also lead to resource wastage (e.g. network, storage, etc.). The Fog (Edge) computing paradigm has been proposed to counterpart the weakness by pushing processes of knowledge discovery using data analytics to the edges. However, edge devices have limited computational capabilities. Due to inherited strengths and weaknesses, neither Cloud computing nor Fog computing paradigm addresses these challenges alone. Therefore, both paradigms need to work together in order to build an sustainable IoT infrastructure for smart cities. In this paper, we review existing approaches that have been proposed to tackle the challenges in the Fog computing domain. Specifically, we describe several inspiring use case scenarios of Fog computing, identify ten key characteristics and common features of Fog computing, and compare more than 30 existing research efforts in this domain. Based on our review, we further identify several major functionalities that ideal Fog computing platforms should support and a number of open challenges towards implementing them, so as to shed light on future research directions on realizing Fog computing for building sustainable smart cities.

CCS Concepts:•**Networks** → **Cloud computing;**•**Computing methodologies** → *Distributed computing methodologies;*•**Computer systems organization** → *Embedded and cyber-physical systems;*•**Hardware** → *Sensor applications and deployments;*

Additional Key Words and Phrases: Internet of Things, Fog Computing, Sustainability, Smart Cities

1. INTRODUCTION

Over the last few years, the Internet of Things (IoT) has become a popular term that many different communities interpret in many different ways. The term IoT has been vaguely described in the literature and has very faded boundaries. From the functional point of view IoT is defined as *"The Internet of Things allows people and things to be connected Anytime, Anyplace, with Anything and Anyone, ideally using Any path/network and Any service"* [European Commission 2008]. Most of the objects (from living room couches to street lights to parking slots) around us are expected to be embedded with sensors. These sensors will have a wide range of sensing capabilities [Perera et al. 2015a]. Then, the collected sensing data needs to be analyzed to derive knowledge in order to assist and accelerate decision making processes [Preden et al. 2015].

Data is the primary commodity in the data driven economy [RCUK Digital Economy HAT Project 2014]. Surprisingly, large volumes of data can provide much more de-

tailed insights into ourselves and our behaviors. Therefore, over the recent years *Big Data* [Zaslavsky et al. 2012] has become a buzz word in both industry and academia. Having said that, *Big Data* is not a new concept or idea. In the early days, *Big Data* was produced by large tech companies such as Google, Yahoo, Microsoft, and large scientific research institution such as European Organization for Nuclear Research (CERN). Due to advances in technology and economy of scale, computational hardware costs have significantly reduced. As a result, sensors are being integrated into all kinds of hardware devices. These sensors generate large volumes of data. Further, due to cheaper storage costs, organizations are storing data for much longer periods than before. However, *Big Data* has created significant challenges in terms of processing and analyzing. As a result, *Big Data* has become a buzz word in industry where different parties attempt to tackle these new challenges. More importantly, *'Big Data'* does not have a clear definition. Typically, *'Big Data'* is defined with the help of its characteristics, widely known as 3V's [Zikopoulos 2012; Zaslavsky et al. 2012]: volume, variety, and velocity.

— **Volume:** *"Volume relates to the size of the data such as terabytes (TB), petabytes (PB), zettabytes (ZB), etc"* [Zaslavsky et al. 2012].
— **Variety:** *"Variety means the types of data. In addition, different sources can produce big data such as sensors, devices, social networks, the web, mobile phones, etc. Therefore, data could be web logs, RFID sensor readings, unstructured social networking data, streamed video and audio, and so on"* [Zaslavsky et al. 2012].
— **Velocity:** *"This means how frequently the data is generated, for example, every millisecond, second, minute, hour, day, week, month, or year. Processing frequency may also differ with user requirements. Some data needs to be processed in real-time and some may only be processed when needed. Typically, we can identify three main categories: occasional, frequent, and real-time"* [Zaslavsky et al. 2012]

Big data is usually discussed with respect to the cloud computing paradigm. Due to the fact that cloud computing theoretically provides infinite amounts of computational and storage resources, the typical ideology is to send all the data back to the cloud for processing. However, this is not always the case. There are downsides of heavily depending on cloud computing. For example, sending all the data collected all the time to cloud has high costs in terms of bandwidth, storage, latency, energy consumption (for communication) and so on. We discuss these issues later in this paper.

To address the weaknesses inherited by the cloud computing paradigm, a notion called *'fog computing'* has been proposed. Fog computing encourages a shift of handling everything in the core of cloud computing to handling at the edges of a network. To be specific, instead of sending all the data collected to the cloud, fog computing suggests to process the data at leaf nodes or at the edges. This idea is also called *'edge analytics'*. Local data processing helps to mitigate the weakness of cloud computing. In addition, fog computing also brings new advantages, such as greater context-awareness, real-time processing, lower bandwidth requirement, etc. We will discuss more advantages in detail later in this paper.

Our objectives in revisiting the literature related to fog computing are threefold: 1) to learn how the fog computing paradigm can be used to build sustainable Smart Cities on top of the IoT infrastructure, 2) to identify the most critical functionalities of an ideal fog computing platform, and 3) to highlight open challenges and to discuss future research directions with respect to developing fog platforms to support a wider ranges of use case scenarios.

This paper is organized into sections as follows: Section 2 introduces the fog computing paradigm from the IoT perspective by discussing its strengths, weaknesses and roles in the IoT era. Section 3 presents a number of use cases that would benefit from

the fog computing paradigm. In Section 4, we identify ten major characteristics and common features of the fog computing paradigm with the support of use case scenarios presented in Section 3. Comparisons and in-depth analysis of existing research efforts are presented in Section 5. Based on our review, we discuss lessons leaned in Section 6. Then, Section 6 discusses future research directions and challenges, before we conclude our review in Section 8.

1.1. Main Contributions

In this section, we summarize the contributions and the novelty of this paper. There have been several surveys conducted in this field. We briefly introduce these surveys in chronological order.

Bonomi et al. [Bonomi et al. 2012] have discussed the role of fog computing in the internet of things domain. Yi et al. [Yi et al. 2015a] have surveyed fog computing under seven themes, namely, fog networking, quality of service (QoS), interfacing and programming model, computation offloading, accounting, billing and monitoring, provisioning and resource management, security and privacy. Yi et al. [Yi et al. 2015b] have surveyed fog computing from the security point of view, where they have focused on number of security aspects such as trust and authentication, network security, secure data storage, secure and private data computation, privacy, access control and intrusion detection.

It is important to note that, none of these surveys have reviewed the fog computing domain from platform developers' and end users' perspectives. Further, this survey is mainly focuses on connectivity and device configuration aspects of the fog computing paradigm. Our objective is to identify major features that fog computing platforms need to support specially towards building sustainable sensing infrastructure for smart city applications. Towards this, we built a taxonomy (of common features) after conducting an extensive literature review. This paper is enriched by both real world use case scenarios and literature findings. We also compare different research efforts conducted in the past and present these in Table 5. Subsequently, we identified research trends and gaps based on the literature review. Finally, we highlighted research challenges and directions.

2. FOG COMPUTING: AN OVERVIEW

Fog computing [Forrest Stroud ; Bonomi et al. 2012] is *"an architecture that uses one or a collaborative multitude of end-user clients or near-user edge devices to carry out a substantial amount of storage (rather than stored primarily in cloud data centers), communication (rather than routed over the internet backbone), and control, configuration, measurement and management (rather than controlled primarily by network gateways such as those in the LTE (telecommunication) core)"* [Chiang 2015; Aazam and Huh 2014].

The fog computing paradigm does not necessarily stick to one architecture, instead it represents a notion that supports to push data analytics towards leaves (i.e., edge nodes) [Yannuzzi et al. 2014]. The important message to understand is *'towards leaf /edge node'*. Therefore, fog computing does not suggest that all analytics needs to be done on leaves, instead it promotes to use leaf nodes as much as possible within realistic limitations. Leaf nodes also represent the closeness to the users. Further, it also suggests that multiple nodes close to the leaves would work in a cooperative manner to accomplish a task. Fog networking constitutes two planes, namely, 1) control plane and 2) data plane [Chiang 2015]. The responsibility of the control plane is to configure and manage the network (e.g., sync routing table information, efficient traffic control). The data plane is responsible for transferring data from the originator to the destination

based on the routing information provided by the control plane logic [Chiang 2015]. Chiang [Chiang 2015] has highlighted main differences between Cloud and Fog.

— Store most of the data at or near the edge instead of sending all the data to the cloud all the time. Features such as Mobility (Section 4.5), Context Discovery and Awareness (Section 4.7), and Data Analytics (Section 4.9) facilitate these characteristics.
— Use local networks (i.e., closer to the leaf nodes) to transfer data to processing nodes instead of using backbone networks (e.g., Internet) to route the data. Features such as Multi-Protocol Support: Communication Level (Section 4.3), Multi-Protocol Support: Application Level (Section 4.4), Mobility (Section 4.5), Security and Privacy (Section 4.10), and Cloud Companion Support (Section 4.11) facilitate these characteristics.
— Process substantial amount of data at the edge devices (e.g. leaf nodes) instead of using cloud computing infrastructure. Features such as Semantic Annotation (Section 4.8), Data Analytics (Section 4.9) and Context Discovery and Awareness (Section 4.7) facilitate these characteristics.
— Edge device are self-governed, managed, and controlled instead of controlled cloud decision makers. Features such as Dynamic Discovery of Internet Objects (Section 4.1), Dynamic Configuration and Device Management (Section 4.2), Data Analytics (Section 4.9), Context Discovery and Awareness (Section 4.7) facilitate these characteristics.

Figure 1 illustrates how data moves from leaf nodes to the cloud backbone devices. The computational capabilities (e.g., processing, memory, communication) of each category is different. Right most device categories have less computational capabilities and cheaper in price. Left most device categories have higher computational capabilities and cost more. Even though there is no strict definition, fog computing refers to the devices belonging to categories 1 to 4. They have some capabilities to process data before sending it to the cloud.

It is important to note that, over time capabilities of these categories will increase dramatically. For example, a few years ago smart phone like devices had less than 1GB memory. However, today we have smart phones with 4GB or more memory. We can also observe the same pattern when considering different versions of Raspberry Pi Devices[1]. The idea is that capabilities of devices of each category will increase over time. However, the difference between categories will always remain due to form factor limitation and more importantly price points. Form factor limitation means that the amount of computational power that can be concentrated into a device always correlates to the size of the device. Some common characteristics of fog computing are *"proximity to end-users and client objectives, dense geographical distribution and local resource pooling, latency reduction for quality of service (QoS) and edge analytics/stream mining, resulting in superior user-experience and redundancy in case of failure"* [Forrest Stroud ; Preden et al. 2015].

---

[1]https://www.raspberrypi.org/magpi/raspberry-pi-3-specs-benchmarks/

Fig. 1.   Categorizations of IoT devices based on their computational capabilities.

One of the common goals of fog computing is to make *'big data'* smaller. This is done through intelligent sensing (i.e. sensing only useful data based on the knowledge and intelligence available locally at a given edge device), filtering, and data analytics (e.g. averaging, knowledge discovery and throwing away raw data). In 2020, the annually captured data in total is foretasted to exceed 1,600 exabytes, or equivalently 1.6 zettabytes [Markkanen 2015]. Fog computing is expected to reduce the amount of data collected and needing to be processed to a manageable level.

Further, data synchronization techniques from low end to high end will play a significant role in making the sensed data manageable, specially at the high ends of fog computing, where data is expected to be gathered and processed [Aazam and Huh 2014]. It is also critical to effectively push back information to the low ends whenever necessary. Typically, there are three schemes for data synchronization: Pull Scheme, Push Scheme, Push-Pull Scheme. The choice of data synchronization strategies will depend on the communication protocols in use and the requirements in the actual fog computing scenarios.

Another advantage of fog computing is lower latency. As shown in Figure 1, it takes time for the data to move from sensing edge devices to the cloud computing backbone where data will be processed. However, the actions based on data analysis need to be taken fast, sometimes in real-time, in order for such actions (or data) to be useful. The fog computing vision fits well in scenarios where data will be processed within edge devices. Most of the smart home event detection scenarios typically need to support quick responses (e.g., if a motion in the living room is detected during day time while occupants are not there, then send an alert to the occupants including an image of the living room). Some enterprise-level IoT applications may require real-time or near real-time physical interactions with the connected assets to avoid any machinery failures.

Fog computing can also ensure higher availability. Connecting to the cloud is less reliable due to a variety of different connectivity issues [Stojmenovic and Wen 2014]. This reduces the dependency on cloud infrastructures and enables edge devices to function uninterruptedly for a reasonable time period and perform analytics even if the connection to the cloud is lost. Madsen et al. [Madsen et al. 2013] have highlighted the importance of developing techniques to improve the reliability of fog computing. Having intelligent edge devices can also increase overall security of an IoT solution by encrypting data at the source. Further, edge devices can process raw data and produce less privacy and security sensitive secondary context knowledge [Stojmenovic and Wen 2014], so the data communication and transfer face a lesser security threat. Further, edge devices that can fully process data locally can eliminate the communication based security threats completely, which is a further advantage in the access to context knowledge. As a result of being close to users or physical fields, fog based IoT applications can gather more knowledge about the local setting.

Considering cost, fog computing will cost more than deploying dumb devices at the edge and thus it would increase the total cost of a fog-based IoT solution. However, in the long term such additional costs will become justified due to potential savings (i.e., reduced connectivity costs and extended life cycles of battery-operated devices, and more importantly potential savings in highly dynamic and analytically complex settings).

The above mentioned advantages, that fog computing brings, are traditional weaknesses of cloud computing. Therefore, it is clear that fog computing can mitigate weaknesses of the cloud computing paradigm. However, fog computing has its own weaknesses as well. Edge devices are less computationally capable (e.g., less processing power, less storage, and less memory). This means that edge devices can only handle smaller amount of data and provide less processing capacity. Further, edge devices

have less knowledge about the big picture. Therefore, some types of processing (i.e., data analysis) that require global knowledge may not be able to perform. In addition to the resource limitations, edge devices may have other limitations, such as energy. Some edge devices may run on batteries or alternative energy sources such as solar power. Therefore, certain types of data processing that require significant amounts of energy may not be suitable to run on edge devices.

Smart home solutions have traditionally used fog computing approaches over the last decade [Perera et al. 2015a; 2014c]. However, to realize the real potentials of fog computing, edge analytics needs to be incorporated with IoT solutions in enterprise-level applications such as smart cities, manufacturing, healthcare, agriculture transportation, etc. In the next section we introduce several different and promising scenarios where fog computing has a potentially important role to play. These use cases have motivated us to explore and investigate challenges related to fog computing.

3. USE CASE SCENARIOS

In this section, we present four different use case scenarios. Our objective is to use these scenarios to extract major functional requirements and characteristics of fog computing platforms. The use cases presented in this section are on 1) Smart Agriculture, 2) Smart Transportation, 3) Smart Healthcare, and 4) Smart Waste Management. These generic use case scenarios were extracted, with minor alterations, from two of our previous publications [Perera et al. 2014b; Perera et al. 2015c]. We refer to the scenarios throughout this paper and have used them to extract major functionalities of fog computing platforms.

3.1. Smart Agriculture

"Agriculture is an importation part of smart cities as it contributes to the food supply-chain that facilitates a large number of communities concentrated into cities. This smart agriculture case study is based on two real world projects: Phenonet [Commonwealth Scientific and Industrial Research Organisation (CSIRO), Australia 2011] and OpenIoT [OpenIoT Consortium 2012]. In this scenario, the general public is not directly involved as in the smart home domain. To be specific, in Figure 2(a), we illustrate how the sensing works in the smart agriculture domain. In the following, we demonstrate the applicability of fog computing towards agricultural research through describing the Phenonet project in detail. Phenonet has a network of sensors collecting information over a field of experimental crops. Researchers at the High Resolution Plant Phenomics Centre are testing a network of smart sensor nodes that are able to monitor plant growth, performance information and climate conditions. Experimental plants are monitored using different types of sensing devices and techniques. First, sensor stations are used to monitor the plants in the field. Second, air balloons called 'blimps' are used to sense the field from sky. Third, field vehicles such as Phenomobiles are used to capture data on plant growth. Fog computing can play a significant role in performing the above mentioned sensing tasks more efficiently and adaptively. Ideally, fog gateway devices can be used in all three types of sensing approaches mentioned above to make the sensing process more efficient" [Perera et al. 2014b]. As shown in Figure 2(a), stationary sensor stations, *Phenomobiles*, and *'blimps'* act as edge devices. They combines both sensors and gateway devices. However, in some situations, sensor stations only have sensors and gateway devices fit into *Phenomobiles* act as the gateway. Average the data over different time frames in order to reduce data storage and communication are the most common data aggregation techniques performed on the gateway devices. Further, data analysis techniques are also used measure quality of service parameters.

"Context information plays a critical role in efficient sensing in smart agriculture related applications. The objective of collecting sensor data is to understand plant growth
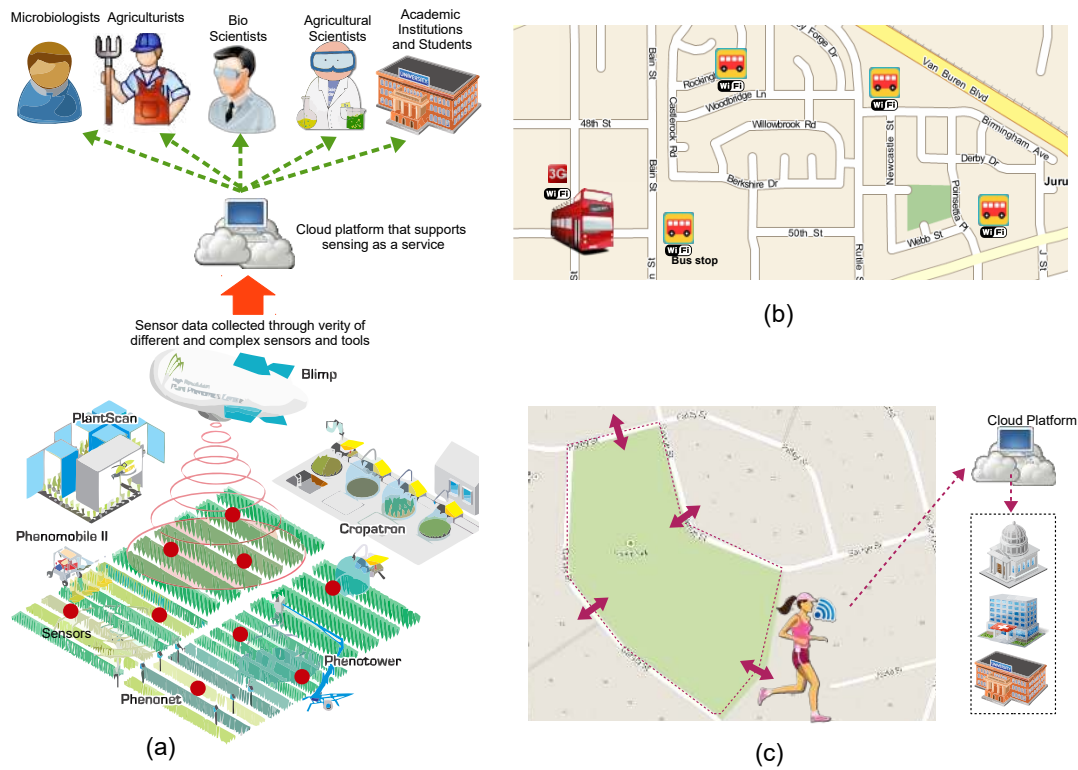
Fig. 2.   Fog Computing based Use Case Scenarios: (a) Different types of agricultural field vehicles can be fitted with sensing devices in order to monitor the growth of seeds and plants, (b) Sensors are deployed in buses in a Smart City environment and the data is expected to be collected based on context information and conditions provided by the data consumer, (c) Wearable sensors can be used to monitor movements of the general public who use public spaces such as parks for exercising and recreational activities. So the authorities can plan further development and upgrades of the infrastructure.

better by applying fusing and reasoning techniques. In order to accomplish this task, sensor data needs to be collected in a timely and location-sensitive manner. Each sensor needs to be configured by considering context information. In some situations, severe frosts and heat events can have a devastating effect on crops. Flowering time is critical for cereal crops and a frost event could damage the flowering mechanism of the plant. However, the ideal sampling rate could vary depending on both the season of the year and the time of day. For example, on one hand, a higher sampling rate is necessary during winter and nighttime. In contrast, lower sampling would be sufficient during summer and daytime. On the other hand, some reasoning approaches may require multiple sensor data readings. One example for this is that, a frost event can be detected by fusing air temperature, soil temperature, and humidity data. However, if the air temperature sensor stops sensing due to malfunction, it would become useless to continue sensing humidity. The reason is that frost events cannot be detected without temperature. In such circumstances, configuring the humidity sensor to sleep is ideal until the temperature sensor is replaced and starts sensing again. Such intelligent (re-)configuration can save energy by eliminating ineffectual sensing and network communication" [Perera et al. 2014b].

3.2. Smart Transportation

"John, a researcher at the Department of the Environment, is interested in measuring and monitoring air pollution in cities. John's team has deployed sensor kits in buses. Each of these sensor kits consists of multiple sensors and a communication device with both WiFi and 3G capabilities. The team has developed an application that processes data collected by these sensor kits. This application consists of a number of different algorithms that analyze and visualize air pollution in a city. However, according to the way that the algorithms are written, John only needs to collect data when a bus is moving. Sensor data that is captured while the bus is stopped at a bus stop, or in traffic, does not add any value. Therefore, John would like to collect sensor data only when the bus is moving. Further, John does not need real-time data in most of the occasions. Therefore, it is sufficient to push the sensor data to the cloud when the bus reaches a bus stop. The communication devices fitted in the bus will connect to the bus stop's WiFi and push the data collected since the last bus stop to the cloud, as illustrated in Figure 2(b). Meanwhile, John is also interested to receive sensor data in real-time when raining. Therefore, when raining, the communication devices need to use 3G to upload the sensor data to the cloud. However, they still need to adhere to the first rule that says sense only when moving" [Perera et al. 2015c]. In this scenario, sensor kits on buses act as ICOs. The gateway devices are fitted to the bus stations.

In this scenario both sensor kits attached to the bus as well as to the bus stop act as fog devices to make the sensing process more efficient. Fog computing technologies are widely used to build connected vehicle based applications [Bonomi et al. 2012; Datta et al. 2015]. Datta et al. [Datta et al. 2015] have proposed an architecture for connected vehicles with Road Side Units (RSU) and M2M Gateways. RSUs are access points where vehicles can connect themselves to M2M gateways. M2M data analytics and vehicle discovery are the core features of this architecture.

3.3. Smart Health and Well-Being

"Michael is working for the Department of Public Health and Well-Being. He has been asked to develop a plan to improve the public health in cities by improving the infrastructure that supports exercise and recreational activities (e.g., parks and the paths that supports jogging, cycling, and venues for bar exercise). Michael developed a wearable light-weight sensor kit that can monitor a variety of different parameters, such as air quality, sound and movement. Further, Michael has recruited volunteers who are willing to wear those sensor kits when exercising. A sensor kit can collect data and push to the volunteer's smart phone. A smart phone application pushes data to the cloud once it gets connected to the Internet. Firstly, Michael only needs to collect data when a volunteer enters the park areas as illustrated in Figure 2(c). Further, Michael only needs to perform sensing only when the volunteers are moving (e.g., walking, running, and cycling). Meanwhile, Michael has noticed that there are a large amount of people coming to the park during the weekend. In order to reduce the burden to the volunteers, Michael only needs to collect data from a maximum of 30 sensors kits (i.e., volunteers) despite the actual number of volunteers visiting the park at weekends" [Perera et al. 2015c]. In this scenario, the sensor kits are ICOs and the mobile devices are the gateways.

In this scenario smart phones act as the fog devices to make the sensing process more efficient. In line with this use case, Zao et al. [Zao et al. 2014] have used fog computing to develop brain computer interaction application. In their application, EEG data is pre-processed on the fog reducing latency and data communication.

3.4. Smart Waste Management

"Waste management is one of the toughest challenges that modern cities have to deal with. Waste management consists of different processes such as collection, transport, processing, disposal, managing, and monitoring of waste materials. These processes cost a significant amount of money, time, and labor. Optimizing waste management processes help to save money that can be used to address other challenges faced by smart cities. Figure 3 illustrates how the sensing works in the waste management domain. In a modern smart city, there are several parties who are interested in waste management (e.g., city council, recycling companies, manufacturing plants, and authorities related to health and safety). For example, a city council may use sensor data to develop optimized garbage collection strategies, so they can save fuel cost of garbage trucks. Additionally, recycling companies can use sensor data to predict and track the amount of waste coming into their plants for further processing so that they can optimize internal processes. Finally, health and safety authorities can monitor and supervise the waste management process without spending a substantial amount of money for manual monitoring inspections" [Perera et al. 2014b].

"In order to perform waste management, different types of sensors need to be deployed in different places such as garbage cans and trucks. These sensors need to detect various kinds of information, including the amount of garbage, types of garbage, and so on. As we have depicted in Figure 3, direct and indirect communication strategies can be used to collect and communicate sensor data to the cloud. Sensors with energy harvesting capabilities are important in this domain [Alvarado et al. 2012]. As represented in step (1) in Figure 3, low powered and low capable sensors [Watteyne et al. 2012] can be used to sense and data can be uploaded to the cloud with the help of nearby infrastructure (e.g., through communication devices attached to street lights or similar infrastructures that have access to rich energy sources and communication capabilities). In addition, when long range communication is not available, data can be uploaded to the cloud with the help of auto-mobiles, as depicted in step (2) in Figure 3, such as garbage trucks, city council vehicles and buses that operate in the near areas. Furthermore, both active and passive sensors can be used to sense the environment [Chaves and Decker 2010]. Direct communication can be done via technologies such as 3G, which makes this approach less dependent on third parties (as depicted in (3) in Figure 3)" [Perera et al. 2014b]. In this scenario, waste bins are fitted with ICOs and street lights and garbage trucks act as gateways.

In this scenario, we highlight the ability of fog computing platforms in collecting data from sensors with short range capabilities and opportunistically uploading data to the
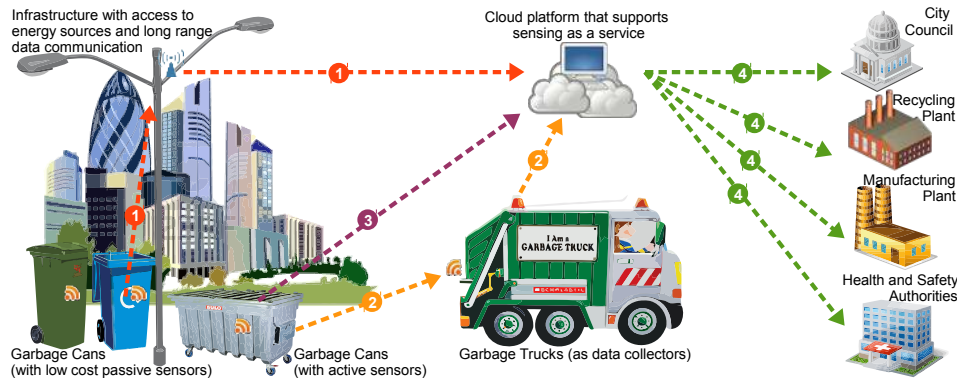

Fig. 3.   Efficient waste management in Smart Cities

cloud after aggregation. For example, sensors may record how the garbage bins get filled over time. However, in order to reduce data communication (also energy usage), they may only send the data point that is sufficient to plot a graph in acceptable level of accuracy.

The above scenarios highlight some major capabilities of the fog computing paradigm. We will revisit these scenarios in the next section, where we identify individual features of fog computing.

3.5. Smart Water Management

The water system of a city is one of the most important aspects for building future smart cities. Effective and energy-efficient transportation and use of water, and cost-effective and environment-friendly treatment of wastewater are critical in smart water management at city-scales. A smart water system is expected to help monitor city-wide water consumption, transportation, prediction of future water use, and so on. For example, Figure 4 describes the needs of water harvesting and ground water monitoring, which will rely on the support from Fog/Cloud computing infrastructure, such as wireless sensors, smart meters, GPS devices, Fog gateways, Cloud platforms, IPv6 technology and long-distance communication protocols like 3G, 4G, LTE, etc. On top of all these useful aspects of the city water network, the smart water system is expected to analyze collected information and generate actionable data for management, reduction of water losses and other improvement of the city water system.
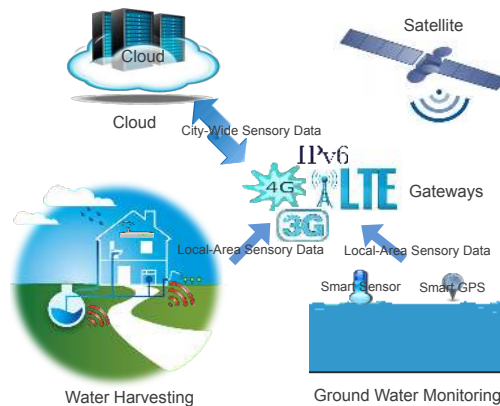

Fig. 4.   Smart Water Management.

3.6. Smart Greenhouse Gases Control

Greenhouse gases control requires collective efforts. Governments can take actions and personal lifestyles matter too. Smart greenhouse gases control systems in future smart cities can help to collect critical information for governments to make better decisions on policies in order to effectively reduce greenhouse gases. Such smart systems can also work with city habitats to guide them on making their own efforts in helping reduce greenhouse gases. Figure 5 shows an example of a future greenhouse gases control system. Smart monitoring can cover homes, schools, offices, factories, vehicles, and so on. Individuals and workers can be reminded instantly based the smart monitoring results for taking simple and quick actions to help reduce greenhouse gases, specially with the help of Fog devices in the Fog smart computing paradigm. Govern-

ments can make timely decisions and policies on top of the smart monitoring results of greenhouses in city-wide environments and infrastructure.

3.7. Smart Power Grid

In future smart cities, smart power grids will be critical in ensuring reliability, availability, and efficiency in city-wide electricity management. Figure 6 demonstrates an example future smart grid system, where Fog/Cloud computing can play a significant role. A successful smart grid system will be able to help improve transmission efficiency of electricity, react and restore timely after power disturbances, reduce operation and management costs, better integrate renewable energy systems, effectively save electricity for future usage, and so on. It will also be critical in building better electricity networks to help bring down electricity bills and balance the whole electricity system. In addition, the smart power grid system should monitor power generation, power demands and help make storage decisions. In terms of security, a smarter grid will also add resiliency to large-scale electric power systems so as to help governments react promptly to emergencies or natural disasters, e.g., severe storms, earthquakes, large solar flares, and even terrorist attacks, etc.

3.8. Smart Retail Store Automation

In future smart cities, shopping experience can be improved dramatically with smart retail store automation. As shown in Figure 7, with the adoption of Fog/Cloud computing technologies, the whole cycle of retail store can be automated with better machine intelligence. For example, a smart retail store physically can support automated scanning, express self-shopping, high-/medium-volume checkout, etc. Virtually, a smart retail store system can help advertise to the best social networks/communities/groups to attract best interests from customers, keep track of sales of assorted products, and hence manage inventory and products ordering effectively and automatically. Cost-effective transportation will also be recommended. Such retail store automation systems will be expected to reduce operation and management costs, react promptly to demands from customers.

4. COMMON FEATURES IN FOG COMPUTING

In this section, we identify the major features that are useful in supporting different types of use cases in smart cities. We extract them by both analyzing use case scenarios and reviewing relevant literature. Note that, there are already a number of different academic and industrial research projects implementing some of the features

Fig. 5.   Smart Greenhouse Gases Control.
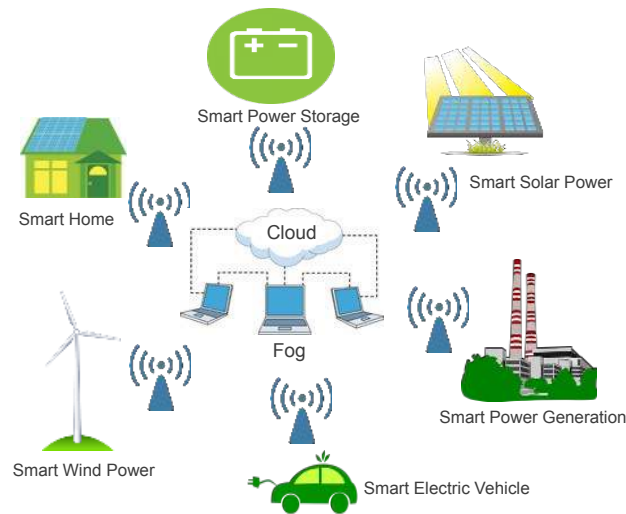
Fig. 6.   Smart Power Grid Control.

Fig. 7.   Smart Retail Automation.

we discuss in this section. More details about those projects will be provided in the next section.

Let us first introduce what are *edge devices / leaf nodes*. Previously, we introduced a spectrum of devices, in Figure 1, that are expected to find use in IoT solutions. In this paper, we use the term *'edge devices'* or *'leaf nodes'* to refer to any of the devices

Fig. 8.   Sample Internet connected objects. ICOs can be in any form as shown

in category 1,2,3, and 4. As shown in Figure 1, these devices have less computational capabilities compared to cloud composing infrastructures. Figure 8 illustrates several different types of edge devices. These devices could be raw sensors (e.g., Libelium sensors [Asin and Gascon 2012]), smart plugs, mobile devices, smart watches, smart bottles, or smart fridges, just to mention a few. Each of these devices can become a leaf node in an IoT application and perform edge analytics. Any data analysis task performed within an edge device (or leaf node) can be identified as edge analytics. For example, without pushing energy consumption data every second to the cloud, a smart plug itself may perform data processing and avoid redundant data. A smart plug may only send data when there is a fluctuation of the energy consumption. This way the smart plug reduces the total data communication by acting as a fog device. In other words, both communication bandwidth and the total amount of data sent to the cloud over a certain period can be reduced. Internet Engineering Task Force (IETF) 7228[2] has proposed a terminology for constrained-node networks. They aim to standardise small devices with severe constraints on power, memory, and processing resources.

IETF defines constrained nodes / devices as *"A node where some of the characteristics that are otherwise pretty much taken for granted for Internet nodes at the time of writing are not attainable, often due to cost constraints and/or physical constraints on characteristics such as size, weight, and available power and energy. The tight limits on power, memory, and processing resources lead to hard upper bounds on state, code space, and processing cycles, making optimization of energy and network bandwidth usage a dominating consideration in all design requirements. Also, some services such as full connectivity and broadcast/multicast may be lacking"*

IETF categorise device based on number of criteria: 1) availability of memory, 2) availability of energy, 3) availability of communication. Under each criteria, they have identified few different classes as follows:

(1) Availability of memory
   — C0: $<<$ 10 KiB data size (e.g., RAM), $<<$ 100 KiB code size (e.g., Flash)
   — C1: appx. 10 KiB data size (e.g., RAM),   100 KiB code size (e.g., Flash)
   — C2: appx. 50 KiB data size (e.g., RAM),   250 KiB code size (e.g., Flash)
(2) Availability of energy
   — E0: Event energy-limited (e.g., Event-based harvesting)
   — E1: Period energy-limited (e.g., Battery that is periodically recharged or replaced)
   — E2: Lifetime energy-limited (e.g., Non-replaceable primary battery)
   — E9: No direct quantitative limitations to available energy (e.g, Mains-powered)
(3) Availability of communication
   — P0: Normally-off (Reattach when required)
   — P1: Low-power (Appears connected, perhaps with high)
   — P9: Always-on (Always connected)

However, for our discussion in this paper, the exact hardware specifications are not that important. In this paper, we only wanted to highlight the differences of device categories broadly. Typical, Micro-controllers without an OS can be identified as ICOs. Gateway devices usually run an Operating system.

Over the next ten sub sections, we will introduce the most common features that need to be facilitated by an ideal fog computing platform for IoT applications. We extracted these features through extensive literature and use case scenario analysis. In each section, we present representative set of approach proposed by researchers in the past. Further details can be obtained though the references presented in Table III.

---

[2]https://tools.ietf.org/html/rfc7228

4.1. Dynamic Discovery of Internet Objects

Discovery in IoT has different meanings. Some discoveries happen in the cloud where a user may try to find an Internet Connected Object[3] (ICO) from a data store or a management system. Figure 8 illustrates some ICOs. In this section, we use the term discovery to refer to the activity of connecting an ICO to a gateway device [Aazam and Huh 2014] in order to build a fog network. Some major challenges in discovery include heterogeneity, security, and dynamicity. Heterogeneity is in multiple levels from communication protocols, application protocols, discovery sequences, sensing and capabilities, and so on. For example, each ICO may have different sensing capabilities as shown in Figure 9. Further, each sensor may use different types of application level discovery sequences and security measurements, which are hard to standardize [Perera et al. 2014b]. An example is shown in Figure 10.
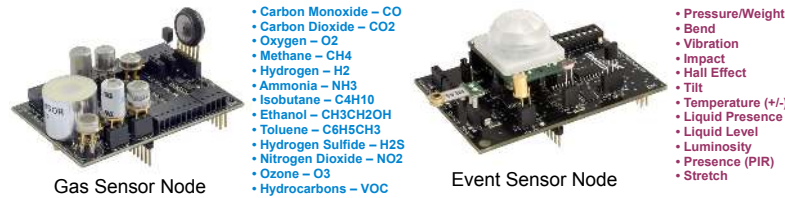

- Carbon Monoxide – CO
- Carbon Dioxide – CO2
- Oxygen – O2
- Methane – CH4
- Hydrogen – H2
- Ammonia – NH3
- Isobutane – C4H10
- Ethanol – CH3CH2OH
- Toluene – C6H5CH3
- Hydrogen Sulfide – H2S
- Nitrogen Dioxide – NO2
- Ozone – O3
- Hydrocarbons – VOC

Gas Sensor Node

- Pressure/Weight
- Bend
- Vibration
- Impact
- Hall Effect
- Tilt
- Temperature (+/-)
- Liquid Presence
- Liquid Level
- Luminosity
- Presence (PIR)
- Stretch

Event Sensor Node

Fig. 9.   Heterogeneity in terms of sensing/measurement capabilities of sensor nodes

1 Hello!
2 Hi!
3 Get_Identification
4 Type: Gas_Node; Manufacturer: Libelium
1 Who!
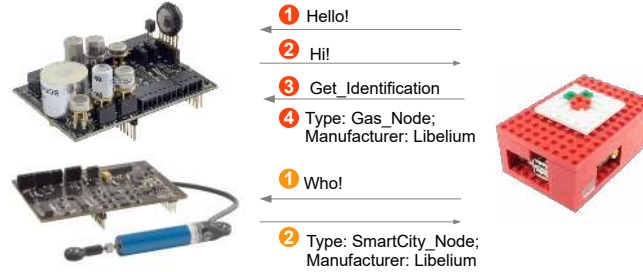2 Type: SmartCity_Node; Manufacturer: Libelium

Fig. 10.   Heterogeneity in terms of communication sequences.

Alongside sensing data from environments, reaction to environmental changes is critical. To this end, actuators can be used to put something into automatic action and can work with assorted sensors. In Fog computing and Internet of Things, actuators will play an important role, where machine actions can be taken automatically without human intervention. This is important specially in the situations where human beings will be at risk or instant actions are required, and so on.

Actuators can be categorized into linear actuators and non-linear actuators, depending on the direction of motion generated by the actuator. These actuators can also be categorized into types of actuators based on the source of energy for the motion, including mechanical actuators, hydraulic actuators, pneumatic actuators, piezoelectric

---

[3]We use both terms, *'Internet Connected Objects'*, *'objects'* and *'things'* interchangeably to give the same meaning as they are frequently used in IoT related documentation. Some other terms used by the research community are *'smart objects'*, *'devices'*, *'nodes'* [Belli et al. 2015].

actuators, etc. Take linear actuators as an example, some common linear actuators include [Gonzalez 2016]: Pneumatic linear actuators rely on pressure from an external compressor or manual pump to produce required motion. Hydraulic linear actuators are similar to pneumatic actuators, but they gain force from an incompressible liquid from a pump. An electric linear actuator relies on electrical energy to drive motion. Different types of actuators have their own advantages and disadvantages in a number of aspects, e.g., pressure limitation, temperature scope, accuracy, reaction time, maintenance and programming cost, etc [Gonzalez 2016]. In microelectromechanical systems (MEMS), microactuators are fundamental and are designed based on different actuation principles, such as shape-memory alloys, electrostatic, electrothermal, piezoelectric, pneumatic and electromagnetic etc [Jia and Xu 2013].

On the other hand, nanoscale technology will develop devices as small as one to a few hundred nanometers (10^{-9} meters) [Tracy 2016]. Dynamic discovery and configuration is critical in nanoscale sensor domain as the number small devices are growing rapidly and they are getting deployed everywhere. These nano sensors need to be discovered and configured continuously as they tend to be unreliable. They also tend to move from one place to another. It is reported that the U.S. National Nanotechnology Initiative requested $1.5 billion in federal funding in the year of 2016, which was also roughly the average amount of annual funding awarded to the U.S. National Nanotechnology Initiative in the last 15 years. Nowadays, there are some significant advances in the development of nanosensors [Garcia-Martinez]. For example, some nanosensors to date created by the tools of synthetic biology can be used to modify single-celled organisms, such as bacteria. Other nanosensors made from non-biological materials include carbon nanotubes, which can both sense and signal, acting as wireless nanoantennas.

There are many potential applications of sensors at nanoscale (nanosensors). For example, in-body networks can help monitor real-time blood, sickness and breath tests. Hence, intrabody nano-networks for healthcare applications can provide real-time health information and work status, where personal health can be monitored anywhere and anytime [Akyildiz and Jornet 2010]. Further, the spread of viruses and diseases can be monitored in public locations and real-time actions can be arranged. In the future, billions of nanosensors will help harvest huge amounts of real-time information of our cities, homes, offices, factories, and even our bodies. Fog computing and Cloud computing technologies can then provide much more detailed, inexpensive and up-to-date pictures of our living environments, leading to the Internet of Nano-Things.

The ICO detection and discovery may happen in two different ways as shown in Figure 11. One method is that ICOs may continuously search for fog gateway devices while gateway devices are passively waiting to be connected with ICOs. In this method, gateway devices become discoverable. In the other approach, fog gateway devices may continuously search for ICOs around them. In this method ICOs need to be discoverable. Typically, both methods may use low range communication protocols. We will briefly introduce several different types of discovery and communication protocols proposed in the IoT domain in Section 4.4.

4.2. Dynamic Configuration and Device Management

In IoT, connection establishment and configuration need to be occurred in two different places. Firstly, ICOs need to be connected to gateway devices. Secondly, fog gateways need to connect to IoT cloud platforms. Above two approaches are illustrated in Figure 12. It is important to note that industrial IoT cloud platforms such as IBM Blumix and Microsoft Azure IoT distinctly identify sensing devices and fog gateways as two separate categories and provide specialized configuration options unique to each of the categories.
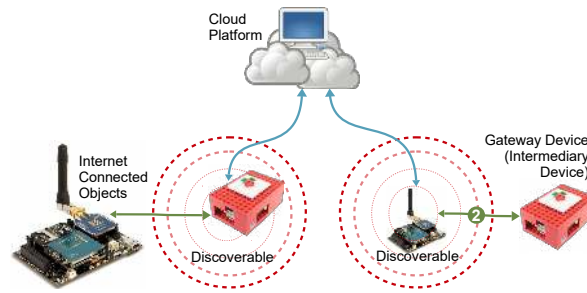
Fig. 11.   ICO detection and discovery

Fig. 12.   Connectivity between ICOs and Cloud platforms need to be established and configured.

Let us walk through the process of connection establishment and configuration in an IoT application. First, ICOs need to connect to its corresponding fog gateway. To do this, both parties need to negotiate and agree to a common protocol and message sequence [Perera et al. 2014a]. Next, ICOs need to introduce them to fog gateways by describing themselves (e.g., sensor ID, manufacture details, sensor types). Then, the fog gateways can use such information to configure themselves accordingly (e.g., prepare to accept data in a certain way) [Perera et al. 2014a]. Further more, ICOs also need to configure their scheduling calendar, communication frequency, data acquisition methods and sampling rate [Perera et al. 2014a]. In the next step, fog gateways need to send the data they collected from ICOs to the IoT cloud platforms. This allows the IoT clod platform to prepare itself to accept incoming data and send commands back to the ICOs. We identify this process as registration. As a result of the registration process, the cloud IoT platform gets to know about the data availability through each fog gateway. Additionally, the registration process provides several pieces of information to the cloud about fog gateways and ICOs (e.g., location, energy source and level). The cloud IoT platforms can use such information to develop energy and privacy aware data collection plans [Perera et al. 2014b]. As we will discuss later, such efficient data collection plans are vital towards building sustainable smart cities. Typical, each ICO may have support one protocol and message sequence that can be used for configuration [Perera et al. 2014a]. Gateways will be required to intelligently find out which message sequence need to be used to configure a given ICO. The ICO description languages such as Hypercat [Hypercat Consortium 2016] can be used to identify how to configure each ICO when a given ICO met a gateway for the first time.

In a typical pervasive computing environment, only few ICOs are used in an application (e.g., smart home, smart office). However, the IoT paradigm aims to connect billions of sensors to the Internet. Therefore, establishing connectivity between ICOs and IoT applications becomes a challenge that developers need to face more frequently. It is inefficient and cumbersome to connect ICOs to IoT applications manually through hard wired coding [Perera et al. 2014a]. It is essential to develop automated (or at least semi-automated) techniques to support dynamic configuration of ICOs. Towards this, one of the challenges is to develop ICO description techniques so the IoT appli-

cation can find more information about each ICO and how to connect to them (e.g., sensors' capabilities, data structures they produce, hardware/driver level configuration details). Over the last few years, there has been few different approaches that aims to tackle this challenge of describing ICOs. Sensor ontologies [Compton et al. 2009], Transducer Electronic Data Sheet (TEDS) [IEEE Instrumentation and Measurement Society 2007], Sensor Device Definitions [Perera et al. 2012], and Sensor Markup Languages (SensorML) [Botts and Robin 2007] to name a few. Further, there are recent application level protocols and frameworks proposed to automate the ICO configuration process and make it scalable. Some examples are AllJoyn, Iotivity [Linux Foundation 2016b], HyperCat [Hypercat Consortium 2016]. We will briefly introduce these protocols and frameworks in Section 4.4.

Based on Responsibility. As shown in Figure 13, data can be captured by sensors using two main approaches [Pietschmann et al. 2008]: push and pull. A detailed comparison is presented in Table V in [Perera et al. 2014a].

— Pull: The fog gateways retrieve data from ICOs using this techniques. Further, IoT cloud platforms may also use pull techniques to acquire data from fog gateways.
— Push: The ICOs may use this techniques to send data to fog gateway. Similarly, the fog gateway may also use this techniques to push data to the IoT cloud platform.
The physical or virtual sensor pushes data to the software component that is responsible to acquiring sensor data periodically or instantly. However, periodical or instant pushing can be employed to facilitate a publish and subscribe model.
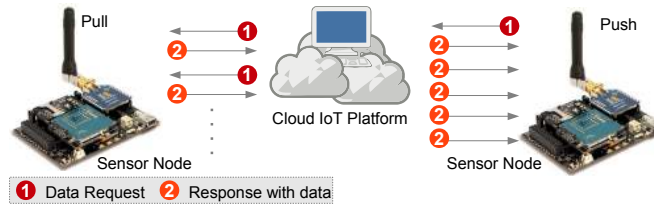

Fig. 13. Both pull (left side) and push (right side) techniques can be used retrieve data from ICOs

Based on Frequency. More broadly, data is captured by sensors using two main approaches: instant events and interval events. A detailed comparison is presented in Table VI in [Perera et al. 2014a].

— Instant (also known as threshold violation): This method sends data to a certain destination when an event get triggered. An event can be triggered by the sensors themselves (e.g., temperature reach a certain number, motion sensor detect a motion). This approach supports both push and pull techniques. However, ICO can only use the push method where it sends data when an event occurs. On the other hand, a fog gateways may use pull approach to query it's ICOs when an event occurs.
— Interval (also known as periodically): This method sends data to a certain destination periodically. That means that the event is triggered by *time*. For example, an ICO may be configured to sense and send data every 20 seconds. On the other hand, a fog gateway may be configured to pull data from ICOs periodically every 20 seconds. We identify data sensing date as *'sampling rate'*. This approach supports both push and pull techniques.

So far we have discussed several different parameters that need to be configured in both ICOs and fog gateways. Such configurations are dynamically performed based
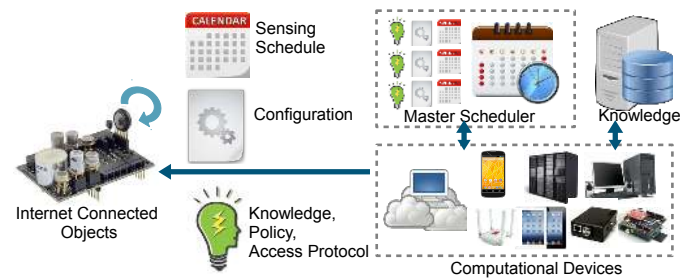
Fig. 14.   Dynamic configuration ICOs and fog gateways based on context

on the application requirements. Sampling rate and communication frequency are the two most important parameters, because they make direct impact on the lifetime of ICOs due to energy constraints. This is a very important factor in building sustainable smart cities. Typically, both ICOs and fog gateways do not have the full picture of the sensing application. Therefore, ideally, the responsibility of designing an efficient sensing plan needs to be passed-on to the cloud IoT platforms. Firstly, the cloud IoT platforms needs to establish an overall sensing strategy by considering the overall requirements of the smart city application. Secondly, it needs to delegate the sensing responsibility to individual fog gateways and ICOs by dividing the overall sensing plan into mini schedules. These mini sensing schedules and configuration details then need to be pushed into both gateways and ICOs as shown in Figure 14. However, it is important to note that overall sensing requirements of a smart city application may change over time at runtime. Therefore, the above mentioned process may need to take place repeatedly to maintain the overall efficiency of the application over time [Jiang Zhu et al. 2013]. Both ICOs and fog gateways are typically unreliable and prone to breakdowns and malfunction that will require replacements. Cloud companions can be efficiently used to keep track of both ICOs and gateways. Ideally, an image (or a configuration clone) needs to be store in the cloud so that in case of a breakdown, new devices can be quickly deployed and configured using the cloud images instead of conducting discovery and configuration processes which take time and energy.

4.3. Multi-Protocol Support: Communication Level

Even though our intention is not to survey different communication protocols, we briefly introduce some major communication protocols used in smart city applications and especially, in the fog computing domain. It is important to note that none of these protocols is superior to another. This is simply because all of them are superior in some aspects and weak in other aspects. Ideally, each protocol is designed to support different use case scenarios efficiently than other alternatives. In the fog computing domain, specially from the gateways point of view, it is important to support different types of communication protocols so that a wider range of edge nodes or ICOs can be connected to these gateways. In Figure 17, we present a comparison of IoT communication protocols from three different perspectives, namely, (a) communication range, (b) bandwidth, and (c) power consumption.

Figure 15 presents a simplified version of Open Systems Interconnection model (OSI model) [Reiter 2014]. In parallel, we present an example using TCP/IP stack. Before we get into the discussion of different communication protocols, it is useful to understand where each protocol fits in. The link layer is responsible for converting radio signals to bits and vise versa. The network layer provides addressing capabilities that are being used to route data through networks while the transport layer manages the

communication between two application endpoints. Finally, the application layer is responsible for data formatting.
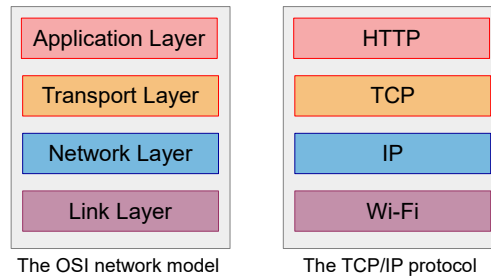

Fig. 15. Simplified Version of OSI Model and TCP/IP/HTTP Stack

In Figure 16, we illustrate how different communication protocols fit in with respect to the OSI model [Reiter 2014]. Some of the protocols are designed to communicate over short distances and are more suitable to conduct communication between ICOs and fog gateways. Other protocols are typically suitable to communicate between gateway devices and the cloud infrastructure. As a result, in fog computing, gateways should be able to deal with different types of communication protocols, mostly in parallel as illustrated in Figure 12.
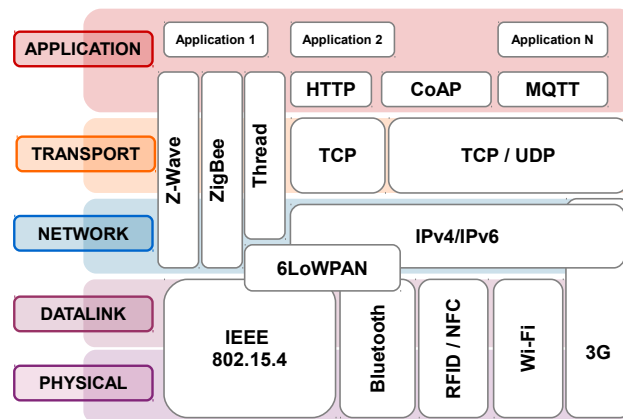
Fig. 16. Overview of Communication Protocols with respect to OSI model [Reiter 2014]

As we mentioned earlier, each communication protocol has its own strengths and weaknesses [Reiter 2014]. For example, one issue in TCP/IP based communication is that it is a fairly complex protocol. TCP/IP is larger in size and requires comparatively considerable amount of computational resources and memory. Such hardware requirements are sometimes hard to facilitate by ICOs in IoT due to miniature sizes and cost related issues. The complexity also leads to larger data packets and more power required to send and receive packets. To address this issue, different parties have developed a variety of different protocols as discussed below. However, today hardware is becoming cheaper and smaller than ever before. This movement encourages adopting TCP/IP even for resource constrained ICOs. The network topologies supported by each protocol is important as they impact performance and cost of deployment. They helps

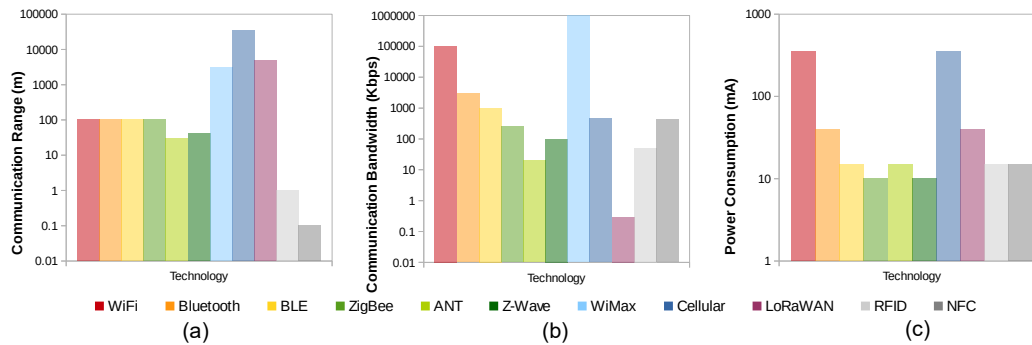Fig. 17.   (a) Communication Ranges; (b) Communication Bandwidth; (c) Power Consumption. It is important to note that the above measurements can vary greatly based on both external factors, e.g., how the measurements are taken, and how the hardware devices are being built. Further, some protocols have a number of different variations that have different capabilities with regards to the above aspects. To simplify the presentation, we restrict our summary to the most popular version of the protocols. Our intention is to give a broader look at how these different protocols fit in the IoT space. Data Sources: [Halperin et al. 2010; Gomez et al. 2013; Bluetooth 2005; Mohandas et al. 1060; Dementyev et al. 2013; Reiter 2014; Nuaymi 2007]

to determine which protocol to use in which parts of the fog network in a given IoT application scenario.

Wi-Fi: This is one of the most commonly used wireless local area network (WLAN)technology [Yeh et al. 2003]. Wi-Fi can be used to connect different types of nodes (e.g., ICOs) to a network so they can communicate with each other. It is developed by Wi-Fi Alliance [Henry and Luo 2002]. Substantial amounts of IoT devices are expected to connect to the Internet via Wi-Fi [Texas Instruments 2014]. However, most of the time, Wi-Fi is used by high end devices such as computers, tablets, mobile phones, gaming devices printers, cameras, and more recently smart refrigerators and microwaves. These high end devices are connected to a WLAN using Wi-Fi through as access point (or hotspot). Typically, 20 meters (66 feet) coverage can be provided by a single hotspot. However, coverage may be extended in out doors due to lack of obstacles. Typically, up to 250 devices can be connected to each Wi-Fi access point in parallel [Reiter 2014]. Typically, an IoT device can be connected to the Internet using Wi-Fi over a year using two AA alkaline batteries by using efficient power management (e.g., fast on-off time, long sleeps). Wi-Fi is an implementation of the standard IEEE 802.11. Wi-Fi Protected Access (WPA), Wired Equivalent Privacy (WEP), and WPA with Advanced Encryption Standard (WAP2) have been introduced over time where the latter ones are better then the previous [Lashkari et al. 2009]. However, non of these methods are considered fully secure. Virtual Private Networks (VPN) [Harmening 2013] and secure Hypertext Transfer Protocol over Transport Layer Security (HTTPS) are recommended to use to secure the Wi-Fi based communication. Wi-Fi mostly uses the star network topology to connect devices.

Bluetooth: This is also one of the most common wireless communication technologies developed to enable wireless personal area networks (WPANs) [Zimmerman 1996]. Bluetooth is deigned to support data communication over short distances. This standard is developed by Bluetooth Special Interest Group and standardized as IEEE 802.15.1 [Bluetooth 2005]. Bluetooth has three variations in terms of device classes where each class supports different communication ranges at the cost of power con-

sumption[4]. In IoT context, Bluetooth is used to enable communication between low-powered sensors and fog gateways. Bluetooth can only connect 8 devices at a time using a star network topology [Reiter 2014]. The time taken to connect two Bluetooth devices is less than 6 seconds.

Bluetooth Low Energy (BLE) / Bluetooth Smart: This is a variation of traditional Bluetooth specifically designed to operate under low energy consumption [Heydon and Hunn 2012]. It can communicate within 100m at 1 Mbps using less than 10mA. Further, in contrast to Bluetooth, it does not have limitations on number of devices to be connected. BLE is widely used in developing beacons that are being used for indoor localizations and related IoT applications [Nilsson et al. 2016]. The time taken to connect two blueetooth devices is less than 0.006 seconds [Mohandas et al. 1060].

Zigbee / ZigBee PRO: ZigBee is a protocol suite that covers network, transport, and application layers. ZigBee physical and link layer support is built on IEEE 802.15.4 specification.It is one of the most popular, low-cost, low-throughput, and low-power wireless mesh networking standards [Reiter 2014]. In mesh networks, data moves from one node to another in different directions until it reaches the destination. ZigBee can be configured to operate with very long sleep intervals. It can also be configured to have very low duty cycles. [Alliance 2008]. ZigBee can connect 65,536 devices to a network. ZigBee PRO is an enhanced version of ZigBee. In order to connect ZigBee devices to Internet, application level gateway is required. This particular gateway needs to connect to the ZigBee network as a node and it also needs connect to the Internet using either Ethernet or Wi-Fi. Additionally, the gateway also needs to support TCP/IP communication so it can transfer data from Zigbee network to the Internet.

ANT: This is an proprietary protocol designed for low bit-rate and low power sensor networks. Conceptually, ANT[5] is similar to both ZigBee and BLE [Dementyev et al. 2013]. ANT defines a wireless communication protocol stack. In an ANT network, nodes can play dual roles where they can be either masters or slaves. As a result, each node can transmit, receive or do both in order to enable routing data between nodes within the network [Buratti et al. 2009]. One of the primary application of ANT is for fitness and sport related IoT devices. ANT can have long sleep periods during which power is consumed in micro amps. ANT can connect 65,533 devices to a network. ANT supports point-to-point, star, tree, and mesh topologies.

Z-Wave: This is a reliable, low-latency wireless communication certification specifically designed to facilitate machine to machine communication in order to support smart home applications [Z-Wave 2015]. The technology is designed to run on low powered battery operated devices devices. Each Z-Wave network can have up to 232 nodes per controller node. Controllers nodes and slave devices (nodes) are the two types of nodes that each Z-Wave network has. Z-Wave operates on mesh network topologies.

6LowPAN: 6LoWPAN is an acronym of *"IPv6 over Low power, low throughput Wireless Personal Area Networks"* [Shelby and Bormann 2009]. This protocol aims at enabling IP based communication for small low powered devices with limited processing capabilities. IoT is one of the main application domains for which 6LoWPAN has been designed. The 6LoWPAN standard only defines an efficient adaptation layer between the network layer and the data link layer as shown in Figure 17. 6LoWPAN needs an connectivity to the Internet in some way (e.g., Ethernet or Wi-Fi gateway). 6LoWPAN is built on the 802.15.4. Therefore, it has the advantages such as *"mesh network topol-*

---

[4]Device class 1 consumes 100 mA and intended transmission range is 100m. Device class 2 consumes 2.5 mA and intended transmission range is around 10m. Device class 3 consumes 1 mA and intended transmission range is less than 10m

[5]https://www.thisisant.com/developer/ant-plus/ant-antplus-defined

ogy, large network size, reliable communication and low power consumption" [Ma and Luo 2008] as well as advantages of IP-based communication.

Thread: Thread is a network protocol that is IPv6-based, royalty-free protocol for smart home devices to communicate over a network [Thread Group 2015]. It is designed to compete with Z-Wave and Zigbee. Thread runs over 6LoWPAN, which in turn uses the IEEE 802.15.4 wireless protocol. As a result, Thread runs on mesh network topologies and can handle up to 250 nodes.

WiMax: Worldwide Interoperability for Microwave Access is a set of standards for wireless communications [Nuaymi 2007]. Roughly, WiMax can be considered as a a standardized wireless version of Ethernet. It focuses on enabling broadband access to users. WiMax is an interoperable implementation of the IEEE 802.16. Compared to Wi-Fi, WiMax provides higher speeds (1 Gbps) over greater distances (3 Km) and for a greater number of users. WiMax is a competitor of Long-Term Evolution (LTE).

Cellular (GSM / HSPA): High Speed Packet Access (HSPA) protocol enables long range support for IoT applications. It sends data over the exiting cellular networks. Data communications will cost in terms of both money and power. Cellular can communicate over 35km using GSM and 200km over HSPA. Cellular also has the capability to transfer data at different speeds based on the technology such as *"35-170kbps (GPRS), 120-384kbps (EDGE), 384Kbps-2Mbps (UMTS), 600kbps-10Mbps (HSPA), 3-10Mbps (LTE)"* [RS Components 2015].

LPWAN (LoRaWAN): Long Power Wide Area Network (LPWAN) [LinkLabs 2016] is a low powered, low cost, low bit rate protocol design for two way secure communication in the IoT domain. LPWAN allows battery-operated sensors to communicate over long distances. LPWAN gateways deployed in a building or tower can connect to sensors more than miles away (2-5km (urban environment), 15km (suburban environment)). It can also penetrate through water, underground, or in basements (up to 50 meters). *Star-of-stars* topology is being typically used by LPWAN networks. Moreover, LPWAN uses gateways to bridge the communication between Internet and leaf nodes [Vangelista et al. 2015]. Data can be communicated at 0.3-50 kbps [LinkLabs 2016]. Some competing LPWAN protocols proposed are "Sigfox (30-50km rural environments, 3-10km urban environments, up to 10-1000bps), Neul (10km range, up to 100kbps)" [RS Components 2015], and Weightless [LinkLabs 2016].

RFID: Radio-frequency identification [Want 2006] uses electromagnetic fields to transfer data. The tags contain electronically stored information. RFID systems can be categorized based on the types of tags and readers. Passive Reader Active Tag (PRAT) comprises a passive reader and active tags where the reader receives radio signals from active tags. Active Reader Passive Tag (ARPT) comprises an active reader and passive tags. The active reader transmits the interrogator signals and read the data stored in the passive tag. Active Reader Active Tag (ARAT) has an active reader and active tags. Active RFID can perform communication over 100m and Passive RFID is normally limited within around 100cm. RFID tags come in different sizes and look like stickers, cards, and so on. RFID tags are widely used in IoT applications specially to enrich non-electronic dump objects. RFIDs are commonly used for tasks such as tracking (persons, animals, goods), management of asserts, contact-less payment, travel cards (e.g. buses, trains, tubes), and automated toll collection.

NFC: Near field communication [Coskun et al. 2013] is a communication protocol that enables communication between devices within 10cm. Similar competing technologies are bar codes and LF passive RFID tags. NFC supports a peer to peer communication model. NFC devices can work in three modes, namely, NFC card emulation (for smart card based payments), NFC reader/writer (for smart tag based smart posters), and NFC peer-to-peer (for machine to machine communication). Typically, NFC tags are passive data stores that can contain between 96 and 8,192 bytes. NFCs

Table I. Performance Comparison Between HTTP and MQTT [Dutta 2013]

Characterisitcs		3G		Wi-Fi	
		HTTP	MQTT	HTTP	MQTT
Received Messages	Messages / Hour	1,708	160,278	3,628	263,314
	Percent Battery / Hour	18.43%	16.13%	3.45%	4.23%
	Percent Battery / Messages	0.01709	0.00010	0.00095	0.00002
	Messages Received	240/1024	1024/1024	524/1024	1024/1024
Send Messages	Messages / Hour	1,926	21,685	5,229	23,184
	Percent Battery / Hour	18.796%	17.806%	5.446%	3.66%
	Percent Battery / Messages	0.00975	0.00082	0.00104	0.00016

The tests were done by sending and receiving 1024 messages of 1 byte each.

are read-only in normal use. NFC standards cover communication protocols and data exchange formats. These standards are generally based on existing RFID standards. NFC is not limited to tag based communication. NFC is also widely used to enable communication between two smart devices, such as between smart phones and washing machines [Want 2011].

4.4. Multi-Protocol Support: Application Level

In addition to the network communication level protocols discussed earlier, a number of application level protocols heva been proposed and developed to support different types of applications. Primarily, in smart city applications, there are three common models of communication, namely, Device-to-Device (D2D), Device-to-Server (D2S), and Server-to-Server (S2S) [Buyya and Vahid Dastjerdi 2016]. At the moment, smart city domain does not have a single widely accepted protocol supporting its wide range of requirements. Some of the requirements are *"1) broadcasting information from one to many, 2) listening for events whenever they may happen, 3) distributing small packets of data in huge volumes, 4) pushing information over unreliable networks, 5) high sensitivity to volume (cost) of data being transmitted, 6) high sensitivity to power consumption (battery-powered devices), 7) high sensitivity to responsiveness (i.e., near real-time delivery of information), 8) security and privacy, and 9) scalability"* [Dutta 2013]. From here on, we review several popular application level protocols used by smart city applications.

HTTP: Hypertext Transfer Protocol (HTTP) [Fielding et al. 1999] is an application layer protocol designed for distributed hypertext documents and media. Hypertext is structured text. HTTP is a D2S protocol where the device makes requests and the server responds by returning requester hypertext. In Table I, we present a performance evaluation between HTTP and MQTT protocol [Stanford-clark and Truong 2008; Dutta 2013]. MQTT protocol, which we discuss later in this section, is a lightweight protocol specifically designed for IoT needs. MQTT is one of the potential candidates to be used in smart city applications. According to the results, MQTT outperforms HTTP where HTTP uses more battery, is less reliable, and a lot slower than MQTT. The main reason to present this table is to show that some of the widely used application protocols in the Internet domain are not suitable for the IoT domain.

XMPP: Extensible Messaging and Presence Protocol (XMPP) [Saint-Andre 2011] is a message oriented protocol designed for (near) real-time communication. It uses XML to format and model its data. This protocol is widely used for multi-party communication (e.g., voice and video communication) [Schneider 2013]. XMPP is originally designed and developed by Jabber open-source community. XMPP is an open standard approved by IETF [Saint-Andre 2011]. In order to support secure communication, secure authentication (SASL) and encryption (TLS) have been built into the core XMPP. XMPP is scalable even for 100K nodes. XMPP has much higher overhead compared to MQTT.

MQTT: Message Queue Telemetry Transport (MQTT) [Stanford-clark and Truong 2008] is a D2S protocol which focuses on telemetry, or remote monitoring. MQTT designed to acquire data from large number of sources (e.g., ICOs) and communicate them to IoT sensing infrastructure (e.g., IoT cloud platforms). MQTT is initially developed by IBM [Lampkin et al. 2012] for satellite communications with oil-field equipment [Hunkeler et al. 2008]. It is important to note that MQTT is an OASIS open-standard [OASIS 2014]. MQTT is a lightweight protocol. Its header packet size is 2 bytes. Further, client libraries also have low foot print (e.g., C# M2Mqtt library sized at 30KB). In IoT context, MQTT is typically used to collect data from a large number of devices that need to be monitored through the cloud. MQTT is not designed to be fast or real-time. A *hub-and-spoke* is the commonly used architectural pattern when using MQTT [Schneider 2013]. MQTT is designed to operate using a 'publish/subscribe' model. As a result, it requires a message broker to manage and route messages between connected ICOs [Thangavel et al. 2014]. Through the 'publish/subscribe' broker, MQTT enables efficient many-to-many communication. MQTT uses TCP as it is reliable, ordered and error-checked [Thangavel et al. 2014]. MQTT uses Secure Sockets Layer (SSL) and Transport Layer Security(TLS) [Thomas 2000] for security by having encrypted payload [Singh et al. 2015]. Having said that, MQTT is a mature and stable protocol compared to CoAP [Shelby et al. 2014]. MQTT is a binary protocol that performs asynchronous communication.

CoAP: Constrained Application Protocol (CoAP) [Shelby et al. 2014] is an open standard application level protocol specifically designed to enable communication between resource constrained devices [Bormann et al. 2012]. IETF's [Shelby et al. 2014] sample node specification is 8-bit micro-controller with very less amounts of memory that runs on IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs) with throughput of 10s of kbps. CoAP header packet size is 4 bytes [Shelby et al. 2014]. CoAP is a client-server protocol where it supports one-to-one (machine-to-machine) communication. However, CoAP also supports multi-cast communication [Rahman and Dijk 2014]. CoAP is HTTP-like protocol specially similar to how RESTful communication works. Interoperability with HTTP / RESTful is supported through proxies e.g., simple middleware services) [Castellani et al. 2011]. Resource discovery support is built into CoAP. CoAP uses UDP in transport layer which is less reliable than TCP. As a result of being run over UDP (i.e., connectionless datagrams), CoAP's has faster wake-up and transmit cycles. Further, CoAP also has smaller packets with less overheads. SSL/TLS [Thomas 2000] based security is not possible in CoAP as it runs on UDP, not on TCP. However, CoAP uses Datagram Transport Layer Security (DTLS) [Raza et al. 2012] to provide security similar to TLS using encryption based security techniques [Raza et al. 2013].

AMQP: Advanced Message Queuing Protocol (AMQP) [Vinoski 2006] is an application layer, message oriented protocol that follows open standards. AMQP is an OASIS standard. Further, AMQP is secure and reliable protocol that supports queuing and routing. AMQP can work in both point-to-point and publish-and-subscribe settings. AMQP has a packet size of 60 bytes and security is based on SSL/TLS [Thomas 2000]. AMQP aims not to lose messages. It uses TCP to assure reliability. AMQP is mostly used in business messaging where it was initially being developed to support banking applications at JPMorgan Chase in London, UK [O'Hara 2007]. In the IoT applications, AMQP is useful for the control plane or server-based analysis functions [Schneider 2013]. AMQP is similar to MQTT from the functional perspective. However, MQTT is designed as a low-overhead, simple to implement protocol that can be run on many small, relatively dumb devices in order to send small messages on low-bandwidth networks. In contrast, AMQP is a very comprehensive protocol designed to support a large number of message queuing based use cases. AMQP uses a buffer-orientated approach

enabling high-performance at the cost of larger foot print. For example, MQTT does not support transactions whereas AMQP does support [Team 2013].

IoTivity: IoTivity [Linux Foundation 2016b] is not necessary a protocol, but an open source framework that aims to act as a middleware where it offers four primary functionalities, namely, 1) devices and resources discovery in proximity and remotely, 2) data transmission based on messaging and streaming models, 3) data management through collection, storage and analysis of data from various resources, and 4) device management by configuring, provisioning and diagnostics of devices. IoTivity is a project hosted by Linux Foundation and sponsored by Open Interconnect Consortium [Linux Foundation 2016b]. The objective is to design an extensible and robust architecture for smart and thin ICOs, specially for IoT applications. It aims at enabling seamless device-to-device connectivity. IoTivity is primarily targeting device with platforms such as Ubuntu (Linux), Android, Arduino, and Tizen. It is expected to support multiple protocols where it currently supports CoAP and MQTT. Therefore, CoAP supports CoAP service discovery. IoTivity is currently using UDP for transports and JSON payload is supported using CBOR serialization (cbor.io). IoTivity is a RESTful API and supports point-to-point and point-to-multipoint network topologies. IoTivity uses JSON to model data and RAML (raml.org) to model interactions. It also provides security via DTLS [Raza et al. 2012] link-layer using technologies such as ECC, AES, X509 [Delfs and Knebl 2015].

HyperCat: HyperCat [Hypercat Consortium 2016] is a design description framework that allows to expose IoT assets over the web. It is built on top of web standards, e.g., HTTPS, REST/HATEOAS, JSON. HyperCat provides standard mechanisms to semantically annotate its resources and related APIs. HyperCat is designed to be an open standard. It is also a JSON-based hypermedia catalogue format with very low footprint. HyperCat is capable of making URIs available and discoverable to outside applications or services. In HyperCat catalogues, there are not any limitations on how many URIs can be exposed. Further, unlimited RDF-like triple statements can be used to describe each URI (i.e. resource). At minimum, developers need to provide an URL so the clients can access the catalogue using an HTTP GET request. HyperCat does not intend to standardize how IoT resources are accessed, instead it provides a standard mechanism to describe how each IoT resource can be accessed. That means HyperCat provides a common standard to describe APIs and how they can be used by an external entity (e.g., applications). In this way, developers can develop their IoT solutions or devices in anyway they like and use Hypercat specification to describe how to interact with their solutions using a common specification. HyperCat is a Web framework so HTTPS and other Web security mechanisms are supported.

AllJoyn: AllJoyn [Linux Foundation 2016a] is an open source software framework that allows devices to communicate with each other in close proximity. At a high level, both IoTivity [Linux Foundation 2016b] and AllJoyn [Linux Foundation 2016a] aim to achieve same goals, namely, standardization of wired protocols, standardization of schema definitions, standardization of data models, transport and OS independence, collaborative development, open source and freely available, and proximal discovery. However, proximity is an important feature in AllJoyn than IoTivity. AllJoyn is a full stack that consists its own application protocol and several other features. This framework offers an easy way for IoT devices to advertise their services and capabilities. Nearby devices can be connected to each other and discover each other's capabilities. AllJoyn goes further in terms of functionalities it offers and use cases it supports in comparison to application level data exchange protocols such as HyperCat [Hypercat Consortium 2016]. AllJoyn's aims to provide solutions to common IoT challenges as presented in [Linux Foundation 2016a]. AllJoyn currently supports Android, Arduino,

iOS, Linux, OS X, and Windows. AllJoyn uses a binary payload with DBUS[6] serialization. It is designed to run on mesh of stars network topology. AllJoyn uses XML to define schema. AllJoyn follows a publish/subscribe interaction model complemented by Remote Method Invocations. AllJoyn provides application level security via technologies such as ECC, AES, X509 [Delfs and Knebl 2015].

In the following Table II, we summarise the application protocols using fog computing related characteristics. Column 2 denotes on which part of the fog architecture, the given protocol will be mostly used (Device-to-Device, Device-to-Server). We categorised the application level protocols based on their latency in column 3. In column 4, we denote which category of devices will use a given protocols for their communication (Refer Figure 1). At the beginning of Section 4.4, we presented nine requirements that an ideal IoT application level protocol should support. We map those requirements to each protocol in column 5.

Table II. Comparision of IoT Application Level Protocols

Protocol	Usage	Latency	Supported Device Category	Supported Requirements
HTTP(S)	D2S, S2S	High	3,4,5,6	8,9
XMPP	D2S	Low	4,5,6	1,2,4,8,9
MQTT	D2S	Low	1,2,3,4,5,6	2,3,4,5,6,7,8,9
CoAP	D2D, D2S	Low	1,2,3,4,5,6	3,5,6,7,8,9
AMQP	S2S	High	4,5,6	2,4,8,9
IoTivity	D2D	High	1,2,3,4	8,9
HyperCat	D2D	High	1,2,3,4	8,9
AllJoyn	D2D	High	1,2,3,4	1,8,9

In *Supported Requirements*: 1) broadcasting information from one to many, 2) listening for events whenever they may happen, 3) distributing small packets of data in huge volumes, 4) pushing information over unreliable networks, 5) high sensitivity to volume (cost) of data being transmitted, 6) high sensitivity to power consumption (battery-powered devices), 7) high sensitivity to responsiveness (i.e., near real-time delivery of information), 8) security and privacy, and 9) scalability

4.5. Mobility

Mobility is an important aspect in the fog computing domain, specially in many smart city applications [Yannuzzi et al. 2014]. We have discussed a number of smart city applications that will be benefited by mobility in Section 3. In some applications, ICOs at the edge do not require to maintain communication with gateway devices all the time. Due to the fact that gateway devices have more sophisticated capabilities, they are comparatively expensive. Therefore it would be a waste to deploy unnecessary gateway devices [Chirila et al. 2016]. Mobility allows small number of gateway devices to manage a large number of edge ICOs that are deployed across large geographical areas. In such circumstances, fog gateway devices will move through a path that will allow them to make temporary connections with edge ICOs in order to perform different types of management tasks, such as data acquisition, ICO configuration, evaluating device health statuses and so on. Mobility also highlights the importance of dynamic discovery and configuration. Each time a gateway moves, the discoverer including security procedure needs to be executed repeatedly to make sure ICOs are well maintained [Perera et al. 2014a].

---

[6]https://dbus.freedesktop.org/doc/dbus-specification.html

Discovery in fog computing need to be done efficiently and effectively. The reason is that each ICO will have very limited amount of time to connected to a gateway and perform some data communication before either the ICO itself or the corresponding gateway moves away [Ishino et al. 2015]. The challenge is to find a common ground as soon as possible so hey can initiate thee communication [Perera et al. 2014b]. First, they need to find out which communication technology to use. Specially, in circumstance where a gateway may have multiple communication capabilities (e.g. WiFi, Bluetooth, Zigbee) and the ICO may only have one. For energy reasons, gateway may not want to keep all its communication technologies 'ON' all the time. So the challenge is to efficiently and effectively find out, which technology to use on a given location and time period while on the move. Next, they need to find out which application level protocol they would want to use by consider the capabilities of both ICOs and gateway devices. As discussed in Section 4.4, there are many frameworks that has been proposed to make discovery in IoT much easier and efficient (e.g. HyperCat, AllJoyn, IoTivity). Predictive and opportunistic models will be useful in handling these challenges [Pozza et al. 2015; Higuchi et al. 2014].

Fog gateways can also be in the form of drones [Sathiaseelan et al. 2016]. Such drone based fog gateways are useful in many smart city applications including disaster recovery, wild fire monitoring and so on. In Phenonet [Commonwealth Scientific and Industrial Research Organisation (CSIRO), Australia 2011], mobile robots are used as gateways.

4.6. General Data Considerations

Cosnidering the exlpoitations so far in the paper, it should be very clear that data is a key aspect of Fog computing systems; the next few subsections will be concerned with different aspects of data analytics and gathering but it is sensible to consider some generic Data issues first. In this section we will briefly explore management of data and the different kinds of data that can exist based on their origin.

Managing the vast amount of data, which can exist in different parts of the system is a very important issue, and one that requires further research. However, Fog computing presents a unique opportunity to move from data to information very quickly, with information that is of long-term value then being send and stored in the cloud (a problem that is reasonably well understood) and information that is of short-term value being used to act immediately and then being discarded. Information of the latter kind could be room temperature readings that are used to activate cooloing or heating systems. The challenge which requires further research in this space lies in determining , ideally automatically, the value proposition of specific data items and making the decision to not keep certain data. The section on context awareness discusses these aspects, and data analytics does of course have the purpose to convert data to information.

A further aspect of data management is focused around the idea that we have different data in the system: data can be sensed or meassured directly; data could exist because it was communicated from a nother part of the system or data could be derived by some analytics or processing approach. This data is often combined with stored data to make decisions or enrich analytics. Ultimately the question of managament is less a question of where the data stems from, but rather on of how realiable and current the data is. For example a temperature reading gathered from a trusted high quality sensor is likely very good, while a derived conclusion of the wereabouts of a person based on the last registration to a cell phone tower from a while ago is probably not too reliable [Cai and Zhu 2015]. A whole field of study has emerged that is considering quality of data (QoD) and is generally cosnidered as 'Data Science'.

4.7. Context Discovery and Awareness

This is an important feature that needs to be supported in fog computing platforms as shown in Figure 18. Specially, the ability to discover context is one of the primary advantages in fog computing over cloud computing. Being close to the edge nodes (ICOs), fog gateway devices have more chances and ability to infer context information such as location, environmental conditions, nearby devices and their capabilities, comparing different ICOs and identifying any malfunctioning ICOs, and so on. *"Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves"* [Dey 2001]. Further, context information is [Perera et al. 2014a] typically useful in inferring knowledge about what is exactly happening in the field. For example, identifying a malfunctioning ICO by comparing data from nearby ICOs is critical, so the cloud can ignore the data items captured by the malfunctioning ICOs. Context data can provide information about data quality which is also has a direct impact on the fused results [Perera et al. 2014a]. Context data can be used to develop efficient and effective data collection plans specially, when multiple ICOs available near by that offer similar information.

In data analytics, inaccurate input data is likely to produce inaccurate results. *"A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task"* [Dey 2001]. Ideally, fog gateways should be context-aware and intelligent enough so it automatically configures itself to semantically annotate the sensor data appropriately based on the location it is being deployed and capabilities it has. Another context-aware requirement is co-operative and opportunistic sensing. ICOs and fog gateways should be able to balance their workload with neighbouring gateways and ICOs in order to make sure no resources are wasted due to redundant sensing [Aazam and Huh 2015b]. Context-aware data communication is an important part of connected vehicles applications where data resources are constrained [Truong et al. 2015]. Policy driven resources management in fog computing is discussed in detail in [Dsouza et al. 2014]. Context data also has a significant impact on data collection and fusion strategies [De Paola et al. 2016; Jiang et al. 2014]. For example, as explained in Section 3.1, IoT application may decide to change the sampling rates of the data collection depending on the context information (e.g., temperature). Further, data aggregation time frames may also get altered based on context data.
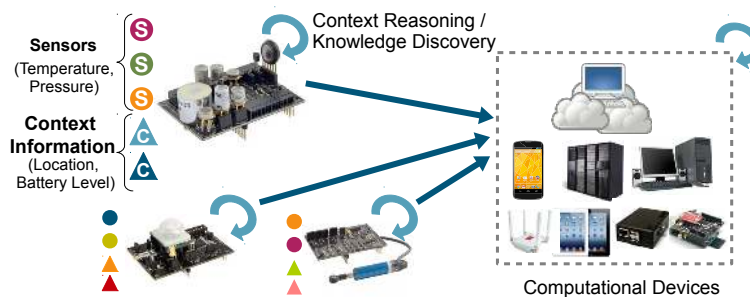

Fig. 18.   Context Discovery

4.8. Semantic Annotation

The process of *"associating metadata with resources (audio, video, structured text, unstructured text, web pages, images, etc.) is called annotation"* [Cardoso and Sheth 2006]. Further, semantic annotation can be identified as method of using semantic metadata to annotate a given resource [Cardoso and Sheth 2006]. Metadata is *"data that provides information about other data"* [National Information Standards Organization 2004]. There are three main types of metadata, namely 1) structural, 2) descriptive, and 3) administrative metadata. Structural metadata provides information about who the data is being structure and stored (e.g. whether the data is structure based on the location they are being generated or whether data is structured based on the type of the sensor). Descriptive metadata provides a way to annotate data to help identify and discover a given resource (e.g. when is the data generated, who is the originator, and so on). Administrative metadata provides information about managing a given resource (e.g., how a particular resources being created, managed, who have access to a particular resource and so on) [National Information Standards Organization 2004].

Semantic annotation is strongly linked to the context discovery we discussed in Section 4.7. Typically, in smart city applications, context information is semantically annotated to the data. We have discussed what context information is in [Perera et al. 2014a] in detail. Any piece of context information becomes an potential candidate to be used to annotate sensor data [Wei and Barnaghi 2009]. semantic annotation could be inferred from the sensor data or entered by users. Typically, ICOs are resource constrained devices and sometimes they may not have sufficient resources or knowledge to annotate data with semantics. Therefore, responsibility of semantic annotation could be assigned to fog gateways [European Research Cluster on the Internet of Things 2015]

4.9. Data Analytics

Fog gateway devices are expected to have limited computational capabilities. In such circumstances, one of requirements is to have module based data analytical components that can be remotely pushed into fog devices on demand. As these fog gateway devices need to be designed to support a large variety of smart city applications, it is wasteful to install a large number of data analytical modules. Ideally, these devices should have only installed with the most common data analytical capabilities, such as average, ignoring outliers, etc. Other types of data analytics should be installed ondemand. It is also important to provide necessary tools that can be used as templates to build such data analytical components. It is also ideal to support different runtime paradigms that are commonly used to build data analytics components such as Java, Python, R, Matlab, and so on [Bordawekar et al. 2015]. Each of these modules would be black boxes running on an individual sandboxes where they intake certain types of inputs and generate certain types of outputs. Some type of permission and validation procedures will be required to ensure privacy and security. It is important to note that fog gateway devices are also resource constrained devices (better off compared to edge ICOs) [Aazam and Huh 2014]. Therefore, some types of data analytics may not be able to perform on them due to computational complexities and resource limitations [Bordawekar et al. 2015; Aazam and Huh 2015b]. Dimension reduction techniques can be used to reduce the amount data that need to be communicated to the cloud for further analysis [Fodor 2002].

Aazam and Huh [Aazam and Huh 2015b] have presented an approach to estimate resource consumption in a fog computing IoT application. For example, data analytics could vary from identifying user activity by fusing accelerometer data to identify sentiment by analysing an image captured by a fog gateway located in a bus station. It

is important to note that most the data in IoT are captured as data streaming. Therefore, it is critical to support data stream analytics capabilities in both go gateways and cloud companion platforms [Belli et al. 2015; Aberer et al. 2007].

Data stream analytics is an important aspect in IoT domain as well as in fog computing. In IoT, ICOs are more like to stream data into IoT application than sending data as batches. Though an IoT application may received large number of data streams from large number of ICOs, each gateway will receive only few data stream. Therefore, each gateway is only need to deal with few data streams [Puschmann et al. 2016]. Data stream analytics are discussed in detail in [Liu et al. 2014; Singh et al. 2016; Singh and Reddy 2015]. The challenge is to find out how to bring data stream analytics towards fog gateway from the cloud, in order to reduce network communication and energy consumptions.

4.10. Security and Privacy

This is an important non-functional requirement in smart city applications [Kulkarni et al. 2012]. We can identify four broader categories under this theme namely, 1) privacy, 2) authenticity, 3) confidentiality, and 4) integrity. Privacy safeguards that only the approved ICOs and gateways are a part of the network. Authentication is a major challenge in fog computing [Stojmenovic and Wen 2014]. Authenticity aims to verify genuineness of the data sender. For example, public-Key crypto systems [Rivest et al. 1978] are used to achieve this. Confidentiality make sure that only the proposed destination can read the data. AES 128 and AES 256 [Daemen and Rijmen 2002] or similar techniques are used to assure the confidentiality. Integrity safeguards that the original data is not being altered during the transmission and the information received by the destination is same as the information send by the originator. For example, hash algorithms such as MD5 and SHA [Delfs and Knebl 2015] can be used to generate checksum of the message and to ensure that the integrity of the message. In fog computing paradigms, we can identify several different data communication patterns that need to be secure: 1) ICO to ICO , 2) ICO to Gateway, 3) ICO to Cloud, 4) Gateway to Cloud, 5) Cloud to ICO/Gateway [Gascon 2015]. Let us briefly introduce each of these patterns, which can be secured using different techniques.

1) ICO to ICO: In order to avoid external ICOs (including malicious ICOs) being connected to a network and observing the data communicated back and forth, ICOs can use link layer based encryption techniques. AES 128 [Daemen and Rijmen 2002] (Symmetric Encryption) is the most widely used algorithm to address this issue. Specialized hardware can be used to manage the encryptions. In this communication pattern, each network will have a shared common key pre-shared by all the ICOs [Gascon 2015].

2) ICO to Gateway: Networking and application layer based symmetric encryption can be used to secure this type of communication. In ICO to ICO communication, each node has its own private key to ensure authenticity. Gateways are expected to have all the encryption key corresponds to each ICO. Gateways need them to decrypt the data sent by ICOs. Data generated by one ICO may hop through multiple neighboring ICOs to reach the gateway. However, intermediary ICOs will not be able to read the packet as it does not have the private key of the generator. With the help of random internal seed and a sequence number, fog gateways can verify the integrity of data using checksums [Gascon 2015].

AES 256 [Daemen and Rijmen 2002] (Symmetric Encryption) is the most widely used algorithm to secure this type of communication pattern. Man-in-the-middle attacks are the most common type of security challenges in fog networks [Stojmenovic and Wen 2014]. Such attacks are hard to detect specially in the scenarios we discussed in Section 3.

3) ICO to Cloud: At times, it is necessary for ICOs to directly connect to the cloud platform. In this pattern, without storing the keys in the gateway, the keys are stored in the cloud, so the cloud can decry the data sent by a known ICO. AES 256 [Daemen and Rijmen 2002] (Symmetric Encryption) is the most widely used algorithm to secure this type of communication pattern and typically software libraries help to handle the encryption.

4) Gateway to Cloud:. HyperText Transfer Protocol over SSL/TLS (HTTPS) is commonly used to secure this type of communication. Typically, IoT cloud platforms are Web servers and fog gateways may use HTTPS to communicate with the server.

5) Cloud to ICO/Gateway: This type of communication pattern is primarily required to initiate secure connections between ICOs, gateways, and the cloud. To initiate AES 256, it is essential to use a shared key between each origin node (i.e. sensor node) and the gateway or the cloud destination server. Public-Key cryptosystems [Rivest et al. 1978] are used to achieve this.

Most IoT platforms have the implicit assumption that *"You feed your data to our platform, we support the data analytics, event detection, actuation and other functionalities"*. However, as these platforms receive more and more personal data and once these platforms start sharing data between different parties, privacy becomes a serious concern and a great challenge to address [Perera et al. 2015b].

Sending all the collected data all the time to the cloud consumes more data communication bandwidth and energy. Another major issue is privacy [Kulkarni et al. 2012]. Raw data collection can always lead to privacy violations at later stages during the data life cycle. Therefore, an important principle in protecting user privacy is to collect only the amount of data that is sufficient to achieve the task at hand [Gaura et al. 2013]. Data collection and analytics is a double edged sword. Data gathering and analytics help us to derive useful new knowledge. However, such discovery of new knowledge could also lead to violation of someone's privacy. Therefore, data analytics in the IoT domain need to be carefully and ethically monitored to ensure no privacy violations would occur during the process. Privacy issues become more critical in IoT applications in smart homes and smart wearable domains.

4.11. Cloud Companion Support

As we mentioned earlier, fog computing is not a solution fitting for all scenarios, neither can it be used as a standalone solution. Fog computing generally goes hand-in-hand with cloud computing [Yannuzzi et al. 2014]. As a result, it is important for fog gateways to have the built in ability to communicate with cloud IoT platforms. Ideally, fog computing should have a pluggable architecture so adding support to a new cloud platform should not be difficult. Standardized interfaces opened-up through HTTP REST APIs (or similar) would enable interoperability. Use of semantic annotation or adoption of techniques such as HyperCat [Hypercat Consortium 2016] would also enable such interoperability. Cloud platforms and fog gateways should be able to exchange commands (i.e., the control plane) as well as data (i.e., the data plane). Some common cloud computing platforms are discussed in [Perera et al. 2015a]. For example, industrial IoT cloud platform IBM Blumix provides a way to covert *Raspberry Pi* devices into fog gateways through an open source framework called *Node-RED* (nodered.org).

5. EXISTING RESEARCH EFFORTS AND TRENDS

Based on the identified characteristics and common features in the previous section, in this section, we select and compare more than 30 representative existing research efforts in Fog computing. All these research efforts focus on at least two of the common features of Fog computing, as shown in Table 5.

It is easy to observe that Cloud Companion Support and Data Analytics are the most popular features in these Fog computing research efforts. This indicates that many applications employ Fog computing to do small data analytics tasks but meanwhile, still rely on Cloud computing platforms to perform large tasks, which normally cannot be done on Fog devices. There also exist a few research efforts where Fog devices will take care of almost all the data analytics tasks, such as Gazis et al. [Gazis et al. 2015], Kulkarni et al. [Kulkarni et al. 2012], Preden et al. [Preden et al. 2015], and Oueis et al. [Oueis et al. 2015]. The reasons for these include real-time local decision making [Gazis et al. 2015; Preden et al. 2015], privacy [Kulkarni et al. 2012], and load balancing [Gazis et al. 2015; Oueis et al. 2015].

Following the above two most popular features, Multi-Protocol Support at Communication Level, Mobility, and Security and Privacy are also popular features in Fog computing applications. Regarding Multi-Protocol Support at Communication Level, since computing happens at the edge of a network, supporting communication to a variety of devices is of great importance. Further, to make Fog computing more useful, mobility support is also critical. Some research effort even emphasizes that, to this end, a distributed directory system is necessary and mobility techniques support, such as LISP protocol support, should be provided [Jiang Zhu et al. 2013]. Finally, Security and Privacy attract much attention. The main reason for this is that in Fog computing, the generated data is very sensitive to locations, identities, and times, which all pose great risk of information and privacy disclosure.

According to the table, the least implemented common features of Fog computing are Multi-Protocol Support at Application Level, Context Discovery and Awareness, and Semantic Annotation. It seems that many research efforts are happy with supporting only one application protocol, which is sufficient for most of the time. There are also not many applications requiring Context Discovery and Awareness, where the main reason could be that the application scenarios are very clear and contexts are quite certain. In terms of Semantic Annotation, for now it receives least attention in the listed research efforts shown in Table III. This may be due to the fact that Fog computing is still at its early stage and more efforts are devoted to other seemingly more important features.

Among these research efforts, Gia et al. [Gia et al. 2015] provide an interesting case study on Fog computing in Healthcare Internet of Things. When applying the Fog computing technology, smart gateways are deployed with embedded data mining, distributed storage, and notification services. As a case study, Electrocardiogram (ECG) feature extraction is examined. Specifically, ECG signals are transmitted to smart gateways, where feature extractions are performed. In this way, it is possible to provide real-time analytics and offer low-latency response. It is demonstrated that such Fog computing approach can help to save more than 90% bandwidth compared with traditional Cloud computing approaches with lower latency.

Regarding security, Stolfo et al. [Stolfo et al. 2012] propose an cloud-based approach that can be used to mitigate attacks, that are focused on data theft, by exploiting the advantages of Fog computing. The main idea is to implement two additional security features, including *User Behavior Profiling* and *Decoys*, on fog gateway devices [Stolfo et al. 2012]. Firstly, User Behavior Profiling can be achieved by training one-class support vector machines, where building a classifier without sharing data from different users is feasible [Stolfo et al. 2012]. Meanwhile, Decoys feature refers to placing different types of important-looking documents in some highly conspicuous locations by the legitimate user, in order to detect suspicious access. Since these two new features are built on top of existing cloud security features, better security can be achieved.

Similarly, in another scenario, Dsouza et al. [Dsouza et al. 2014] investigates Fog computing in supporting secure, private and safe real-time smart transportation systems, which should accommodate dynamic traffic changes and alert potential conflicts

Table III. Evaluation of Existing Research Efforts

	DDIO	DCDM	MPS-C	MPS-A	Mob	CDA	SA	DA	SP	CCS
[Farris et al. 2016]	✓				✓					✓
[Tang et al. 2015]								✓		✓
[Belli et al. 2015]		✓		✓				✓		✓
[Gazis et al. 2015]			✓					✓		
[Abdullahi et al. 2015]	✓		✓							✓
[Gu et al. 2015]	✓	✓			✓					
[Truong et al. 2015]		✓	✓							✓
[Stolfo et al. 2012]					✓				✓	✓
[Jiang Zhu et al. 2013]		✓	✓		✓			✓		✓
[Zao et al. 2014]				✓				✓	✓	✓
[Aazam and Huh 2014]								✓	✓	✓
[Kulkarni et al. 2012]								✓	✓	
[Dsouza et al. 2014]					✓	✓		✓	✓	✓
[Aazam and Huh 2015b]	✓	✓	✓					✓		✓
[Preden et al. 2015]	✓	✓			✓	✓		✓		
[Oueis et al. 2015]		✓						✓		
[Cao et al. 2015]								✓		✓
[Su et al. 2015]		✓								✓
[Al Faruque and Vatanparvar 2016]	✓	✓	✓	✓					✓	✓
[Giang et al. 2015]					✓			✓	✓	✓
[Hassan et al. 2015]					✓			✓		✓
[Ismail et al. 2015]		✓								✓
[Gia et al. 2015]			✓			✓		✓		✓
[Li et al. 2015]			✓					✓		✓
[Farris et al. 2015]		✓			✓			✓	✓	✓
[Dubey et al. 2015]			✓					✓	✓	✓
[Nishio et al. 2013]			✓		✓			✓		✓
[Stantchev et al. 2015]			✓						✓	✓
[Sehgal et al. 2015]	✓		✓					✓	✓	✓
[Aazam and Huh 2015a]					✓	✓	✓	✓		✓
[Suciu et al. 2013]			✓		✓			✓	✓	✓
[Bruneo et al. 2016]		✓			✓	✓		✓		✓

DDIO: Dynamic Discovery of Internet Objects, **DCDM:** Dynamic Configuration and Device Management, **MPS-C:** Multi-Protocol Support: Communication Level, **MPS-A:** Multi-Protocol Support: Application Level, **Mob**: Mobility, **CDA:** Context Discovery and Awareness, **SA**: Semantic Annotation, **DA**: Data Analytics, **SP:** Security and Privacy, **CCS:** Cloud Companion Support.

and safety issues to travelers. To achieve this, a robust policy management framework is proposed, which can ensure both interoperability and secure communication in a fog ecosystem.

6. LESSONS LEARNED

In this section, we take each of the major features identified in Section 4 and explain how they would contribute towards building sustainable sensing infrastructures for smart cities. Due to high demand and public pressure towards building smarter living

spaces for people, both government and private sector entities are forced to initiated smart cite programs. Smart cities investments are designed to focus of finding ICT based sustainable solutions to address growing issues [Perera et al. 2014b]. By definition, Smart Cities [Caragliu et al. 2009] have *"six characteristics: smart economy, smart people, smart governance, smart mobility, smart environment and smart living"* [Giffinger et al. 2007]. Sustainability is defined as *"...able to be used without being completely used up or destroyed.."* and *"...able to last or continue for a long time"* (merriam-webster.com/dictionary/sustainability). Sustainability has the same meaning in smart cities context as well.

Smart Cities require to accommodate growing population and their needs with limited resources. Smart Cities also need to make sure the limited resources do not run out [Aazam and Huh 2015b]. The best strategy to achieve this is to use resources in the most efficient and optimum ways. To do this, it is essential to understand how cities and its citizens behave and consume resources (e.g., patterns, practices, norms and so on). All the information that is required to understand cities and their citizens is hidden in IoT data. To discover knowledge and insights from IoT data, we need to collect and analyses them in large scale. Sensing infrastructure is a critical element in collecting and analyzing IoT data. Fog computing brings efficiency and sustainability to the sensing infrastructure as follows.

Dynamic discovery of ICOs help towards building sustainable sensing infrastructure in multiple ways. First, it reduces the deployment time (therefore cost) of sensing infrastructure. Secondly, it allows ICOs to be moved at runtime. For example, a new ICO can be deployed in the field at any time without changing existing infrastructure. Dynamic configuration plays a critical role in sustainability as it allows to reconfigure sensing parameters at run time (e.g., sampling rate, communication frequency, which sensors to activate). These parameters have direct impact on the energy consumption and the life time of the infrastructure, especially when ICOs are battery/solar powered. Efficient and optimum management of these parameters can make sure that a given infrastructure can sustain for a long period of time without recharging or replacement.

Choosing the most appropriate communication protocols is paramount in building sustainable sensing infrastructure. As we have presented in Section 4.3, each protocol has its own strengthens and weaknesses. Some protocols use a far less amount of energy as they are specifically designed to operate using batteries for years. Further, fog gateways may also change the communication protocols they use at run time opportunistically to get the advantage of their surrounding resources. For example, cellular based mobile fog gateways may opportunistically use a Wi-Fi zone to upload the data quickly to the cloud so it can save energy and costs compared to cellular communication. Similarly, selecting the right application level protocols also has a significant impact of he sustainability of the infrastructure. For example, we have presented a comparison between HTTP and MQTT in Table I. It clearly shows how much energy can be saved by using MQTT over HTTP. Therefore, it is critical to think about which application protocol is sufficient to accomplish the needs of a smart city application.

Mobility helps to build smart city applications with less resources. For example, in Section 3, we discussed a number of use cases that involve mobility of ICOs and fog gateways. Mobility helps to cover a large geographical area with fewer ICOs and gateways by reducing the deployment costs significantly. Such reductions can help to build sustainable sensing infrastructure for smart cities.

Basic management of data and assuring quality of data has challenges that are fundamental to Fog computing, but Fog computing also provides great opportunities to allow for storage of valuable information rather than data which might quickly become irrelevant. Semantic annotation allows data to be more meaningful. It reduces the amount of efforts that need to be put in when recovering the meaning out of data. When

sensor data is annotated with context information, it is easy for algorithms to discover knowledge later in the data analytical pipeline. This reduces overall computation costs (i.e., energy requirement throughout the data flow) significantly, which is better for sustainability.

It is vital to apply analytics over data at the right time and in the right place. Data in smart cities flows from ICOs to the cloud. It is always better to apply the analytics to the data at early stages of the pipeline. This reduces the amount of data that needs to be communicated to the cloud and save data communication costs, storage costs, and computational costs. Data analytics typical analyzes comparatively large amounts of data and produces summarized results. This reduction leads to less amounts of data being transferred to the cloud and saving significant costs (e.g., bandwidth, energy, storage).

Security and privacy are two of the most important factors that impact towards sustainability. No sensing infrastructure can survive without them. In order to function without disruption, security and privacy need to be guaranteed. Otherwise, it would affect sustainability in two ways; 1) citizens will oppose the deployments and the usage of sensing infrastructure as it affects their personal security and privacy, and 2) applications that are built around sensing infrastructure will get disrupted and users who depend on those applications will get affected significantly and could lead to losses, disappointments, frustrations, and chaoses.

Context-awareness helps to reduce energy consumption in most of the use case scenarios. It can decide 1) when to start sensing and when to stop, 2) when to use which protocol, 3) when to transmit data, and 4)when to increase or decrease sampling rates using context information. We discussed these aspects earlier in Section 4.7. Further, load balancing and opportunistic sensing will help fog gateways and ICOs to efficiently balance their workload with neighbors and avoid redundant sensing and resource wastage. Intelligent and efficient planning can be used to make sure all neighboring ICOs and gateways do not go off-line at the same time due to energy running out. This allows recharging and replenishment to be taken place without significant disruption. Having said all these advantages in fog computing, almost no sensing infrastructure can survive without cloud platforms. Cloud platforms are scalable and can accommodate the growing needs in smart cities. Cloud based large scale data analytics and knowledge discovery (e.g., prediction) can make all the sensing efforts useful and create values. Such value creation directly contributes towards the long term sustainability of sensing infrastructure.

7. CHALLENGES AND OPPORTUNITIES

So far we have discussed ten common features that are important in a fog computing platform in order to support building sustainable smart cities. It is important to note that these features are research areas that need to be explored further and significantly. We have provided a number of references that readers can use to look further into challenges on each of these areas.

In this section, we highlight the needs of fog platforms that support all the features discussed in this paper. Both IoT and fog computing are comparatively immature fields. Therefore, a large amount of experimentation needs to be done. An ideal fog computing platform should be able to provide a framework that others can use it to test different approaches, techniques, and algorithms. For example, there are many ways to autonomously annotate data with semantics within a fog gateway. It is not possible to develop one universal approach or algorithm to annotate data. Therefore, a fog platform should be built in such a away that anyone can write exertions to support new ways of annotating data. This is similar to what we see today in mobile app markets. In a mobile app market, there are many different apps that are designed to

perform the same task at high level (e.g., TODO List apps). However, each of these applications is designed in a unique way and offers slightly different capabilities. Users can easily uninstall one app and install a new one depending on their requirements at a given time. However, it is important to note that a mobile app does need to follow certain guidelines in order for them to work in a given platform. Similarly, there should be some standardizations so these plug-ins (or extensions) would work seamlessly in the fog platforms, but the implementation should be open for creativity.

Further, such platforms should provide built-in supports for different types of communication and application level protocols. This does not mean that all these protocols should be supported at all time. However, a fog platform should be able to download plug-ins to provide support for these different protocols without requiring much effort. Further, providing support for existing data analytics frameworks is also important. For example, some analytical plug-ins may be written in R, Python, Matlab, Java, C/C++ and so on. All these different approaches need to be supported so the existing code bases can be easily reused to perform data analytics. At high level, it is useful to build an ecosystem with interchangeable plug-ins that support different implementations of the features we discussed.

Due to low computational resources of fog gateways, it is important that these plug-ins can be easily removed to avoid resource wastage when not required in a given fog gateway. Despite we see a large number IoT cloud platforms in the market in both academia (e.g., OpenIoT) and industry (e.g., IBM Bluemix, Microsoft Azure IoT), a fog computing platform that supports all the features we listed in this article are yet to be researched and developed. We believe such fog platforms would be greatly beneficial to the research and industrial communities once available as by then people can easily test new fog computing related approaches, techniques and algorithms.

According to the literature comparison presented in Table III, context discovery and awareness [Perera et al. 2014a] and semantic annotation is fairly ignored. As we discussed in Section 6, these two functionalities has the potential to improve the sustainability of a sensing infrastructure significantly. Challenges in semantic annotation and interoperability in IoT domain are discussed in detail in [European Research Cluster on the Internet of Things 2015].

Distributed intelligence will be very critical in making fog computing platforms successful in building future smart cities. The main reason is that prompt reactions and best possible decisions should be achieved in a timely manner to make future cities smarter, safer and more living-enjoyable. This requires a large amount of research efforts in putting distributed intelligence in place properly across smart things, buildings, fog devices/gateways, and cloud computing infrastructure in a city. This process will involve many important aspects, such as available domain-dependent knowledge/intelligence, combination of business logic, engineering processes and government policies, cost-efficient and computation-efficient and context-/semantics-aware computing models for real-time decision making, etc.

Security is also a critical issue in building future smart cities. This mainly refers to security of fog computing platforms, including potential cyber attacks to smart things, fog devices/gateways, and trust and authentication, network security, and data security, etc. For example, cyber attacks to smart things, fog devices/gateways can dysfunction smart things, fog devices/gateways and pose risks in failure of providing proper services to the city and making wrong decisions in reaction to emergencies and disasters. Failure of ensuring trust and authentication will also put any large-scale fog computing platforms at risk, potentially leading to intentional and accidental misbehavior, criminal activities and so on. Network security is also of great importance since network attacks such as jamming attacks, sniffer attacks and so on can create huge risks in fog computing systems, potentially leading to chaos of the whole fog computing

systems. Data security will also be critical. Sensitive and/or valuable data generated from any fog computing platforms should be kept secure.

8. CONCLUSIONS

Due to the improvements in sensing technology and reduction in costs, sensing capabilities are expected to be integrated into everyday objects around us. There is a natural tendency that smart city applications are being built in a centralized manner. That means all the data collected by sensors are transferred to a cloud node for knowledge discovery. However, this is a very inefficient approach from both computational and communication perspectives. To address this issue, the fog computing paradigm has been proposed. Specially, more and more industrial IoT platform developers, such as Microsoft, IBM, and Intel, are now moving towards utilizing fog gateway devices to perform edge analytics. Fog computing brings sustainability to the smart city applications. In this survey paper, we have analysed and evaluated different types of fog computing and edge analytics research efforts to understand what are the most important functionalities of a fog computing platform. We have also discussed the major trends in this field that were identified during the survey. Our finding clearly highlight the importance of fog computing platforms in order to build sustainable IoT infrastructure for smart cities.

ACKNOWLEDGMENTS

REFERENCES

Mohammad Aazam and Eui Nam Huh. 2014. Fog computing and smart gateway based communication for cloud of things. In *Proceedings - 2014 International Conference on Future Internet of Things and Cloud, FiCloud 2014*. IEEE, 464–470. DOI:http://dx.doi.org/10.1109/FiCloud.2014.83

Mohammad Aazam and Eui Nam Huh. 2015a. E-HAMC: Leveraging Fog computing for emergency alert service. In *2015 IEEE International Conference on Pervasive Computing and Communication Workshops, PerCom Workshops 2015*. IEEE, 518–523. DOI:http://dx.doi.org/10.1109/PERCOMW.2015.7134091

Mohammad Aazam and Eui Nam Huh. 2015b. Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT. In *Proceedings - International Conference on Advanced Information Networking and Applications, AINA*, Vol. 2015-April. IEEE, 687–694. DOI:http://dx.doi.org/10.1109/AINA.2015.254

Ibrahim Abdullahi, Suki Arif, and Suhaidi Hassan. 2015. Ubiquitous shift with information centric network caching using fog computing. In *Advances in Intelligent Systems and Computing*, Vol. 331. Springer International Publishing, 327–335. DOI:http://dx.doi.org/10.1007/978-3-319-13153-5_32

Karl Aberer, Manfred Hauswirth, and Ali Salehi. 2007. Infrastructure for Data Processing in Large-Scale Interconnected Sensor Networks. In *International Conference on Mobile Data Management*. 198–205. DOI:http://dx.doi.org/10.1109/MDM.2007.36

I F Akyildiz and J M Jornet. 2010. The Internet of Nano-Things. *IEEE Wireless Communications* 17, 6 (dec 2010), 58–63. DOI:http://dx.doi.org/10.1109/MWC.2010.5675779

Mohammad Abdullah Al Faruque and Korosh Vatanparvar. 2016. Energy Management-as-a-Service over Fog Computing Platform. *IEEE Internet of Things Journal* 3, 2 (apr 2016), 161–169. DOI:http://dx.doi.org/10.1109/JIOT.2015.2471260

Zigbee Alliance. 2008. Zigbee Specification. *Zigbee Alliance website* (2008), 1–604.

U Alvarado, A Juanicorena, I Adin, B Sedano, I Gutirrez, and J de N. 2012. Energy harvesting technologies for low-power electronics. *Transactions on Emerging Tele communications Technologies* 23, 8 (2012), 728–741. DOI:http://dx.doi.org/10.1002/ett.2529

Alicia Asin and David Gascon. 2012. *50 Sensor Applications for a Smarter World*. Technical Report. Libelium Comunicaciones Distribuidas.

Laura Belli, Simone Cirani, Gianluigi Ferrari, Lorenzo Melegari, and Marco Picone. 2015. A graph-based cloud architecture for big stream real-time applications in the internet of things. In *Communications in Computer and Information Science*, Vol. 508. Springer International Publishing, 91–105. DOI:http://dx.doi.org/10.1007/978-3-319-14886-1_10

SIG Bluetooth. 2005. Specification of the Bluetooth system. *Core, version* 1 (2005), 2005–10.

Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. 2012. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing - MCC '12*. ACM Press, New York, New York, USA, 13. DOI:http://dx.doi.org/10.1145/2342509.2342513

Rajesh Bordawekar, Bob Blainey, and Ruchir Puri. 2015. Analyzing Analytics. *Synthesis Lectures on Computer Architecture* 10, 4 (nov 2015), 1–124. DOI:http://dx.doi.org/10.2200/S00678ED1V01Y201511CAC035

Carsten Bormann, Angelo P. Castellani, and Zach Shelby. 2012. CoAP: An application protocol for billions of tiny internet nodes. *IEEE Internet Computing* 16, 2 (2012), 62–67. DOI:http://dx.doi.org/10.1109/MIC.2012.29

Mike Botts and Alexandre Robin. 2007. *OpenGIS Sensor Model Language (SensorML) Implementation Specification*. Technical Report. Open Geospatial Consortium Inc.

D Bruneo, S Distefano, F Longo, G Merlino, A Puliafito, V D'Amico, M Sapienza, and G Torrisi. 2016. Stack4Things as a fog computing platform for Smart City applications. In *2016 IEEE Conference on Computer Communications Workshops (INFOCOM Workshops)*. 848–853. DOI:http://dx.doi.org/10.1109/INFCOMW.2016.7562195

Chiara Buratti, Andrea Conti, Davide Dardari, and Roberto Verdone. 2009. An overview on wireless sensor networks technology and evolution. (2009). DOI:http://dx.doi.org/10.3390/s90906869

Rajkumar Buyya and Amir Vahid Dastjerdi. 2016. *Internet of things : principles and paradigms*. Morgan Kaufmann. 354 pages.

Li Cai and Yangyong Zhu. 2015. The Challenges of Data Quality and Data Quality Assessment in the Big Data Era. *Data Science Journal* 14, 2 (2015). DOI:http://dx.doi.org/10.5334/dsj-2015-002

Yu Cao, Songqing Chen, Peng Hou, and Donald Brown. 2015. FAST: A fog computing assisted distributed analytics system to monitor fall for stroke mitigation. In *Proceedings of the 2015 IEEE International Conference on Networking, Architecture and Storage, NAS 2015*. IEEE, 2–11. DOI:http://dx.doi.org/10.1109/NAS.2015.7255196

Andrea Caragliu, Chiara Del Bo, and Peter Nijkamp. 2009. Smart cities in Europe. In *3rd Central European Conference in Regional Science-CERS*. 45–59.

Jorge Cardoso and Amit Sheth. 2006. The Semantic Web and its Applications. In *Semantic Web Services Processes and Applications*. Vol. 3. Springer US, Boston, MA, 3–33. DOI:http://dx.doi.org/10.1007/978-0-387-34685-4_1

Angelo P. Castellani, Akbar Rahman, Esko Dijk, Thomas Fossati, and Salvatore Loreto. 2011. Best practices for HTTP-CoAP mapping implementation. (2011). http://tools.ietf.org/html/draft-castellani-core-http-mapping-02

Leonardo Weiss Ferreira Chaves and Christian Decker. 2010. A survey on organic smart labels for the Internet-of-Things. In *Networked Sensing Systems (INSS), 2010 Seventh International Conference on*. 161–164. DOI:http://dx.doi.org/10.1109/INSS.2010.5573467

Mung Chiang. 2015. *Fog Networking:An Overview on Research Opportunities*. Technical Report. http://www.princeton.edu/

Stefana Chirila, Camelia Lemnaru, and Mihaela Dinsoreanu. 2016. Semantic-based IoT device discovery and recommendation mechanism. In *2016 IEEE 12th International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE, 111–116. DOI:http://dx.doi.org/10.1109/ICCP.2016.7737131

Commonwealth Scientific and Industrial Research Organisation (CSIRO), Australia. 2011. Phenonet: Distributed Sensor Network for Phenomics supported by High Resolution Plant Phenomics Centre, CSIRO ICT Centre, and CSIRO Sensor and Sensor Networks TCP. (2011).

Michael Compton, Corey Henson, Holger Neuhaus, Laurent Lefort, and Amit Sheth. 2009. A Survey of the Semantic Specification of Sensors. In *2nd International Workshop on Semantic Sensor Networks, at 8th International Semantic Web Conference,*.

Vedat Coskun, Busra Ozdenizci, and Kerem Ok. 2013. A survey on near field communication (NFC) technology. (2013). DOI:http://dx.doi.org/10.1007/s11277-012-0935-5

Joan Daemen and Vincent Rijmen. 2002. *The design of AES- the Advanced Encryption Standard*. Spring{\-}er-Ver{\-}lag. 238 pages.

Soumya Kanti Datta, Christian Bonnet, and Jerome Haerri. 2015. Fog Computing architecture to enable consumer centric Internet of Things services. In *Proceedings of the International Symposium on Consumer Electronics, ISCE*, Vol. 2015-Augus. DOI:http://dx.doi.org/10.1109/ISCE.2015.7177778

Alessandra De Paola, Pierluca Ferraro, Salvatore Gaglio, Giuseppe Lo Re, and Sajal Das. 2016. An Adaptive Bayesian System for Context-Aware Data Fusion in Smart Environments. *IEEE Transactions on Mobile Computing* (2016), 1–1. DOI:http://dx.doi.org/10.1109/TMC.2016.2599158

Hans. Delfs and Helmut. Knebl. 2015. *Introduction to cryptography: Principles and applications: Third edition*. Springer. 1–508 pages. DOI:http://dx.doi.org/10.1007/978-3-662-47974-2

Artem Dementyev, Steve Hodges, Stuart Taylor, and Josh Smith. 2013. Power Consumption Analysis of Bluetooth Low Energy, ZigBee, and ANT Sensor Nodes in a Cyclic Sleep Scenario. IEEE.

Anind K Dey. 2001. Understanding and Using Context. *Personal Ubiquitous Comput.* 5, 1 (jan 2001), 4–7. DOI:http://dx.doi.org/10.1007/s007790170019

Clinton Dsouza, Gail Joon Ahn, and Marthony Taguinod. 2014. Policy-driven security management for fog computing: Preliminary framework and a case study. In *Proceedings of the 2014 IEEE 15th International Conference on Information Reuse and Integration, IEEE IRI 2014*. IEEE, 16–23. DOI:http://dx.doi.org/10.1109/IRI.2014.7051866

Harishchandra Dubey, Jing Yang, Nick Constant, Amir Mohammad Amiri, Qing Yang, and Kunal Makodiya. 2015. Fog Data: Enhancing Telehealth Big Data Through Fog Computing. *Proceedings of the ASE Big-Data & SocialInformatics 2015* (2015), 14:1—-14:6. DOI:http://dx.doi.org/10.1145/2818869.2818889

Aditya Dutta. 2013. Why HTTP is not enough for the Internet of Things. (2013).

European Commission. 2008. *Internet of Things in 2020 Road Map For The Future*. Technical Report. Working Group RFID of the ETP EPOSS.

European Research Cluster on the Internet of Things. 2015. *Internet of Things - IoT Semantic Interoperability: research challeges, best practices, recommendations and next steps*. Technical Report. 48 pages.

Ivan Farris, Roberto Girau, Leonardo Militano, Michele Nitti, Luigi Atzori, Antonio Iera, and Giacomo Morabito. 2015. Social Virtual Objects in the Edge Cloud. *IEEE Cloud Computing* 2, 6 (nov 2015), 20–28. DOI:http://dx.doi.org/10.1109/MCC.2015.116

I. Farris, L. Militano, M. Nitti, L. Atzori, and A. Iera. 2016. Federated edge-assisted mobile clouds for service provisioning in heterogeneous IoT environments. In *IEEE World Forum on Internet of Things, WF-IoT 2015 - Proceedings*. IEEE, 591–596. DOI:http://dx.doi.org/10.1109/WF-IoT.2015.7389120

R Fielding, J Gettys, J Mogul, H Frystyk, L Masinter, P Leach, and T Berners-Lee. 1999. RFC 2616 - Hypertext Transfer Protocol - HTTP/1.1. (1999). DOI:http://dx.doi.org/rfc/rfc2616.txt

Imola K Fodor. 2002. A survey of dimension reduction techniques. *Library* 18, 1 (2002), 1–18. DOI:http://dx.doi.org/10.2172/15002155

Forrest Stroud. Fog Computing. (????). http://www.webopedia.com/TERM/F/fog-computing.html

Javier Garcia-Martinez. The Internet of Things Goes Nano. *https://www.scientificamerican.com/article/the-internet-of-things-goes-nano/, Retrieved November 2016* (????).

David Gascon. 2015. *IoT Security Infographic Privacy, Authenticity, Confidentiality and Integrity of the Sensor Data. The Invisible Asset*. Technical Report. Libelium.

Elena I. Gaura, James Brusey, Michael Allen, Ross Wilkins, Dan Goldsmith, and Ramona Rednic. 2013. Edge mining the internet of things. *IEEE Sensors Journal* 13, 10 (oct 2013), 3816–3825.

Vangelis Gazis, Alessandro Leonardi, Kostas Mathioudakis, Konstantinos Sasloglou, Panayotis Kikiras, and Raghuram Sudhaakar. 2015. Components of fog computing in an industrial internet of things context. In *2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking - Workshops, SECON Workshops 2015*. IEEE, 37–42. DOI:http://dx.doi.org/10.1109/SECONW.2015.7328144

Tuan Nguyen Gia, Mingzhe Jiang, Amir Mohammad Rahmani, Tomi Westerlund, Pasi Liljeberg, and Hannu Tenhunen. 2015. Fog computing in healthcare Internet of Things: A case study on ECG feature extraction. In *Proceedings - 15th IEEE International Conference on Computer and Information Technology, CIT 2015, 14th IEEE International Conference on Ubiquitous Computing and Communications, IUCC 2015, 13th IEEE International Conference on Dependable, Autonomic and Se*. IEEE, 356–363. DOI:http://dx.doi.org/10.1109/CIT/IUCC/DASC/PICOM.2015.51

Nam Ky Giang, Michael Blackstock, Rodger Lea, and Victor C M Leung. 2015. Developing IoT applications in the Fog: A Distributed Dataflow approach. In *Proceedings - 2015 5th International Conference on the Internet of Things, IoT 2015*. IEEE, 155–162. DOI:http://dx.doi.org/10.1109/IOT.2015.7356560

Rudolf Giffinger, Christian Fertner, Hans Kramar, Robert Kalasek, Natasa Pichler-Milanovic, and Evert Meijers. 2007. *Smart cities Ranking of European medium-sized cities*. Research project report. Centre of Regional Science, Vienna UT.

Karina Gomez, Tinku Rasheed, Roberto Riggio, Daniele Miorandi, Cigdem Sengul, and Nico Bayer. 2013. Achilles and the tortoise: Power consumption in IEEE 802.11n and IEEE 802.11g networks. In *2013 IEEE Online Conference on Green Communications, OnlineGreenComm 2013*. IEEE, 20–26. DOI:http://dx.doi.org/10.1109/OnlineGreenCom.2013.6731023

Carlos Gonzalez. 2016. What is the Difference between Pneumatic, Hydraulic, and Electrical Actuators. (2016). http://machinedesign.com/linear-motion/what-s-difference-between-pneumatic-hydraulic-and-electrical-actuators [Retrieved November 2016].

Lin Gu, Deze Zeng, Song Guo, Ahmed Barnawi, and Yong Xiang. 2015. Cost-Efficient Resource Management in Fog Computing Supported Medical CPS. *IEEE Transactions on Emerging Topics in Computing* 6750, c (2015), 1–1. DOI:http://dx.doi.org/10.1109/TETC.2015.2508382

Daniel Halperin, Ben Greenstein, Anmol Sheth, and David Wetherall. 2010. Demystifying 802.11n power consumption. (2010).

James T. Harmening. 2013. Virtual Private Networks. In *Computer and Information Security Handbook*. 855–867. DOI:http://dx.doi.org/10.1016/B978-0-12-394397-2.00048-9

Mohammed A. Hassan, Mengbai Xiao, Qi Wei, and Songqing Chen. 2015. Help your mobile applications with fog computing. In *2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking - Workshops, SECON Workshops 2015*. IEEE, 49–54. DOI:http://dx.doi.org/10.1109/SECONW.2015.7328146

Paul S. Henry and Hui Luo. 2002. WiFi: What's next? *IEEE Communications Magazine* 40, 12 (dec 2002), 66–72. DOI:http://dx.doi.org/10.1109/MCOM.2002.1106162

R Heydon and N Hunn. 2012. *Bluetooth Low Energy : The Developer's Handbook*. 368 pages. https://www.bluetooth.org/DocMan/handlers/DownloadDoc.ashx

Takamasa Higuchi, Hirozumi Yamaguchi, Teruo Higashino, and Mineo Takai. 2014. A neighbor collaboration mechanism for mobile crowd sensing in opportunistic networks. In *2014 IEEE International Conference on Communications (ICC)*. IEEE, 42–47. DOI:http://dx.doi.org/10.1109/ICC.2014.6883292

Urs Hunkeler, Hong Linh Truong, and Andy Stanford-Clark. 2008. MQTT-S A publish/subscribe protocol for Wireless Sensor Networks. *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE '08)* (2008), 791–798. DOI:http://dx.doi.org/10.1109/COMSWA.2008.4554519

Hypercat Consortium. 2016. Hypercat. (2016). http://www.hypercat.io/

IEEE Instrumentation and Measurement Society. 2007. IEEE Standard for a Smart Transducer Interface for Sensors and Actuators Wireless Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats. *IEEE Std 1451.5-2007* (2007), C1 –236. DOI:http://dx.doi.org/10.1109/IEEESTD.2007.4346346

Masanori Ishino, Yuki Koizumi, and Toru Hasegawa. 2015. Leveraging proximity services for relay device discovery in user-provided IoT networks. In *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. IEEE, 553–558. DOI:http://dx.doi.org/10.1109/WF-IoT.2015.7389114

Bukhary Ikhwan Ismail, Ehsan Mostajeran Goortani, Mohd Bazli Ab Karim, Wong Ming Tat, Sharipah Setapa, Jing Yuan Luke, and Ong Hong Hoe. 2015. Evaluation of Docker as Edge computing platform. In *2015 IEEE Conference on Open Systems (ICOS)*. IEEE, 130–135. DOI:http://dx.doi.org/10.1109/ICOS.2015.7377291

Yukun Jia and Qingsong Xu. 2013. MEMS Microgripper Actuators and Sensors: The State-of-the-Art Survey. *Recent Patents on Mechanical Engineering* 6, 2 (2013), 132–142. http://www.eurekaselect.com/108839

Qingyun Jiang, Ruichun Tang, Peishun Liu, Yue Qiu, and Huimin Xu. 2014. Research on dynamic data fusion algorithm based on context awareness. In *2014 IEEE International Conference on Progress in Informatics and Computing*. IEEE, 529–534. DOI:http://dx.doi.org/10.1109/PIC.2014.6972391

Jiang Jiang Zhu, D. S. Chan, M. S. Prabhu, P. Natarajan, Hao Hao Hu, and F. Bonomi. 2013. Improving Web Sites Performance Using Edge Servers in Fog Computing Architecture. *2013 IEEE Seventh International Symposium on Service-Oriented System Engineering* (mar 2013), 320–323. DOI:http://dx.doi.org/10.1109/SOSE.2013.73

Saurabh Kulkarni, Shayan Saha, and Ryler Hockenbury. 2012. Preserving privacy in sensor-fog networks. In *2014 9th International Conference for Internet Technology and Secured Transactions, ICITST 2014*. IEEE, 96–99. DOI:http://dx.doi.org/10.1109/ICITST.2014.7038785

Valerie Lampkin, Weng Tat Leong, Leonardo Olivera, Sweta Rawat, Nagesh Subrahmanyam, and Rong Xiang. 2012. Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry. *IBM Redbooks* (2012), 270.

Arash Habibi Lashkari, Masood Mansoori, and Amir Seyed Danesh. 2009. Wired equivalent privacy (WEP) versus Wi-Fi protected access (WPA). In *2009 International Conference on Signal Processing Systems, ICSPS 2009*. 445–449. DOI:http://dx.doi.org/10.1109/ICSPS.2009.87

Jianhua Li, Jiong Jin, Dong Yuan, Marimuthu Palaniswami, and Klaus Moessner. 2015. EHOPES: Data-centered Fog platform for smart living. In *25th International Telecommunication Networks and Applications Conference, ITNAC 2015*. IEEE, 308–313. DOI:http://dx.doi.org/10.1109/ATNAC.2015.7366831

LinkLabs. 2016. *Low Power, Wide Area Networks*. Technical Report.

Linux Foundation. 2016a. AllJoyn Framework. (2016). https://allseenalliance.org/framework

Linux Foundation. 2016b. IoTivity. (2016). https://www.iotivity.org/

Xiufeng Liu, Nadeem Iftikhar, and Xike Xie. 2014. Survey of real-time processing systems for big data. In *Proceedings of the 18th International Database Engineering & Applications Symposium on - IDEAS '14*. ACM Press, New York, New York, USA, 356–361. DOI:http://dx.doi.org/10.1145/2628194.2628251

Xin Ma and Wei Luo. 2008. The analysis of 6LowPAN technology. In *Proceedings - 2008 Pacific-Asia Workshop on Computational Intelligence and Industrial Application, PACIIA 2008*, Vol. 1. IEEE, 963–966. DOI:http://dx.doi.org/10.1109/PACIIA.2008.72

H. Madsen, G. Albeanu, Bernard Burtschy, and Fl. Popentiu-Vladicescu. 2013. Reliability in the utility computing era: Towards reliable fog computing. In *International Conference on Systems, Signals, and Image Processing*. IEEE, 43–46. DOI:http://dx.doi.org/10.1109/IWSSIP.2013.6623445

Aapo Markkanen. 2015. Competitive Edge from Edge Intelligence IoT Analytics Today and in 2020. (2015). http://www.4cad.fr/content/files/Competitive-Edge-from-Edge-Intelligence-IoT-Whitepaper.pdf

Ajay Mohandas, Khoshrav Doctor, Shubham Jayawant, Mohit Pattni, and Era Johri. 1060. NFC vs Bluetooth. *International Journal of Multidisciplinary and Scientific Emerging Research* 4, 1 (1060), 2349–6037.

National Information Standards Organization. 2004. Understanding Metadata. *National Information Standards* MD:NISO Press (2004), 20. DOI:http://dx.doi.org/10.1017/S0003055403000534

Tommy Nilsson, Carl Hogsden, Charith Perera, Saeed Aghaee, David Scruton, Andreas Lund, and Alan F. Blackwell. 2016. Applying Seamful Design in Location-based Mobile Museum Applications. *ACM Transactions on Multimedia Computing Communications and Applications (TOMM)* (2016).

Takayuki Nishio, Ryoichi Shinkuma, Tatsuro Takahashi, and Narayan B. Mandayam. 2013. Service-oriented heterogeneous resource sharing for optimizing service latency in mobile cloud. In *Proceedings of the first international workshop on Mobile cloud computing & networking - MobileCloud '13*. ACM Press, New York, New York, USA, 19. DOI:http://dx.doi.org/10.1145/2492348.2492354

Loutfi Nuaymi. 2007. *WiMAX: Technology for Broadband Wireless Access*. 1–283 pages. DOI:http://dx.doi.org/10.1002/9780470319055

OASIS. 2014. MQTT Version 3.1.1. *OASIS Standard* October (2014), 81. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

John O'Hara. 2007. Toward a commodity enterprise middleware. *Queue* 5, 4 (may 2007), 48–55. DOI:http://dx.doi.org/10.1145/1255421.1255424

OpenIoT Consortium. 2012. Open Source Solution for the Internet of Things into the Cloud. (jan 2012).

Jessica Oueis, Emilio Calvanese Strinati, and Sergio Barbarossa. 2015. The Fog Balancing: Load Distribution for Small Cell Cloud Computing. *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)* (may 2015), 1–6. DOI:http://dx.doi.org/10.1109/VTCSpring.2015.7146129

Charith Perera, Prem Prakash Jayaraman, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. 2014a. *Context-aware Dynamic Discovery and Configuration of Things in Smart Environments*. Springer International Publishing, Cham, Chapter Context-Aw, 215–241. DOI:http://dx.doi.org/10.1007/978-3-319-05029-4_9

C Perera, P P Jayaraman, A Zaslavsky, D Georgakopoulos, and P Christen. 2014b. Sensor discovery and configuration framework for the Internet of Things paradigm. In *Internet of Things (WF-IoT), 2014 IEEE World Forum on*. 94–99. DOI:http://dx.doi.org/10.1109/WF-IoT.2014.6803127

Charith Perera, Chi Harold Liu, and Srimal Jayawardena. 2014c. A Survey on Internet of Things From Industrial Market Perspective. *IEEE Access* 2 (2014), 1660–1679. DOI:http://dx.doi.org/10.1109/ACCESS.2015.2389854

Charith Perera, Chi Harold Liu, and Srimal Jayawardena. 2015a. The Emerging Internet of Things Marketplace from an Industrial Perspective: A Survey. *IEEE Transactions on Emerging Topics in Computing* 3, 4 (2015), 585–598.

Charith Perera, Rajiv Ranjan, Lizhe Wang, Samee U. Khan, and Albert Y. Zomaya. 2015b. Big data privacy in the internet of things era. *IT Professional* 17, 3 (2015), 32–39.

Charith Perera, Dumidu Talagala, Chi Harold Liu, and Julio C. Estrella. 2015c. Energy-Efficient Location and Activity-Aware On-Demand Mobile Distributed Sensing Platform for Sensing as a Service in IoT Clouds. *IEEE Transactions on Computational Social Systems* 2, 4 (dec 2015), 171–181.

Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. 2014a. Context Aware Computing for The Internet of Things: A Survey. *Communications Surveys Tutorials, IEEE* 16, 1 (2014), 414 – 454.

Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. 2014b. Sensing as a service model for smart cities supported by Internet of Things. *European Transactions on Telecommunications* 25, 1 (2014), 81–93. DOI:http://dx.doi.org/10.1002/ett.2704

Charith Perera, Arkady Zaslavsky, Peter Christen, Ali Salehi, and Dimitrios Georgakopoulos. 2012. Connecting Mobile Things to Global Sensor Network Middleware using System-generated Wrappers. In *Eleventh ACM International Workshop on Data Engineering for Wireless and Mobile Access (ACM SIGMOD/PODS 2012-Workshop-MobiDE)*. Scottsdale, Arizona, USA, 23–30. DOI:http://dx.doi.org/10.1145/2258056.2258062

S Pietschmann, A Mitschick, R Winkler, and K Meissner. 2008. CroCo: Ontology-Based, Cross-Application Context Management. In *Semantic Media Adaptation and Personalization, 2008. SMAP '08. Third International Workshop on*. 88–93. DOI:http://dx.doi.org/10.1109/SMAP.2008.10

R. Pozza, M. Nati, S. Georgoulas, K. Moessner, and A. Gluhak. 2015. Neighbor Discovery for Opportunistic Networking in Internet of Things Scenarios: A Survey. *IEEE Access* 3 (2015), 1101–1131. DOI:http://dx.doi.org/10.1109/ACCESS.2015.2457031

Jurgo Preden, Jaanus Kaugerand, Erki Suurjaak, Sergei Astapov, Leo Motus, and Raido Pahtma. 2015. Data to decision: pushing situational information needs to the edge of the network. In *2015 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision*. IEEE, 158–164. DOI:http://dx.doi.org/10.1109/COGSIMA.2015.7108192

Daniel Puschmann, Payam Barnaghi, and Rahim Tafazolli. 2016. Adaptive Clustering for Dynamic IoT Data Streams. *IEEE Internet of Things Journal* (2016), 1–1. DOI:http://dx.doi.org/10.1109/JIOT.2016.2618909

A. Rahman and E. Dijk. 2014. Group Communication for the Constrained Application Protocol (CoAP). (2014). https://tools.ietf.org/pdf/rfc7390.pdf

Shahid Raza, Hossein Shafagh, Kasun Hewage, Rene Hummen, and Thiemo Voigt. 2013. Lithe: Lightweight secure CoAP for the internet of things. *IEEE Sensors Journal* 13, 10 (2013), 3711–3720.

Shahid Raza, Daniele Trabalza, and Thiemo Voigt. 2012. 6LoWPAN compressed DTLS for CoAP. In *Proceedings - IEEE International Conference on Distributed Computing in Sensor Systems, DCOSS 2012*. 287–289. DOI:http://dx.doi.org/10.1109/DCOSS.2012.55

RCUK Digital Economy HAT Project. 2014. *Engineering a Market for Personal Data: The Hub-of-all-Things (HAT) A Briefing Paper*. Technical Report. RCUK Digital Economy.

Gil Reiter. 2014. *Wireless connectivity for the Internet of Things*. Technical Report. Texas Instruments Incorporated, Texas. http://www.ti.com.cn/cn/lit/wp/swry010/swry010.pdf

R. L. Rivest, A. Shamir, and L. Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* 21, 2 (feb 1978), 120–126. DOI:http://dx.doi.org/10.1145/359340.359342

RS Components. 2015. 11 Internet of Things (IoT) Protocols You Need to Know About DesignSpark. (2015). http://www.rs-online.com/designspark/electronics/knowledge-item/eleven-internet-of-things-iot-protocols-you-need-to-know-about

P Saint-Andre. 2011. Extensible Messaging and Presence Protocol (XMPP) : Core. (2011).

Arjuna Sathiaseelan, Adisorn Lertsinsrubtavee, Adarsh Jagan, Prakash Baskaran, and Jon Crowcroft. 2016. Cloudrone: Micro Clouds in the Sky. In *ACM Mobisys Dronet*.

Stan Schneider. 2013. Understanding The Protocols Behind The Internet Of Things. (2013). http://electronicdesign.com/iot/understanding-protocols-behind-internet-things

Vivek Kumar Sehgal, Anubhav Patrick, Ashutosh Soni, and Lucky Rajput. 2015. Smart human security framework using internet of things, Cloud and fog computing. *Advances in Intelligent Systems and Computing* 321 (2015), 251–263. DOI:http://dx.doi.org/10.1007/978-3-319-11227-5_22

Zach Shelby and Carsten Bormann. 2009. *6LoWPAN: The Wireless Embedded Internet*. 1–223 pages. DOI:http://dx.doi.org/10.1002/9780470686218

Z Shelby, K Hartke, and C Bormann. 2014. The Constrained Application Protocol (CoAP). *Rfc 7252* (2014), 112. DOI:http://dx.doi.org/10.1007/s13398-014-0173-7.2

Dilpreet Singh and Chandan K Reddy. 2015. A survey on platforms for big data analytics. *Journal of Big Data* 2, 1 (dec 2015), 8. DOI:http://dx.doi.org/10.1186/s40537-014-0008-6

Meena Singh, M. A. Rajan, V. L. Shivraj, and P. Balamuralidhar. 2015. Secure MQTT for Internet of Things (IoT). In *Proceedings - 2015 5th International Conference on Communication Systems and Network Technologies, CSNT 2015*. 746–751. DOI:http://dx.doi.org/10.1109/CSNT.2015.16

Maninder Pal Singh, Mohammad A. Hoque, and Sasu Tarkoma. 2016. A survey of systems for massive stream analytics. (may 2016). http://arxiv.org/abs/1605.09021

Andy Stanford-clark and Hong Linh Truong. 2008. MQTT For Sensor Networks (MQTT-S) Protocol Specification. *mqttorg* (2008), 1–27.

Vladimir Stantchev, Ahmed Barnawi, Sarfaraz Ghulam, Johannes Schubert, and Gerrit Tamm. 2015. Smart Items, Fog and Cloud Computing as Enablers of Servitization in Healthcare. *Sensors & Transducers* 185, 2 (2015), 121–128.

Ivan Stojmenovic and Sheng Wen. 2014. The Fog Computing Paradigm: Scenarios and Security Issues. In *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on*. IEEE, 1–8. DOI:http://dx.doi.org/10.15439/2014F503

Salvatore J. Stolfo, Malek Ben Salem, and Angelos D. Keromytis. 2012. Fog computing: Mitigating insider data theft attacks in the cloud. In *Proceedings - IEEE CS Security and Privacy Workshops, SPW 2012*. IEEE, 125–128. DOI:http://dx.doi.org/10.1109/SPW.2012.19

Jingtao Su, Fuhong Lin, Xianwei Zhou, and Xing Lu. 2015. Steiner tree based optimal resource caching scheme in fog computing. *China Communications* 12, 8 (aug 2015), 161–168. DOI:http://dx.doi.org/10.1109/CC.2015.7224698

G Suciu, A Vulpe, S Halunga, O Fratu, G Todoran, and V Suciu. 2013. Smart Cities Built on Resilient Cloud Computing and Secure Internet of Things. In *2013 19th International Conference on Control Systems and Computer Science*. 513–518. DOI:http://dx.doi.org/10.1109/CSCS.2013.58

Bo Tang, Zhen Chen, Gerald Hefferman, Tao Wei, Haibo He, and Qing Yang. 2015. A Hierarchical Distributed Fog Computing Architecture for Big Data Analysis in Smart Cities. *Proceedings of the ASE BigData & SocialInformatics 2015* (2015), 28:1—-28:6. DOI:http://dx.doi.org/10.1145/2818869.2818898

VFabric Team. 2013. Choosing Your Messaging Protocol: AMQP, MQTT, or STOMP. (2013). http://blogs.vmware.com/vfabric/2013/02/choosing-your-messaging-protocol-amqp-mqtt-or-stomp.html

Texas Instruments. 2014. *Wireless Connectivity*. Technical Report. Texas.

Dinesh Thangavel, Xiaoping Ma, Alvin Valera, Hwee Xian Tan, and Colin Keng Yan Tan. 2014. Performance evaluation of MQTT and CoAP via a common middleware. In *IEEE ISSNIP 2014 - 2014 IEEE 9th International Conference on Intelligent Sensors, Sensor Networks and Information Processing, Conference Proceedings*. DOI:http://dx.doi.org/10.1109/ISSNIP.2014.6827678

Stephen Thomas. 2000. *SSL and TLS essentials*.

Thread Group. 2015. *Thread Stack Fundamentals*. Technical Report.

Phillip Tracy. 2016. What is the internet of things at nanoscale. (2016). http://www.rcrwireless.com/20160912/big-data-analytics/nano-scale-iot-tag31-tag99 [Retrieved November 2016].

Nguyen B. Truong, Gyu Myoung Lee, and Yacine Ghamri-Doudane. 2015. Software defined networking-based vehicular Adhoc Network with Fog Computing. In *Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management, IM 2015*. IEEE, 1202–1207. DOI:http://dx.doi.org/10.1109/INM.2015.7140467

Lorenzo Vangelista, Andrea Zanella, and Michele Zorzi. 2015. Long-range IoT technologies: The dawn of LoRaTM. In *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, Vol. 159. 51–58. DOI:http://dx.doi.org/10.1007/978-3-319-27072-2_7

Steve Vinoski. 2006. Advanced message queuing protocol. *IEEE Internet Computing* 10, 6 (2006), 87–89. DOI:http://dx.doi.org/10.1109/MIC.2006.116

Roy Want. 2006. An introduction to RFID technology. (2006). DOI:http://dx.doi.org/10.1109/MPRV.2006.2

Roy Want. 2011. Near Field Communication. *Pervasive Computing, IEEE, 10(3)* (2011), 4–7. DOI:http://dx.doi.org/10.1109/MPRV.2011.55

Thomas Watteyne, Xavier Vilajosana, Branko Kerkez, Fabien Chraim, Kevin Weekly, Qin Wang, Steven Glaser, and Kris Pister. 2012. OpenWSN: a standards-based low-power wireless development environment. *Transactions on Emerging Telecommunications Technologies* 23, 5 (2012), 480–493. DOI:http://dx.doi.org/10.1002/ett.2558

Wang Wei and Payam Barnaghi. 2009. Semantic annotation and reasoning for sensor data. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 5741 LNCS. Springer Berlin Heidelberg, 66–76. DOI:http://dx.doi.org/10.1007/978-3-642-04471-7_6

M. Yannuzzi, R. Milito, R. Serral-Gracia, D. Montero, and M. Nemirovsky. 2014. Key ingredients in an IoT recipe: Fog Computing, Cloud computing, and more Fog Computing. *2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)* (dec 2014), 325–329. DOI:http://dx.doi.org/10.1109/CAMAD.2014.7033259

Jui-Hung Yeh, Jyh-Cheng Chen, and Chi-Chen Lee. 2003. WLAN standards. *IEEE Potentials* 22, 4 (2003), 16–22. DOI:http://dx.doi.org/10.1109/MP.2003.1238688

Shanhe Yi, Cheng Li, and Qun Li. 2015a. A Survey of Fog Computing. In *Proceedings of the 2015 Workshop on Mobile Big Data - Mobidata '15*. ACM Press, New York, New York, USA, 37–42. DOI:http://dx.doi.org/10.1145/2757384.2757397

Shanhe Yi, Zhengrui Qin, and Qun Li. 2015b. Security and Privacy Issues of Fog Computing: A Survey. Springer International Publishing, 685–695. DOI:http://dx.doi.org/10.1007/978-3-319-21837-3_67

Z-Wave. 2015. About Z-Wave. (2015). http://www.z-wave.com/z-wave

John K. Zao, Tchin Tze Gan, Chun Kai You, Sergio Jose Rodriguez Mendez, Cheng En Chung, Yu Te Wang, Tim Mullen, and Tzyy Ping Jung. 2014. Augmented brain computer interaction based on fog computing and linked data. In *Proceedings - 2014 International Conference on Intelligent Environments, IE 2014.* IEEE, 374–377. DOI:http://dx.doi.org/10.1109/IE.2014.54

Arkady Zaslavsky, Charith Perera, and Dimitrios Georgakopoulos. 2012. Sensing as a Service and Big Data. In *International Conference on Advances in Cloud Computing (ACC)*. Bangalore, India, 21–29.

Paul. Zikopoulos. 2012. *Understanding big data : analytics for enterprise class Hadoop and streaming data*. McGraw-Hill. 141 pages.

T. G. Zimmerman. 1996. Personal Area Networks: Near-field intrabody communication. *IBM Systems Journal* 35, 3.4 (1996), 609–617. DOI:http://dx.doi.org/10.1147/sj.353.0609