Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications

Ala Al-Fuqaha, *Senior Member, IEEE*, Mohsen Guizani, *Fellow, IEEE*, Mehdi Mohammadi, *Student Member, IEEE*, Mohammed Aledhari, *Student Member, IEEE*, and Moussa Ayyash, *Senior Member, IEEE*

***Abstract*—This paper provides an overview of the Internet of Things (IoT) with emphasis on enabling technologies, protocols, and application issues. The IoT is enabled by the latest developments in RFID, smart sensors, communication technologies, and Internet protocols. The basic premise is to have smart sensors collaborate directly without human involvement to deliver a new class of applications. The current revolution in Internet, mobile, and machine-to-machine (M2M) technologies can be seen as the first phase of the IoT. In the coming years, the IoT is expected to bridge diverse technologies to enable new applications by connecting physical objects together in support of intelligent decision making. This paper starts by providing a horizontal overview of the IoT. Then, we give an overview of some technical details that pertain to the IoT enabling technologies, protocols, and applications. Compared to other survey papers in the field, our objective is to provide a more thorough summary of the most relevant protocols and application issues to enable researchers and application developers to get up to speed quickly on how the different protocols fit together to deliver desired functionalities without having to go through RFCs and the standards specifications. We also provide an overview of some of the key IoT challenges presented in the recent literature and provide a summary of related research work. Moreover, we explore the relation between the IoT and other emerging technologies including big data analytics and cloud and fog computing. We also present the need for better horizontal integration among IoT services. Finally, we present detailed service use-cases to illustrate how the different protocols presented in the paper fit together to deliver desired IoT services.**

***Index Terms*—Internet of Things (IoT), CoAP, MQTT, AMQP, XMPP, DDS, mDNS, IoT gateway.**

I. Introduction

A GROWING number of physical objects are being connected to the Internet at an unprecedented rate realizing the idea of the Internet of Things (IoT). A basic example of such objects includes thermostats and HVAC (Heating, Ventilation, and Air Conditioning) monitoring and control systems that enable smart homes. There are also other domains and environments in which the IoT can play a remarkable role and improve the quality of our lives. These applications include transportation, healthcare, industrial automation, and emergency response to natural and man-made disasters where human decision making is difficult.

Fig. 1. The overall picture of IoT emphasizing the vertical markets and the horizontal integration between them.

The IoT enables physical objects to see, hear, think and perform jobs by having them "talk" together, to share information and to coordinate decisions. The IoT transforms these objects from being traditional to smart by exploiting its underlying technologies such as ubiquitous and pervasive computing, embedded devices, communication technologies, sensor networks, Internet protocols and applications. Smart objects along with their supposed tasks constitute domain specific applications (vertical markets) while ubiquitous computing and analytical services form application domain independent services (horizontal markets). Fig. 1 illustrates the overall concept of the IoT in which every domain specific application is interacting with domain independent services, whereas in each domain sensors and actuators communicate directly with each other.

Over time, the IoT is expected to have significant home and business applications, to contribute to the quality of life and to grow the world's economy. For example, smart-homes will enable their residents to automatically open their garage when reaching home, prepare their coffee, control climate control systems, TVs and other appliances. In order to realize this potential growth, emerging technologies and innovations, and service

Manuscript received October 11, 2014; revised April 11, 2015; accepted May 25, 2015. Date of publication June 15, 2015; date of current version November 18, 2015.

A. Al-Fuqaha, M. Mohammadi, and M. Aledhari are with the Department of Computer Science, Western Michigan University, Kalamazoo, MI 49008 USA (e-mail: ala.al-fuqaha@wmich.edu; mehdi.mohammadi@wmich.edu; mohammed.aledhari@wmich.edu).

M. Guizani is with the Department of Computer Science & Engineering, Qatar University, Doha 2713, Qatar (e-mail: mguizani@ieee.org).

M. Ayyash is with the Department of Information Studies, Chicago State University, Chicago, IL 60628 USA (e-mail: mayyash@csu.edu).

Digital Object Identifier 10.1109/COMST.2015.2444095

applications need to grow proportionally to match market demands and customer needs. Furthermore, devices need to be developed to fit customer requirements in terms of availability anywhere and anytime. Also, new protocols are required for communication compatibility between heterogeneous things (living things, vehicles, phones, appliances, goods, etc.).

Moreover, architecture standardization can be seen as a backbone for the IoT to create a competitive environment for companies to deliver quality products. In addition, the traditional Internet architecture needs to be revised to match the IoT challenges. For example, the tremendous number of objects willing to connect to the Internet should be considered in many underlying protocols. In 2010, the number of Internet connected objects had surpassed the earth's human population [1]. Therefore, utilizing a large addressing space (e.g., IPv6) becomes necessary to meet customer demands for smart objects. Security and privacy are other important requirements for the IoT due to the inherent heterogeneity of the Internet connected objects and the ability to monitor and control physical objects. Furthermore, management and monitoring of the IoT should take place to ensure the delivery of high-quality services to customers at an efficient cost.

There are several published survey papers that cover different aspects of the IoT technology. For example, the survey by Atzori *et al.* [2] covers the main communication enabling technologies, wired and wireless and the elements of wireless sensor networks (WSNs). In [3], the authors address the IoT architecture and the challenges of developing and deploying IoT applications. Enabling technologies and application services using a centralized cloud vision are presented in [4]. The authors in [5] provide a survey of the IoT for specialized clinical wireless devices using 6LoWPAN/IEEE 802.15.4, Bluetooth and NFC for mHealth and eHealth applications. Moreover, [6] addresses the IoT in terms of enabling technologies with emphasis on RFID and its potential applications. IoT challenges are presented in [7] to bridge the gap between research and practical aspects. An overview of the current IETF standards and challenges for the IoT has been presented in [8].

The outline of the contributions of this paper relative to the recent literature in the field can be summarized as:

- Compared to other survey papers in the field, this survey provides a deeper summary of the most relevant IETF, IEEE and EPCglobal protocols and standards to enable researchers to get up to speed quickly without having to dig through the details presented in the RFCs and the standards specifications.
- We provide an overview of some of the key IoT challenges presented in the recent literature and provide a summary of related research work. Moreover, we explore the relation between the IoT and other emerging technologies including big data analytics and cloud and fog computing.
- We present the need for better horizontal integration among IoT services.
- We also present detailed service use-cases to illustrate how the different protocols presented in the paper fit together to deliver desired IoT services.

The rest of this paper is organized as follows: Section II provides a summary of the market opportunity that is enabled by the IoT. Sections III and IV discuss the overall architecture of the IoT and its elements, respectively. Current protocols and standards of the IoT are presented in Section V. Security, trust, monitoring, management and Quality of Service (QoS) issues are discussed in Section VI. The interplay between big data and the IoT and the need to manage and analyze massive amounts of data generated by the IoT is the focus of Section VII. In Section VIII, we present the need for intelligent IoT data-exchange and management services to achieve better horizontal integration among IoT services. The integration of the different IoT protocols to deliver desired functionalities is presented in Section IX using some use-cases of IoT applications and services. Finally, Section X presents a summary of lessons learned and concludes this study.

II. Market Opportunity

The IoT offers a great market opportunity for equipment manufacturers, Internet service providers and application developers. The IoT smart objects are expected to reach 212 billion entities deployed globally by the end of 2020 [9]. By 2022, M2M traffic flows are expected to constitute up to 45% of the whole Internet traffic [1], [9], [10]. Beyond these predictions, McKinsey Global Institute reported that the number of connected machines (units) has grown 300% over the last 5 years [11]. Traffic monitoring of a cellular network in the U.S. also showed an increase of 250% for M2M traffic volume in 2011 [12].

Economic growth of IoT-based services is also considerable for businesses. Healthcare and manufacturing applications are projected to form the biggest economic impact. Healthcare applications and related IoT-based services such as mobile health (m-Health) and telecare that enable medical wellness, prevention, diagnosis, treatment and monitoring services to be delivered efficiently through electronic media are expected to create about \$1.1–\$2.5 trillion in growth annually by the global economy by 2025. The whole annual economic impact caused by the IoT is estimated to be in range of \$2.7 trillion to \$6.2 trillion by 2025 [11]. Fig. 2 shows the projected market share of dominant IoT applications [11].

On the other hand, Wikibon predicts that the value created from the industrial Internet to be about \$1279 billion in 2020 with Return on Investment (ROI) growing to 149% compared to 13% in 2012 [13]. Moreover, Navigant recently reported that the Building Automation Systems (BAS) market is expected to rise from \$58.1 billion in 2013 to reach \$100.8 billion by 2021; a 60% increase [14].

All these statistics, however, point to a potentially significant and fast-pace growth of the IoT in the near future, related industries and services. This progression provides a unique opportunity for traditional equipment and appliance manufacturers to transform their products into "smart things." Spreading the IoT and related services globally requires Internet Service Providers (ISPs) to provision their networks to provide QoS for a mix of M2M, *person-to-machine* (P2M) and *person-to-person* (P2P) traffic flows.

III. IoT Architecture

The IoT should be capable of interconnecting billions or trillions of heterogeneous objects through the Internet, so there is a

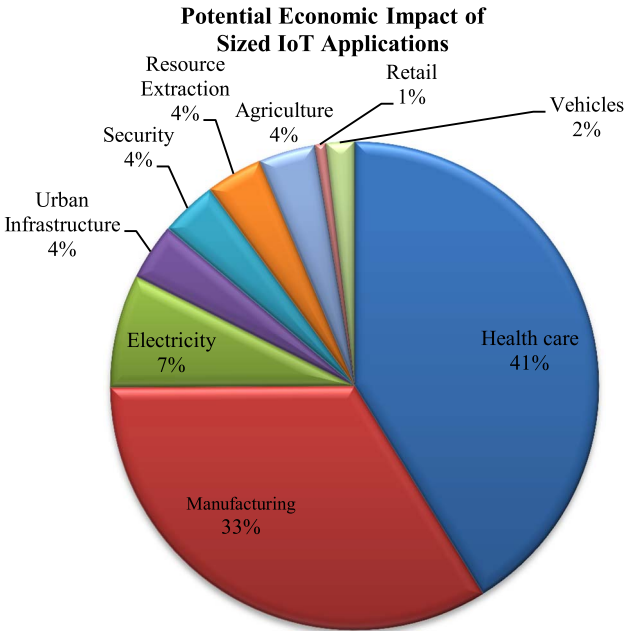

Fig. 2. Projected market share of dominant IoT applications by 2025.

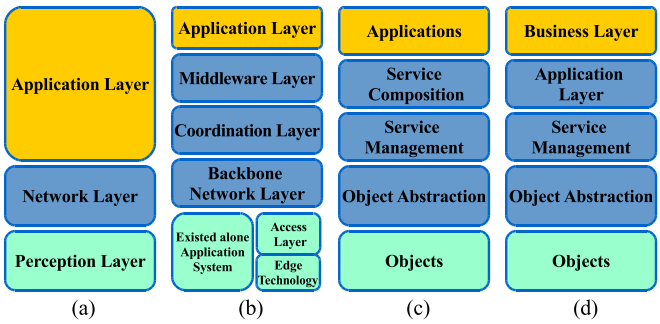

Fig. 3. The IoT architecture. (a) Three-layer. (b) Middle-ware based. (c) SOA based. (d) Five-layer.

critical need for a flexible layered architecture. The ever increasing number of proposed architectures has not yet converged to a reference model [15]. Meanwhile, there are some projects like IoT-A [16] which try to design a common architecture based on the analysis of the needs of researchers and the industry.

From the pool of proposed models, the basic model is a 3-layer architecture [3], [17], [18] consisting of the Application, Network, and Perception Layers. In the recent literature, however, some other models have been proposed that add more abstraction to the IoT architecture [2], [3], [17]–[20]. Fig. 3 illustrates some common architectures among them is the 5-layer model (not to be confused with the TCP/IP layers) which has been used in [3], [17], [18]. Next, we provide a brief discussion on these five layers.

A. Objects Layer

The first layer, the Objects (devices) or perception layer, represents the physical sensors of the IoT that aim to collect and process information. This layer includes sensors and actuators to perform different functionalities such as querying location, temperature, weight, motion, vibration, acceleration, humidity, etc. Standardized plug-and-play mechanisms need to be used by the perception layer to configure heterogeneous objects [17], [18]. The perception layer digitizes and transfers data to the Object Abstraction layer through secure channels. The big data created by the IoT are initiated at this layer.

B. Object Abstraction Layer

Object Abstraction transfers data produced by the Objects layer to the Service Management layer through secure channels. Data can be transferred through various technologies such as RFID, 3G, GSM, UMTS, WiFi, Bluetooth Low Energy, infrared, ZigBee, etc. Furthermore, other functions like cloud computing and data management processes are handled at this layer [17].

C. Service Management Layer

Service Management or Middleware (pairing) layer pairs a service with its requester based on addresses and names. This layer enables the IoT application programmers to work with heterogeneous objects without consideration to a specific hardware platform. Also, this layer processes received data, makes decisions, and delivers the required services over the network wire protocols [3], [18], [20].

D. Application Layer

The application layer provides the services requested by customers. For instance, the application layer can provide temperature and air humidity measurements to the customer who asks for that data. The importance of this layer for the IoT is that it has the ability to provide high-quality smart services to meet customers' needs. The application layer covers numerous vertical markets such as smart home, smart building, transportation, industrial automation and smart healthcare [3], [17]–[19].

E. Business Layer

The business (management) layer manages the overall IoT system activities and services. The responsibilities of this layer are to build a business model, graphs, flowcharts, etc. based on the received data from the Application layer. It is also supposed to design, analyze, implement, evaluate, monitor, and develop IoT system related elements. The Business Layer makes it possible to support decision-making processes based on Big Data analysis. In addition, monitoring and management of the underlying four layers is achieved at this layer. Moreover, this layer compares the output of each layer with the expected output to enhance services and maintain users' privacy [3], [18].

Remarks: The architectures that borrow their layers and concepts from network stacks (like the three-layer model) do not conform to real IoT environments since, e.g., the "Network Layer" does not cover all underlying technologies that transfer data to an IoT platform. In addition, these models have been designed to address specific types of communication media such as WSNs. More importantly, the layers are supposed to be run on resource-constrained devices while having a layer like "Service Composition" in SOA-based architecture takes rather

Fig. 4. The IoT elements.

a big fraction of the time and energy of the device to communicate with other devices and integrate the required services.

In the five-layer model, the Application Layer is the interface by which end-users can interact with a device and query for interesting data. It also provides an interface to the Business Layer where high-level analysis and reports can be produced. The control mechanisms of accessing data in the application layer are also handled at this layer. This layer is hosted on powerful devices due to its complex and enormous computational needs. Considering these points on the one hand and sticking to the simplicity of the architecture on the other hand, the five-layer architecture is the most applicable model for IoT applications.

IV. IoT Elements

Understanding the IoT building blocks helps to gain a better insight into the real meaning and functionality of the IoT. In the following sections we discuss six main elements needed to deliver the functionality of the IoT as illustrated in Fig. 4. Table II shows the categories of these elements and examples of each category.

A. Identification

Identification is crucial for the IoT to name and match services with their demand. Many identification methods are available for the IoT such as electronic product codes (EPC) and ubiquitous codes (uCode) [21]. Furthermore, addressing the IoT objects is critical to differentiate between object ID and its address. Object ID refers to its name such as "T1" for a particular temperature sensor and object's address refers to its address within a communications network. In addition, addressing methods of IoT objects include IPv6 and IPv4. 6LoWPAN [22], [23] provides a compression mechanism over IPv6 headers that makes IPv6 addressing appropriate for low power wireless networks. Distinguishing between object's identification and address is imperative since identification methods are not globally unique, so addressing assists to uniquely identify objects. In addition, objects within the network might use public IPs and not private ones. Identification methods are used to provide a clear identity for each object within the network.

B. Sensing

The IoT sensing means gathering data from related objects within the network and sending it back to a data warehouse, database, or cloud. The collected data is analyzed to take specific actions based on required services. The IoT sensors can be smart sensors, actuators or wearable sensing devices. For example, companies like Wemo, revolv and SmartThings offer smart hubs and mobile applications that enable people to monitor and control thousands of smart devices and appliances inside buildings using their smartphones [24]–[26].

Single Board Computers (SBCs) integrated with sensors and built-in TCP/IP and security functionalities are typically used to realize IoT products (e.g., Arduino Yun, Raspberry PI, BeagleBone Black, etc.). Such devices typically connect to a central management portal to provide the required data by customers.

C. Communication

The IoT communication technologies connect heterogeneous objects together to deliver specific smart services. Typically, the IoT nodes should operate using low power in the presence of lossy and noisy communication links. Examples of communication protocols used for the IoT are WiFi, Bluetooth, IEEE 802.15.4, Z-wave, and LTE-Advanced. Some specific communication technologies are also in use like RFID, Near Field Communication (NFC) and *ultra-wide bandwidth* (UWB). RFID is the first technology used to realize the M2M concept (RFID tag and reader). The RFID tag represents a simple chip or label attached to provide object's identity. The RFID reader transmits a query signal to the tag and receives reflected signal from the tag, which in turn is passed to the database. The database connects to a processing center to identify objects based on the reflected signals within a (10 cm to 200 m) range [27]. RFID tags can be active, passive or semi-passive/active. Active tags are powered by battery while passive ones do not need battery. Semi-passive/active tags use board power when needed.

The NFC protocol works at high frequency band at 13.56 MHz and supports data rate up to 424 kbps. The applicable range is up to 10 cm where communication between active readers and passive tags or two active readers can occur [28]. The UWB communication technology is designed to support communications within a low range coverage area using low energy and high bandwidth whose applications to connect sensors have been increased recently [29].

Another communication technology is WiFi that uses radio waves to exchange data amongst things within 100 m range [30]. WiFi allows smart devices to communicate and exchange information without using a router in some *ad hoc* configurations. Bluetooth presents a communication technology that is used to exchange data between devices over short distances using short-wavelength radio to minimize power consumption [31]. Recently, the Bluetooth *special interest group* (SIG) produced Bluetooth 4.1 that provides Bluetooth Low Energy as well as high-speed and IP connectivity to support IoT [32]. The IEEE 802.15.4 standard specifies both a physical layer and a medium access control for low power wireless networks targeting reliable and scalable communications [33].

TABLE I
COMMON OPERATING SYSTEMS USED IN IOT ENVIRONMENTS

Operating System	Language Support	Minimum Memory (KB)	Event-based Programming	Multi-threading	Dynamic Memory
TinyOS	nesC	1	Yes	Partial	Yes
Contiki	C	2	Yes	Yes	Yes
LiteOS	C	4	Yes	Yes	Yes
Riot OS	C/C++	1.5	No	Yes	Yes
Android	Java	-	Yes	Yes	Yes

LTE (Long-Term Evolution) is originally a standard wireless communication for high-speed data transfer between mobile phones based on GSM/UMTS network technologies [34]. It can cover fast-travelling devices and provide multicasting and broadcasting services. LTE-A (LTE Advanced) [35] is an improved version of LTE including bandwidth extension which supports up to 100 MHz, downlink and uplink spatial multiplexing, extended coverage, higher throughput and lower latencies.

D. Computation

Processing units (e.g., microcontrollers, microprocessors, SOCs, FPGAs) and software applications represent the "brain" and the computational ability of the IoT. Various hardware platforms were developed to run IoT applications such as Arduino, UDOO, FriendlyARM, Intel Galileo, Raspberry PI, Gadgeteer, BeagleBone, Cubieboard, Z1, WiSense, Mulle, and T-Mote Sky.

Furthermore, many software platforms are utilized to provide IoT functionalities. Among these platforms, Operating Systems are vital since they run for the whole activation time of a device. There are several Real-Time Operating Systems (RTOS) that are good candidates for the development of RTOS-based IoT applications. For instance, the Contiki RTOS has been used widely in IoT scenarios. Contiki has a simulator called Cooja which allows researcher and developers to simulate and emulate IoT and wireless sensor network (WSN) applications [36]. TinyOS [37], LiteOS [38] and Riot OS [39] also offer light weight OS designed for IoT environments. Moreover, some auto industry leaders with Google established the Open Auto Alliance (OAA) and are planning to bring new features to the Android platform to accelerate the adoption of the Internet of Vehicles (IoV) paradigm [40]. Some features of these operating systems are compared in Table I.

Cloud Platforms form another important computational part of the IoT. These platforms provide facilities for smart objects to send their data to the cloud, for big data to be processed in real-time, and eventually for end-users to benefit from the knowledge extracted from the collected big data. There are a lot of free and commercial cloud platforms and frameworks available to host IoT services. Some of these services are introduced in Section VII-B.

E. Services

Overall, IoT services can be categorized under four classes [41], [42]: Identity-related Services, Information Aggregation Services, Collaborative-Aware Services and Ubiquitous Services. Identity-related services are the most basic and important services that are used in other types of services. Every application that needs to bring real world objects to the virtual world has to identify those objects. Information Aggregation Services collect and summarize raw sensory measurements that need to be processed and reported to the IoT application. Collaborative-Aware Services act on top of Information Aggregation Services and use the obtained data to make decision and react accordingly. Ubiquitous Services, however, aim to provide Collaborative-Aware Services *anytime* they are needed to *anyone* who needs them *anywhere*. With this categorization, we review some applications of the IoT in the following paragraphs. The ultimate goal of all IoT applications is to reach the level of ubiquitous services. However, this end is not achievable easily since there are a lot of difficulties and challenges that have to be addressed. Most of the existing applications provide identity-related, information aggregation, and collaborative-aware services. Smart healthcare and smart grids fall into the information aggregation category and smart home, smart buildings, intelligent transportation systems (ITS), and industrial automation are closer to the collaborative-aware category.

TABLE II
BUILDING BLOCKS AND TECHNOLOGIES OF THE IOT

IoT Elements		Samples
Identification	**Naming**	EPC, uCode
	Addressing	IPv4, IPv6
Sensing		Smart Sensors, Wearable sensing devices, Embedded sensors, Actuators, RFID tag
Communication		RFID, NFC, UWB, Bluetooth, BLE, IEEE 802.15.4, Z-Wave, WiFi, WiFiDirect, , LTE-A
Computation	**Hardware**	SmartThings, Arduino, Phidgets, Intel Galileo, Raspberry Pi, Gadgeteer, BeagleBone, Cubieboard, Smart Phones
	Software	OS (Contiki, TinyOS, LiteOS, Riot OS, Android); Cloud (Nimbits, Hadoop, etc.)
Service		Identity-related (shipping), Information Aggregation (smart grid), Collaborative-Aware (smart home), Ubiquitous (smart city)
Semantic		RDF, OWL, EXI

Smart home [43] IoT services contribute to enhancing the personal life-style by making it easier and more convenient to monitor and operate home appliances and systems (e.g., air conditioner, heating systems, energy consumption meters, etc.) remotely. For example, a smart home can automatically close the windows and lower the blinds of upstairs windows based on the weather forecast. Smart homes are required to have regular interaction with their internal and external environments [44].

The internal environment may include all the home appliances and devices that are Internet-connected while the external environment consists of entities that are not in control of the smart home such as smart grid entities.

Smart buildings connect *building automation systems* (BAS) to the Internet [45]. BAS allows to control and manage different building devices using sensors and actuators such as HVAC, lighting and shading, security, safety, entertainment, etc. Furthermore, BAS can help to enhance energy consumption and maintenance of buildings. For example, a blinking dishwasher or cooling/heating system can provide indications when there is a problem that needs to be checked and solved. Thus, maintenance requests can be sent out to a contracted company without any human intervention.

Intelligent transportation systems (ITS) or *Transportation Cyber-Physical Systems* (T-CPS) represent integration between computation and communication to monitor and control the transportation network [46], [47]. ITS aims to achieve better reliability, efficiency, availability and safety of the transportation infrastructure. ITS employs four main components, namely: vehicle subsystem (consists of GPS, RFID reader, OBU, and communication), station subsystem (road-side equipment), ITS monitoring center and security subsystem. Moreover, connected vehicles are becoming more important with the aim to make driving more reliable, enjoyable and efficient [48], [49]. For instance, Audi became the first automaker with a license for self-driving in Nevada [50]. Google is another pioneer in this area [51]. Also, in December 2013, Volvo announced its self-driving car to drive about 30 miles in busy roads in Gothenburg, Sweden [52]. Earlier this year, the USDOT announced that it would chart a regulatory path that would require all new automobiles to be equipped with vehicle-to-vehicle (V2V) communications systems sometime in the next several years.

Industrial automation [53], [54], is computerizing robotic devices to complete manufacturing tasks with a minimal human involvement. It allows a group of machines to produce products quickly and more accurately based on four elements: transportation, processing, sensing and communication. The IoT is utilized in industrial automation to control and monitor production machines' operations, functionalities, and productivity rate through the Internet. For instance, if a particular production machine encounters a sudden issue, an IoT system sends a maintenance request immediately to the maintenance department to handle the fix. Furthermore, the IoT increases productivity by analyzing production data, timing and causes of production issues.

Smart healthcare plays a significant role in healthcare applications through embedding sensors and actuators in patients and their medicine for monitoring and tracking purposes. The IoT is used by clinical care to monitor physiological statuses of patients through sensors by collecting and analyzing their information and then sending analyzed patient's data remotely to processing centers to make suitable actions. For example, Masimo Radical-7 monitors the patient's status remotely and reports that to a clinical staff [55]. Recently, IBM utilized RFID technology at one of OhioHealth's hospitals to track hand washing after checking each patient [56]–[58]. That operation could be used to avoid infections that cause about 90 000 deaths and losing about $30 billion annually.

Smart grids [44], [59] utilize the IoT to improve and enhance the energy consumption of houses and buildings. Employing the IoT in smart grids helps power suppliers to control and manage resources to provide power proportionally to the population increase. For example, smart grids use the IoT to connect millions or billions of buildings' meters to the network of energy providers. These meters are used to collect, analyze, control, monitor, and manage energy consumption. The IoT enables energy providers to improve their services to meet consumers' needs. Also, utilizing the IoT in the smart grid reduces the potential failures, increases efficiency and improves quality of services.

A *smart city* which could be seen as an application of ubiquitous services, aims to improve the quality of life in the city by making it easier and more convenient for the residents to find information of interest [60], [61]. In a smart city environment, various systems based on smart technologies are interconnected to provide required services (health, utilities, transportation, government, homes and buildings).

F. Semantics

Semantic in the IoT refers to the ability to extract knowledge smartly by different machines to provide the required services. Knowledge extraction includes discovering and using resources and modeling information. Also, it includes recognizing and analyzing data to make sense of the right decision to provide the exact service [62]. Thus, semantic represents the brain of the IoT by sending demands to the right resource. This requirement is supported by Semantic Web technologies such as the Resource Description Framework (RDF) and the Web Ontology Language (OWL). In 2011, the *World Wide Web consortium* (W3C) adopted the *Efficient XML Interchange* (EXI) format as a recommendation [63].

EXI is important in the context of the IoT because it is designed to optimize XML applications for resource-constrained environments. Furthermore, it reduces bandwidth needs without affecting related resources such as battery life, code size, energy consumed for processing, and memory size. EXI converts XML messages to binary to reduce the needed bandwidth and minimize the required storage size.

Remarks: In this section, the main components of the IoT were identified along with their related standards, technologies and realizations. The variety of standards and technologies in these elements and the way they should interoperate is a main challenge that can impede the development of IoT applications. The heterogeneity of the IoT elements needs a thorough solution to make ubiquitous IoT services a reality. Section VIII addresses this problem by proposing an architectural model that alleviates the interoperability issues caused by the diversity of protocols and technologies utilized in the context of the IoT.

V. IoT Common Standards

Many IoT standards are proposed to facilitate and simplify application programmers' and service providers' jobs. Different groups have been created to provide protocols in support of the IoT including efforts led by the World Wide Web Consortium (W3C), *Internet Engineering Task Force* (IETF), EPCglobal,

TABLE III
STANDARDIZATION EFFORTS IN SUPPORT OF THE IOT

Application Protocol		DDS	CoAP	AMQP	MQTT	MQTT-SN	XMPP	HTTP REST
Service Discovery		mDNS				DNS-SD		
Infrastructure Protocols	Routing Protocol	RPL						
	Network Layer	6LoWPAN				IPv4/IPv6		
	Link Layer	IEEE 802.15.4						
	Physical/ Device Layer	LTE-A	EPCglobal	IEEE 802.15.4		Z-Wave		
Influential Protocols		IEEE 1888.3, IPSec				IEEE 1905.1		

Institute of Electrical and Electronics Engineers (IEEE) and the *European Telecommunications Standards Institute* (ETSI). Table III, provides a summary of the most prominent protocols defined by these groups. In this paper, we classify the IoT protocols into four broad categories, namely: application protocols, service discovery protocols, infrastructure protocols and other influential protocols. However, not all of these protocols have to be bundled together to deliver a given IoT application. Moreover, based on the nature of the IoT application, some standards may not be required to be supported in an application. In the following subsections, we provide an overview of some of the common protocols in these categories and their core functionality.

A. Application Protocols

1) Constrained Application Protocol (CoAP): The IETF Constrained RESTful Environments (CoRE) working group created CoAP, which is an application layer protocol [64], [65] for IoT applications. The CoAP defines a web transfer protocol based on *REpresentational State Transfer* (REST) on top of HTTP functionalities. REST represents a simpler way to exchange data between clients and servers over HTTP [66]. REST can be seen as a cacheable connection protocol that relies on stateless client-server architecture. It is used within mobile and social network applications and it eliminates ambiguity by using HTTP *get*, *post*, *put*, and *delete* methods. REST enables clients and servers to expose and consume web services like the Simple Object Access Protocol (SOAP) but in an easier way using Uniform Resource Identifiers (URIs) as nouns and HTTP *get*, *post*, *put*, and *delete* methods as verbs. REST does not require XML for message exchanges. Unlike REST, CoAP is bound to UDP (not TCP) by default which makes it more suitable for the IoT applications. Furthermore, CoAP modifies some HTTP functionalities to meet the IoT requirements such as low power consumption and operation in the presence of lossy and noisy links. However, since CoAP has been designed based on REST, conversion between these two protocols in REST-CoAP proxies is straightforward. The overall functionality of CoAP protocol is demonstrated in Fig. 5.

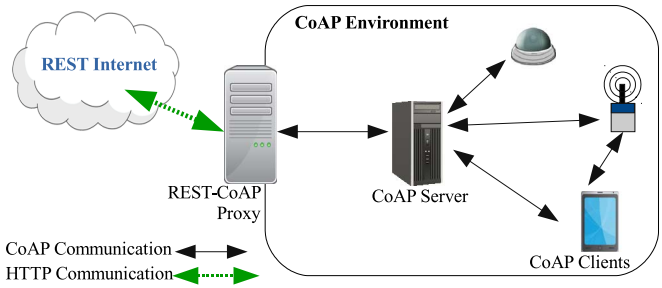

Fig. 5. CoAP functionality.

CoAP aims to enable tiny devices with low power, computation and communication capabilities to utilize RESTful interactions. CoAP can be divided into two sub-layers, namely: the messaging sub-layer and the request/response sub-layer. The messaging sub-layer detects duplications and provides reliable communication over the UDP transport layer using exponential backoff since UDP does not have a built-in error recovery mechanism. The request/response sub-layer on the other hand handles REST communications. CoAP utilizes four types of messages: confirmable, non-confirmable, reset and acknowledgement. Reliability of CoAP is accomplished by a mix of confirmable and non-confirmable messages. It also employs four modes of responses as illustrated in Fig. 6. The separate response mode is used when the server needs to wait for a specific time before replying to the client. In CoAP's non-confirmable response mode, the client sends data without waiting for an ACK message, while message IDs are used to detect duplicates. The server side responds with a RST message when messages are missed or communication issues occur. CoAP, as in HTTP, utilizes methods such as GET, PUT, POST and DELETE to achieve Create, Retrieve, Update and Delete (CRUD) operations. For example, the GET method can be used by a server to inquire the client's temperature using the piggybacked response mode. The client sends back the temperature if it exists; otherwise, it replies with a status code to indicate that the requested data is not found. CoAP uses a simple and small format to encode messages. The first and fixed part of each message is four bytes of header. Then a token value may appear whose length ranges from zero to eight bytes. The token value is used for correlating requests and responses. The options and payload are the next optional fields. A typical CoAP message can be between 10 to 20 bytes [67]. The message format of CoAP packets is depicted in Fig. 7 [64].

The fields in the header are as follows: Ver is the version of CoAP, T is the type of Transaction, OC is Option count, and Code represents the request method (1–10) or response code (40–255). For example the code for GET, POST, PUT, and DELETE is 1, 2, 3, and 4, respectively. The Transaction ID in the header is a unique identifier for matching the response.

Some of the important features provided by CoAP include [65], [68]:

- *Resource observation*: On-demand subscriptions to monitor resources of interest using publish/subscribe mechanism.
- *Block-wise resource transport*: Ability to exchange transceiver data between the client and the server without

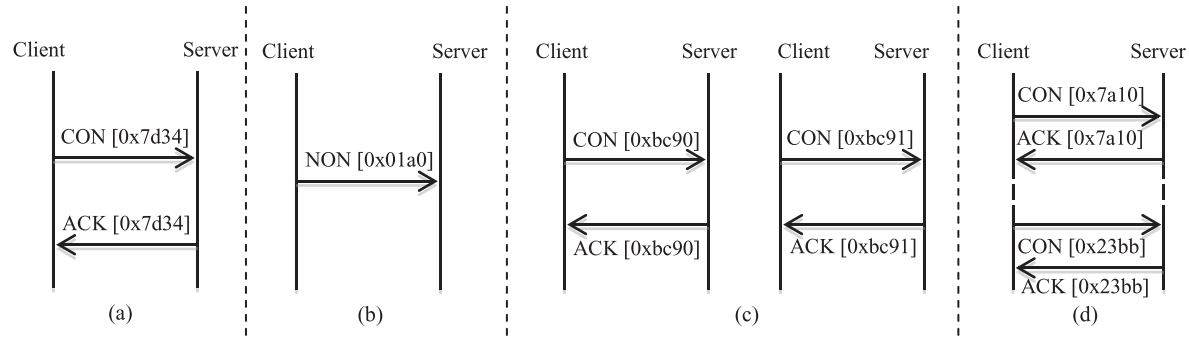

Fig. 6. CoAP message types [64]. (a) Confirmable. (b) Non-confirmable. (c) Piggybacked responses. (d) Separate response.

0 1	2 3	4 5 6 7	8	16 31
Ver	T	OC	Code	Message ID
Token (if any)				
Options (if any)				
Payload (if any)				

Fig. 7. CoAP message format.

the need to update the whole data to reduce the communication overhead.

- *Resource discovery*: Server utilizes well-known URI paths based on the web link fields in CoRE link format to provide resource discovery for the client.
- *Interacting with HTTP*: Flexibility of communicating with several devices because the common REST architecture enables CoAP to interact easily with HTTP through a proxy.
- *Security*: CoAP is a secure protocol since it is built on top of *datagram transport layer security* (DTLS) to guarantee integrity and confidentiality of exchanged messages.

As an example of how an application protocol works in an IoT environment, we provided a sample code in [69]. Since the cloud service for this project, Nimbits, does not support CoAP currently, we used HTTP REST to integrate with Nimbits.

2) Message Queue Telemetry Transport (MQTT): MQTT is a messaging protocol that was introduced by Andy Stanford-Clark of IBM and Arlen Nipper of Arcom (now Eurotech) in 1999 and was standardized in 2013 at OASIS [70]. MQTT aims at connecting embedded devices and networks with applications and middleware. The connection operation uses a routing mechanism (one-to-one, one-to-many, many-to-many) and enables MQTT as an optimal connection protocol for the IoT and M2M.

MQTT utilizes the publish/subscribe pattern to provide transition flexibility and simplicity of implementation as depicted in Fig. 8. Also, MQTT is suitable for resource constrained devices that use unreliable or low bandwidth links. MQTT is built on top of the TCP protocol. It delivers messages through three levels of QoS. Two major specifications exist for MQTT: MQTT v3.1 and MQTT-SN [71] (formerly known as MQTT-S) V1.2. The latter was defined specifically for sensor networks and defines a UDP mapping of MQTT and adds broker support for indexing topic names. The specifications provide three elements: connection semantics, routing, and endpoint.

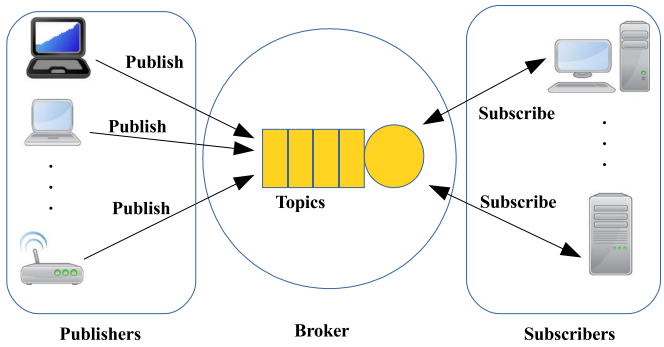

Fig. 8. The architecture of MQTT.

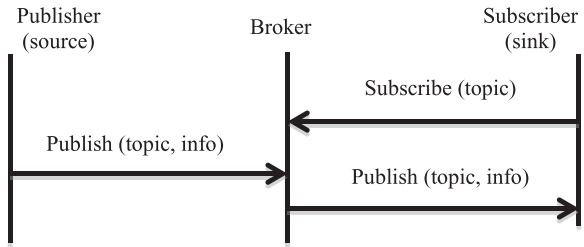

Fig. 9. Publish/subscribe process utilized by MQTT [70].

MQTT simply consists of three components, subscriber, publisher, and broker. An interested device would register as a subscriber for specific topics in order for it to be informed by the broker when publishers publish topics of interest. The publisher acts as a generator of interesting data. After that, the publisher transmits the information to the interested entities (subscribers) through the broker. Furthermore, the broker achieves security by checking authorization of the publishers and the subscribers [71]. Numerous applications utilize the MQTT such as health care, monitoring, energy meter, and Facebook notification. Therefore, the MQTT protocol represents an ideal messaging protocol for the IoT and M2M communications and is able to provide routing for small, cheap, low power and low memory devices in vulnerable and low bandwidth networks. Fig. 9 illustrates the publish/subscribe process utilized by MQTT and Fig. 10 shows the message format used by the MQTT protocol [70]. The first two bytes of message are fixed header. In this format, the value of the Message Type field indicates a variety of messages including CONNECT (1), CONNACK (2), PUBLISH (3), SUBSCRIBE (8) and so on. The DUP flag indicates that the massage is duplicated and that the receiver may have

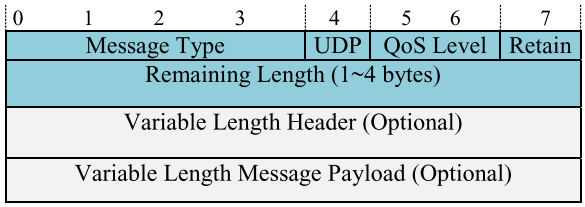

Fig. 10. MQTT message format.

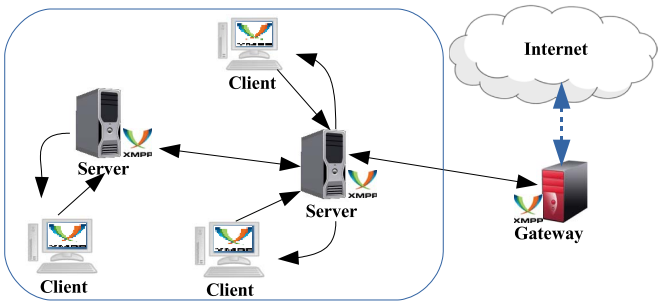

Fig. 11. Communications in XMPP.

received it before. Three levels of QoS for delivery assurance of Publish messages are identified by the QoS Level field. The Retain field informs the server to retain the last received Publish message and submit it to new subscribers as a first message. The Remaining Length field shows the remaining length of the message i.e., the length of the optional parts.

3) Extensible Messaging and Presence Protocol (XMPP): XMPP is an IETF instant messaging (IM) standard that is used for multi-party chatting, voice and video calling and telepresence [72]. XMPP was developed by the Jabber open source community to support an open, secure, spam free and decentralized messaging protocol. XMPP allows users to communicate with each other by sending instant messages on the Internet no matter which operating system they are using. XMPP allows IM applications to achieve authentication, access control, privacy measurement, hop-by-hop and end-to-end encryption, and compatibility with other protocols. Fig. 11 illustrates the overall behavior of XMPP protocol, in which gateways can bridge between foreign messaging networks [73].

Many XMPP features make it a preferred protocol by most IM applications and relevant within the scope of the IoT. It runs over a variety of Internet-based platforms in a decentralized fashion. XMPP is secure and allows for the addition of new applications on top of the core protocols. XMPP connects a client to a server using a stream of XML stanzas. An XML stanza represents a piece of code that is divided into three components: message, presence, and iq (info/query) (See Fig. 12 [72]). Message stanzas identify the source (from) and destination (to) addresses, types, and IDs of XMPP entities that utilize a push method to retrieve data. A message stanza fills the subject and body fields with the message title and contents. The presence stanza shows and notifies customers of status updates as authorized. The *iq* stanza pairs message senders and receivers.

The text based communication in XMPP using XML imposes a rather high network overhead. One solution to this problem is compressing XML streams using EXI [63] which is addressed in [74].

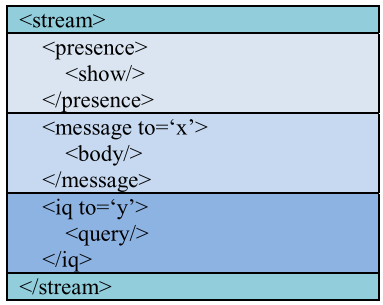

Fig. 12. Structure of XMPP stanza.

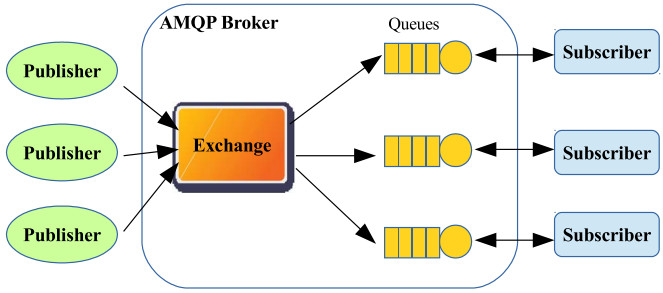

Fig. 13. Publish/subscribe mechanism of AMQP.

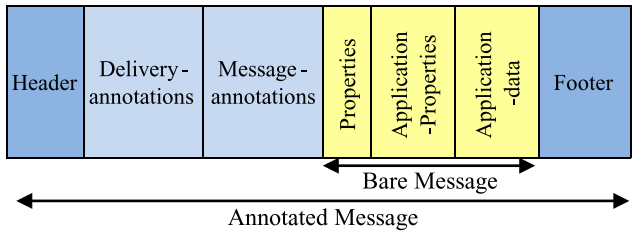

Fig. 14. AMQP message format.

4) Advanced Message Queuing Protocol (AMQP): AMQP [75] is an open standard application layer protocol for the IoT focusing on message-oriented environments. It supports reliable communication via message delivery guarantee primitives including at-most-once, at-least-once and exactly once delivery. AMQP requires a reliable transport protocol like TCP to exchange messages.

By defining a wire-level protocol, AMQP implementations are able to interoperate with each other. Communications are handled by two main components as depicted in Fig. 13: exchanges and message queues. Exchanges are used to route the messages to appropriate queues. Routing between exchanges and message queues is based on some pre-defined rules and conditions. Messages can be stored in message queues and then be sent to receivers. Beyond this type of point-to-point communication, AMQP also supports the publish/subscribe communications model.

AMQP defines a layer of messaging on top of its transport layer. Messaging capabilities are handled in this layer. AMQP defines two types of messages: bare massages that are supplied by the sender and annotated messages that are seen at the receiver. In Fig. 14 the message format of AMQP is shown [75]. The header in this format conveys the delivery parameters including durability, priority, time to live, first acquirer, and delivery count.

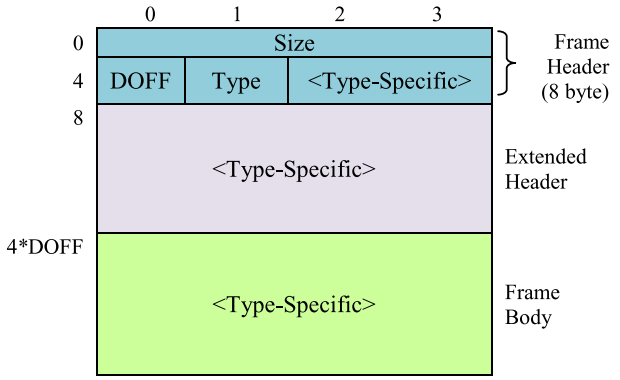

Fig. 15. AMQP frame format.

The transport layer provides the required extension points for the messaging layer. In this layer, communications are frame-oriented. The structure of AMQP frames is illustrated in Fig. 15 [75]. The first four bytes show the frame size. DOFF (Data Offset) gives the position of the body inside the frame. The Type field indicates the format and purpose of the frame. For example, 0x00 is used to show that the frame is an AMQP frame or type code 0x01 represents a SASL frame.

5) Data Distribution Service (DDS): Data Distribution Service (DDS) is a publish-subscribe protocol for real-time M2M communications that has been developed by Object Management Group (OMG) [76]. In contrast to other publish-subscribe application protocols like MQTT or AMQP, DDS relies on a broker-less architecture and uses multicasting to bring excellent Quality of Service (QoS) and high reliability to its applications. Its broker-less publish-subscribe architecture suits well to the real-time constraints for IoT and M2M communications. DDS supports 23 QoS policies by which a variety of communication criteria like security, urgency, priority, durability, reliability, etc. can be addressed by the developer.

DDS architecture defines two layers: Data-Centric Publish-Subscribe (DCPS) and Data-Local Reconstruction Layer (DLRL). DCPS is responsible for delivering the information to the subscribers. DLRL on the other hand, is an optional layer and serves as the interface to the DCPS functionalities. It facilitates the sharing of distributed data among distributed objects [77].

Five entities are involved with the flow of data in the DCPS layer: (1) Publisher that disseminates data; and (2) DataWriter that is used by the application to interact with the publisher about the values and changes of data specific to a given type. The association of DataWriter and Publisher indicates that the application is going to publish the specified data in a provided context; (3) Subscriber that receives published data and delivers them to the application; (4) DataReader that is employed by the Subscriber to access to the received data; and (5) a Topic that is identified by a data type and a name. Topics relate DataWriters to DataReaders. Data transmission is allowed within a DDS domain which is a virtual environment for connected publishing and subscribing applications. Fig. 16 demonstrates the conceptual architecture of this protocol.

Remarks: Pairwise evaluations and comparisons of these protocols have been reported in the literature. For example, [78] compares the performance of MQTT and CoAP in terms of end-to-end transmission delay and bandwidth usage. Based on their

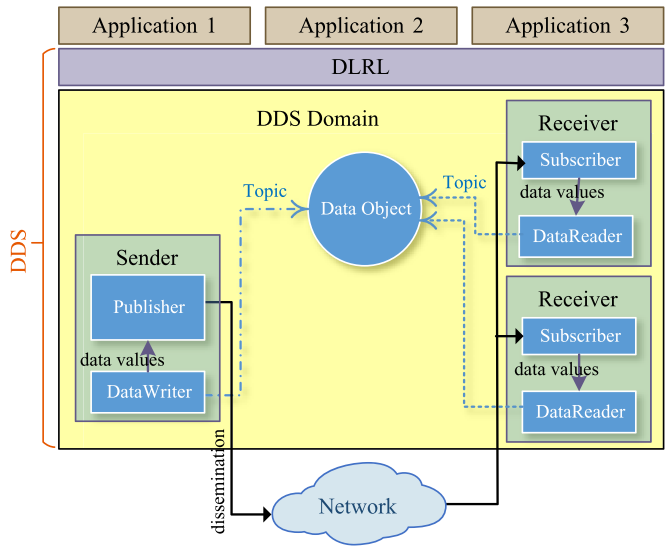

Fig. 16. The conceptual model of DDS.

results, MQTT delivers messages with lower delay than CoAP when the packet loss rate is low. In contrast, when the packet loss rate is high, CoAP outperforms MQTT. In case of small-size messages and a loss rate under 25%, CoAP outperforms MQTT in generating less extra traffic. Another research study [79] compared these two protocols in a smartphone application environment and showed that CoAP's bandwidth usage and round trip time are smaller than those of MQTT.

The performance comparison between CoAP and HTTP is investigated for energy consumption and response time in [67]. Due to its condensed header and small packet size, CoAP is more efficient than HTPP in transmission time and energy usage. The authors in [80] present an evaluation of XMPP to verify its applicability to real-time communications on the web. They assessed the performance of XMPP over HTML5 WebSocket and their results show that XMPP is an efficient option for web applications that require real-time communication. Performance evaluation of AMQP and REST is reported in [81]. To carry out their study, the authors used the average number of exchange messages between the client and the server in a specific interval to measure the performance. Under a high volume of message exchanges, AMQP demonstrated better results than RESTful web services.

An experimental evaluation of two implementations of DDS [77] points out that this protocol scales well when the number of nodes is increased.

To the best of our knowledge, there is no comprehensive evaluation of all these protocols together. However, each of these protocols may perform well in specific scenarios and environments. So it is not feasible to provide a single prescription for all IoT applications. Table IV provides a brief comparison between the common IoT application protocols. The last column in the table indicates the minimum header size required by each protocol.

B. Service Discovery Protocols

The high scalability of the IoT requires a resource management mechanism that is able to register and discover resources

TABLE IV
COMPARISON BETWEEN THE IOT APPLICATION PROTOCOLS

Application Protocol	RESTful	Transport	Publish/ Subscribe	Request/ Response	Security	QoS	Header Size (Byte)
COAP	✓	UDP	✓	✓	DTLS	✓	4
MQTT	×	TCP	✓	×	SSL	✓	2
MQTT-SN	×	TCP	✓	×	SSL	✓	2
XMPP	×	TCP	✓	✓	SSL	×	-
AMQP	×	TCP	✓	×	SSL	✓	8
DDS	×	TCP UDP	✓	×	SSL DTLS	✓	-
HTTP	✓	TCP	×	✓	SSL	×	-

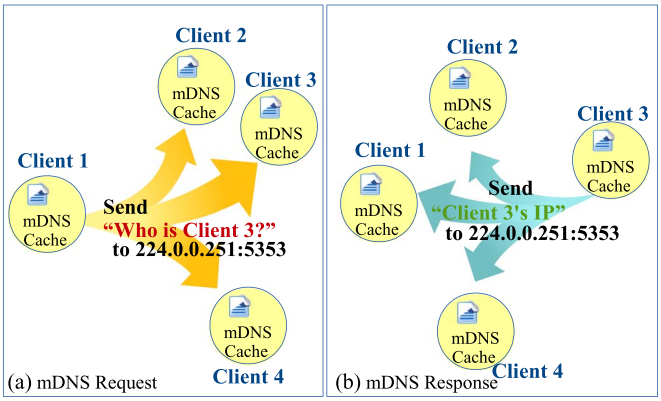

Fig. 17. Request/Response in mDNS protocol.

and services in a self-configured, efficient, and dynamic way. The most dominant protocols in this area are multicast DNS (mDNS) and DNS Service Discovery (DNS-SD) that can discover resources and services offered by IoT devices. Although these two protocols have been designed originally for resource-rich devices, there are research studies that adapt light versions of them for IoT environments [82], [83].

1) Multicast DNS (mDNS): A base service for some IoT applications like chatting is Name Resolution. mDNS is such a service that can perform the task of unicast DNS server [84]. mDNS is flexible due to the fact that the DNS namespace is used locally without extra expenses or configuration. mDNS is an appropriate choice for embedded Internet-based devices due to the facts that a) There is no need for manual reconfiguration or extra administration to manage devices; b) It is able to run without infrastructure; and c) It is able to continue working if failure of infrastructure happens.

mDNS inquires names by sending an IP multicast message to all the nodes in the local domain as shown in Fig. 17. By this query, the client asks devices that have the given name to reply back. When the target machine receives its name, it multicasts a response message which contains its IP address. All devices in the network that obtain the response message update their local cache using the given name and IP address. A practical example that utilizes mDNS service discovery can be found in [69].

2) DNS Service Discovery (DNS-SD): The pairing function of required services by clients using mDNS is called

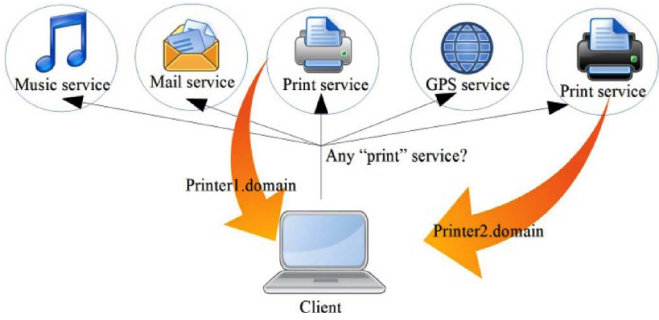

Fig. 18. Discovering print service by DNS-SD.

DNS-based service discovery (DNS-SD). Using this protocol, clients can discover a set of desired services in a specific network by employing standard DNS messages. Fig. 18 provides a visual illustration of how this protocol works. DNS-SD, like mDNS, is part of the zero configuration aids to connect machines without external administration or configuration [85].

Essentially, DNS-SD utilizes mDNS to send DNS packets to specific multicast addresses through UDP. There are two main steps to process Service Discovery: finding host names of required services such as printers and pairing IP addresses with their host names using mDNS. Finding host names is important because IP addresses might change, whereas names do not. The Pairing function multicasts network attachments details like IP, and port number to each related host. Using DNS-SD, the instance names in the network can be kept constant as long as possible to increase trust and reliability. For example, if some clients know and use a specific printer today, they will be enabled to use it thereafter without any problems.

Remarks: IoT needs some sort of architecture without dependency on a configuration mechanism. In such an architecture, smart devices can join the platform or leave it without affecting the behavior of the whole system. mDNS and DNS-SD can smooth this way of development. However, the main drawback of these two protocols is the need for caching DNS entries especially when it comes to resource-constrained devices. However, timing the cache for a specific interval and depleting it can solve this issue. Bonjour and Avahi are two well-known implementations covering both mDNS and DNS-SD.

C. Infrastructure Protocols

1) Routing Protocol for Low Power and Lossy Networks (RPL): The IETF routing over low-power and lossy links (ROLL) working group standardized a link-independent routing protocol based on IPv6 for resource-constrained nodes called RPL [86], [87]. RPL was created to support minimal routing requirements through building a robust topology over lossy links. This routing protocol supports simple and complex traffic models like multipoint-to-point, point-to-multipoint and point-to-point.

A Destination Oriented Directed Acyclic Graph (DODAG) represents the core of RPL that shows a routing diagram of nodes. The DODAG refers to a directed acyclic graph with a single root as shown in Fig. 19. Each node in the DODAG is aware of its parents but they have no information about related children. Also, RPL keeps at least one path for each node to the root and preferred parent to pursue a faster path to increase performance.

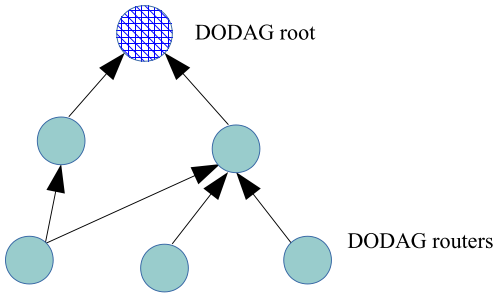

Fig. 19. DODAG topology.

In order to maintain the routing topology and to keep the routing information updated, RPL uses four types of control messages. The most important message is DODAG Information Object (DIO) which is used to keep the current rank (level) of the node, determine the distance of each node to the root based on some specific metrics, and choose the preferred parent path. The other message type is Destination Advertisement Object (DAO). RPL provides upward traffic as well as downward traffic support using DAO messages by which it unicasts destination information towards the selected parents. The third message is DODAG Information Solicitation (DIS) which is used by a node to acquire DIO messages from a reachable adjacent node. The last message type is DAO Acknowledgment (DAO-ACk) which is a response to a DAO message and is sent by a DAO recipient node like a DAO parent or DODAG root [88].

A DODAG starts to be formed when the root, the only node which consist the DODAG, starts sending its location using DIO message to all Low-power Lossy Network (LLN) levels. At each level, recipient routers register parent path and participation paths for each node. They in turn propagate their DIO messages and the whole DODAG gradually is built. When the DODAG is constructed, the *preferred parent* obtained by a router stands as a default path towards the root (upward routes). The root can also store the destination prefixes obtained by DIOs of other routers in its DIO messages to have upward routes. To support downward routes, routers should emit and propagate DAO messages by unicasting to the root through parents. These messages identify the corresponding node of a route prefix as well as crossing route.

RPL routers work under one of two modes of operation (MOP): Non-Storing or Storing modes. In *Non-Storing* mode, RPL routes messages move towards lower levels based on IP source routing, whereas in *Storing mode*, downward routing is based on destination IPv6 addresses [88].

The sample code for Wireless Sensor Network presented in [69] utilizes ContikiRPL as an implementation of the RPL protocol for routing the packets.

2) 6LowPAN: Low power Wireless Personal Area Networks (WPANs) which many IoT communications may rely on have some special characteristics different from former link layer technologies like limited packet size (e.g., maximum 127 bytes for IEEE 802.15.4), various address lengths, and low bandwidth [89]–[91]. So, there was a need to make an adaptation layer that fits IPv6 packets to the IEEE 802.15.4 specifications. The IETF 6LoWPAN working group developed such a standard in 2007. 6LoWPAN is the specification of mapping services required by the IPv6 over Low power WPANs to maintain an IPv6 network [89]. The standard provides header compression to reduce the transmission overhead, fragmentation to meet the IPv6 Maximum Transmission Unit (MTU) requirement, and forwarding to link-layer to support multi-hop delivery [91].

Datagrams enveloped by 6LoWPAN are followed by a combination of some headers. These headers are of four types which are identified by two bits [89]: (00) NO 6LoWPAN Header, (01) Dispatch Header, (10) Mesh Addressing, and (11) Fragmentation. By NO 6LoWPAN Header, packets that do not accord to the 6LoWPAN specification will be discarded. Compression of IPv6 headers or multicasting is performed by specifying Dispatch header. Mesh Addressing header identifies those IEEE 802.15.4 packets that have to be forwarded to the link-layer. For datagrams whose lengths exceed a single IEEE 802.15.4 frame, Fragmentation header should be used.

6LoWPAN removes a lot of IPv6 overheads in such a way that a small IPv6 datagram can be sent over a single IEEE 802.15.4 hop in the best case. It can also compress IPv6 headers to two bytes [91].

3) IEEE 802.15.4: The IEEE 802.15.4 protocol was created to specify a sub-layer for Medium Access Control (MAC) and a physical layer (PHY) for low-rate wireless private area networks (LR-WPAN) [33]. Due to its specifications such as low power consumption, low data rate, low cost, and high message throughput, it also is utilized by the IoT, M2M, and WSNs. It provides a reliable communication, operability on different platforms, and can handle a large number of nodes (about 65 k). It also provides a high level of security, encryption and authentication services. However, it does not provide QoS guarantees. This protocol is the base for the ZigBee protocol as they both focus on offering low data rate services on power constrained devices and they build a complete network protocol stack for WSNs.

IEEE 802.15.4 supports three frequency channel bands and utilizes a direct sequence spread spectrum (DSSS) method. Based on the used frequency channels, the physical layer transmits and receives data over three data rates: 250 kbps at 2.4 GHz, 40 kbps at 915 MHz, and 20 kbps at 868 MHz. Higher frequencies and wider bands provide high throughput and low latency whereas lower frequencies provide better sensitivity and cover larger distances. To reduce potential collisions, IEEE 802.15.4 MAC utilizes the CSMA/CA protocol.

The IEEE 802.15.4 standard supports two types of network nodes: Full and Reduced Function Devices. The *full function device* (FFD) can serve as a *personal area network* (PAN) coordinator or just as a normal node. A coordinator is responsible for creation, control and maintenance of the network. FFDs can store a routing table within their memory and implement a full MAC. They also can communicate with any other devices using any available topology as seen in Fig. 20. The *reduced function devices* (RFD) on the other hand, are very simple nodes with restricted resources. They can only communicate with a coordinator, and are limited to a star topology.

Standard topologies to form IEEE 802.15.4 networks are star, peer-to-peer (mesh), and cluster-tree (See Fig. 20). The star topology contains at least one FFD and some RFDs. The FFD who works as a PAN coordinator should be located at the center of topology and aims to manage and control all the other

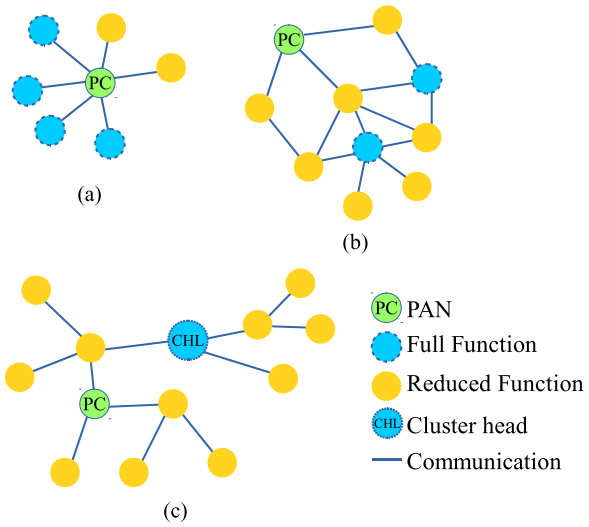

Fig. 20. IEEE 802.15.4 topologies [33]. (a) Star. (b) Peer-to-peer. (c) Cluster-tree.

nodes in the network. The peer-to-peer topology contains a PAN coordinator and other nodes communicate with each other in the same network or through intermediate nodes to other networks. A cluster-tree topology is a special case of the peer-to-peer topology and consists of a PAN coordinator, a cluster head and normal nodes.

4) Bluetooth Low Energy: Bluetooth Low-Energy (BLE) or Bluetooth Smart uses a short range radio with a minimal amount of power to operate for a longer time (even for years) compared to its previous versions. Its range coverage (about 100 meter) is ten times that of the classic Bluetooth while its latency is 15 times shorter [92]. BLE can be operated by a transmission power between 0.01 mW to 10 mW. With these characteristics, BLE is a good candidate for IoT applications [93].

The BLE standard has been developed rapidly by smartphone makers and is now available in most smartphone models. The feasibility of using this standard has been demonstrated in vehicle-to-vehicle communications [92] as well as wireless sensor networks [94]. Compared to ZigBee, BLE is more efficient in terms of energy consumption and the ratio of transmission energy per transmitted bit [95].

BLE's network stack is as follows: In the lowest level of BLE's stack there is a Physical (PHY) Layer which transmits and receives bits. Over the PHY, the Link Layer services including medium access, connection establishment, error control, and flow control are provided. Then, the Logical Link Control and Adaptation Protocol (L2CAP) provides multiplexing for data channels, fragmentation and reassembly of larger packets. The other upper layers are Generic Attribute protocol (GATT) which provides efficient data collection from sensors, and Generic Access Profile (GAP) that allows the application for configuration and operation in different modes such as advertising or scanning, and connection initiation and management [95].

BLE allows devices to operate as masters or slaves in a star topology. For the discovery mechanism, slaves send advertisements over one or more of dedicated advertisement channels. To be discovered as a slave, these channels are scanned by the master. Except for the time when two devices are exchanging data, they are in sleep mode for the rest of the time.

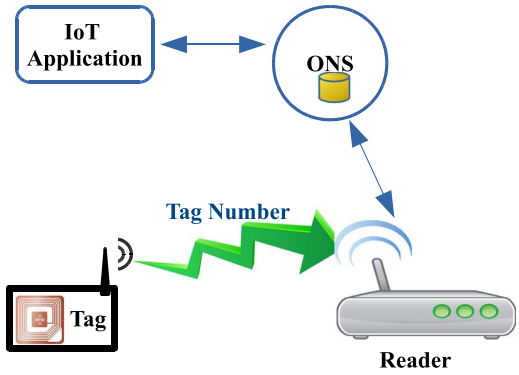

Fig. 21. RFID system.

5) EPCglobal: The Electronic Product Code (EPC) is a unique identification number which is stored on an RFID tag and is used basically in the supply chain management to identify items. EPCglobal as the original organization responsible for the development of EPC, manages EPC and RFID technology and standards. The underlying architecture uses Internet-based RFID technologies along with cheap RFID tags and readers to share product information [96]. This architecture is recognized as a promising technique for the future of the IoT because of its openness, scalability, interoperability and reliability beyond its support to the primary IoT requirements such as objects IDs and service discovery [97].

EPCs are classified into four types: 96-bit, 64-bit (I), 64-bit (II) and 64-bit (III). All types of 64-bit EPCs support about 16 000 companies with unique identities and cover 1 to 9 million types of products and 33 million serial numbers for each type. The 96-bit type supports about 268 million companies with unique identities, 16 million classes of products and 68 billion serial numbers for each class.

The RFID system can be divided into two main components: radio signal transponder (tag) and tag reader. The tag consists of two components: a chip to store the unique identity of the object and an antenna to allow the chip to communicate with the tag reader using radio waves. The tag reader generates a radio frequency field to identify objects through reflected radio waves of the tag. RFID works by sending the tag's number to the tag reader using radio waves as is shown in Fig. 21. After that, the reader passes that number to a specific computer application called the *Object-Naming Services* (ONS). An ONS looks up the tag's details from a database such as when and where it was manufactured.

EPCglobal Network can be divided into five components: EPC, ID system, EPC Middleware, Discovery Services, and EPC Information Services. EPC as a unique number to objects, consists of four parts as seen in Fig. 22 [96], [98].

The ID system links the EPC identities to a database using an EPC reader through the middleware. The discovery service is a mechanism of EPCglobal to find the required data by the tags using the ONS.

The second generation of EPC tags (called Gen 2 tags), launched in mid 2006 aims to globally cover various company products. A Gen 2 tag provides better services for customers than the first generation of tags (known as passive RFID) based on features like: interoperability under heterogeneous objects,

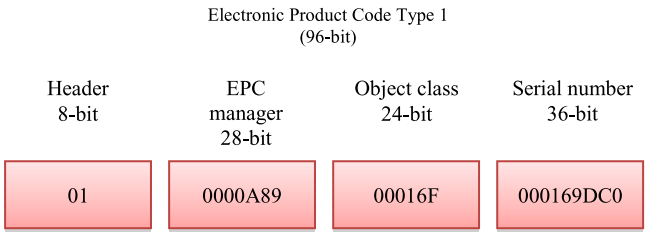

Fig. 22. EPC 96-bit tag parts [96].

TABLE V
EPC TAG CLASSES [96]

EPC	Description	Tag type	Functionality
0	Read only	Passive	Write once and read many times
1	Write once and read only	Passive	Write once and read many times
2	Read/Write	Passive	Read/write many times
3	Read/Write	Semi-Passive	Attached within sensor
4	Read/Write	Active	Attached within sensor while providing a radio wave field to communicate with reader

high performance for all requirements, high reliability, and cheap tags and readers. The different classes of EPC tags are summarized in Table V.

6) LTE-A (Long Term Evolution—Advanced): LTE-A encompasses a set of cellular communication protocols that fit well for Machine-Type Communications (MTC) and IoT infrastructures especially for smart cities where long term durability of infrastructure is expected [99]. Moreover, it outperforms other cellular solutions in terms of service cost and scalability.

At the physical layer, LTE-A uses orthogonal frequency division multiple access (OFDMA) by which the channel bandwidth is partitioned into smaller bands called physical resource blocks (PRB). LTE-A also employs a multiple-component-carrier (CC) spread spectrum technique that allows having up to five 20-MHz bands. The architecture of LTE-A network relies on two essential parts. The first one is the Core Network (CN) which controls mobile devices and deals with IP packet flows. The other part is the Radio Access Network (RAN) which handles wireless communication and radio access and establishes user plane and control plane protocols. RAN mainly consists of base stations (also called evolved NodeBs) that are connected to each other by the X2 interface. The RAN and the CN are connected through the S1 interface. Mobile or MTC devices can connect to base stations directly or through MTC gateway (MTCG). They also can have direct communication with other MTC devices.

However, this protocol has its challenges such as high network congestion when a large number of devices are accessing the network. Another challenge, QoS can be compromised when MTC devices try to access the network via eNB or MTCG selection. These problems have been investigated in [99] along with a solution based on reinforcement learning for eNB selection. In [100], the authors also analyzed the performance of MTC communications with a queuing model as well as eNB selection. Based on their results, when the MTC devices remain inactive for a longer time instead of being active, the throughput of the MTC devices will be improved due to lower contention in the network.

7) Z-Wave: Z-Wave as a low-power wireless communication protocol for Home Automation Networks (HAN) has been used widely in the remote control applications in smart homes as well as small-size commercial domains [101]. This protocol was initially developed by ZenSys (currently Sigma Designs) and later was employed and improved by Z-Wave Alliance. Z-Wave covers about 30 meters point-to-point communication and is specified for applications that need tiny data transmission like light control, household appliance control, smart energy and HVAC, access control, wearable health care control, and fire detection. Z-Wave operates in ISM bands (around 900 MHz) and allows transmission rate of 40 kbps. The recent versions also support up to 200 kbps. Its MAC layer benefits from a collision avoidance mechanism. Reliable transmission is possible in this protocol by optional ACK messages. In its architecture, there are controller and slave nodes. Controllers manage the slaves by sending commands to them. For routing purposes, a controller keeps a table of the whole network topology. Routing in this protocol is performed by source routing method in which a controller submits the path inside a packet.

Remarks: In this section, we reviewed some prominent infrastructure protocols which are needed to establish the underlying communication needed by IoT applications. Here, we review some efficiency and performance aspects of these standards and highlight some of the research studies that evaluated these protocols. In [88], an evaluation of RPL for low-power and lossy networks has been conducted in which several problems are identified including: under specification, incompatibility of modes of operation in storing and non-storing modes, and loops. Another performance analysis of RPL is reported in [102] that identifies fast network set-up and bounded communication delays as its effectiveness, while high overhead is a potential drawback. [103] also has reported some unreliability problems with RPL due to the lack of the complete knowledge of the link qualities. Therefore, since routing is a key element of IoT infrastructure and many other parameters of the IoT systems like reliability, scalability and performance strongly depend on this technology, there is a need for more investigation on improvements and optimizations of routing protocols to meet the IoT requirements.

Analyzing the performance of 6LoWPAN in wireless sensor networks [104] using a point-to-point communication test-bed shows an increase in the round trip delay when the size of the ICMP payload is increased. Some other problems have been reported for a 6LoWPAN gateway such as high rate of packet loss, and ease of interference [105].

Beyond the lower power consumption that BLE demonstrated compared to IEEE 802.15.4 [95], the work in [106] investigated the performance of IEEE 802.15.4 against IEEE 802.11ah (which is a candidate standard for IoT and M2M that is currently in preliminary stages) in terms of throughput and energy usage. Their results show that the IEEE 802.11ah achieves better throughput than IEEE 802.15.4 in both idle and non-idle channels. On the other hand, the energy consumption of the

TABLE VI
CHARACTERISTICS OF PHY PROTOCOLS

PHY Protocol	Spreading Technique	Radio Band (MHz)	MAC Access	Data Rate (bps)	Scalability
IEEE 802.15.4	DSSS	868/ 915/ 2400	TDMA, CSMA/CA	20/ 40/ 250 K	65K nodes
BLE	FHSS	2400	TDMA	1024 K	5917 slaves
EPCglobal	DS-CDMA	860~960	ALOHA	Varies 5~640 K	-
LTE-A	Multiple CC	varies	OFDMA	1G (up), 500M (down)	-
Z-Wave	-	868/908/ 2400	CSMA/CA	40K	232 nodes

IEEE 802.15.4 outperforms the IEEE 802.11ah, especially in dense networks.

In order to decrease the number of collisions in the EPC Gen-2 protocol and to improve tag identification procedure as well, researchers have proposed to use code division multiple access (CDMA) technique instead of the dynamic framed slotted ALOHA technique [107]. A performance evaluation of these techniques has been carried out in [108]. They used the average number of queries and the total number of transmitted bits that are required to identify all tags in the system as their measurement factors. Their results show that the expected number of queries for tag identification using the CDMA technique is lower than the EPC Gen-2 protocol. The reason is that the CDMA technique in this case decreases the number of collisions and consequently the number of queries. But when comparing the number of transmitted bits and the time needed to identify all tags in the system, the EPC Gen-2 protocol performs better than the CDMA technique.

Z-Wave has demonstrated acceptable performance and despite being somehow more expensive than ZigBee, it has been used widely in smart home applications. Furthermore, Z-Wave applications can benefit from the flexibility and security of this protocol. Its overall performance has been reported to be superior to ZigBee's performance [109]. Table VI summarizes the main characteristics of PHY layer protocols used in IoT.

D. Other Influential Protocols

Beyond the standards and protocols that define an operational framework for IoT applications, there are some other considerations like security and interoperability that should be taken into account. Exploiting protocols and standards that cover such considerations influence the acceptability of IoT systems.

1) Security: New features and mechanisms of the IoT cannot be secured by conventional security protocols which are used on the Internet. The security protocols on the Internet are designed to work over standard non-resource constrained devices like desktop and laptop computers. Furthermore, the emergence of new protocols and architectures in support of the IoT points to new security problems and this concern should be considered in all layers of the IoT from the application to the infrastructure layers including securing data inside the resource-constrained devices.

For the secure storage of data, Codo [110] is a security solution at the file system level, designed for the Contiki OS. By caching data for bulk encryption and decryption, Codo could improve the performance for security operations. At the link layer, the IEEE 802.15.4 security protocol provides mechanisms to protect the communication between two neighboring devices [111]. At the network layer, IPSec is the mandatory security protocol for IPv6 network layer. A specification of IPsec for 6LoWPAN has been presented in [112]. Considering the multi-hop nature and the large message sizes in 6LoWPAN networks, IPsec presents more efficient communication than IEEE 802.15.4 security [111]. Since IPSec works at the network layer, it can serve any upper layer including all the application protocols that reply on TCP or UDP. On the other hand, the Transport Layer Security (TLS) is a well-known security protocol that is used to provide secure transport layer for TCP communications. Its counterpart version that secures UDP communications is called Datagram TLS (DTLS).

At the application layer, there are not many security solutions and most of them rely on security protocols at the transport layer i.e., either TLS or DTLS. Some examples of such solutions that support encryption and authentication are EventGuard [113] and QUIP [114]. Accordingly, application protocols have their own security considerations and methods. [115] presented Lithe for Secure CoAP using a compressed version of DTLS and CoAP. Most of the MQTT security solutions seem to be project specific or just leveraging TLS/SSL protocols. OASIS MQTT security subcommittee is working on a standard to secure MQTT messaging using MQTT Cybersecurity Framework [116]. XMPP uses the TLS protocol for securing its streams. It also uses a specific profile of Simple Authentication and Security Layer (SASL) protocol to authenticate streams. AMQP also uses TLS sessions as well as SASL negotiations to secure the underlying communication.

Beyond the encryption and authentication services for the IoT communications, there may be some other vulnerability to wireless attacks from inside the 6LoWPAN network and from the Internet. In such cases Intrusion Detection Systems (IDS) are required. [117] investigated using such systems in the context of IoT environments by considering routing attacks implemented on the Contiki OS.

2) Interoperability (IEEE 1905.1): The diverse devices in IoT environments rely on different network technologies. So, there is a need for interoperation of the underlying technologies. The IEEE 1905.1 standard was designed for convergent digital home networks and heterogeneous technologies [118]. It provides an abstraction layer that hides the diversity of media access control topologies as depicted in Fig. 23, while not requiring changes in the underlying layers. This protocol provides an interface to common home network technologies so that a combination of data link and physical layer protocols including IEEE 1901 over power lines, WiFi/IEEE 802.11 over the various RF bands, Ethernet over twisted pair or fiber cables, and MoCA 1.1 over coaxial cables can coexist with each other.

While the aforementioned standards help IoT to move one step forward towards enhancing the quality of life, other concerns

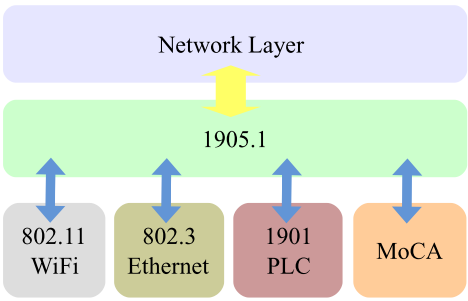

Fig. 23. Considering 1905.1 protocol in network stack.

like environmental impact of IoT devices and technologies, large scale and green deployment of IoT systems [119] remain open.

VI. QOS Criteria, IoT Challenges and Future Directions

Realizing the vision of the IoT is not an easy task due to the many challenges that need to be addressed. Examples of key challenges include availability, reliability, mobility, performance, scalability, interoperability, security, management, and trust. Addressing these challenges enables service providers and application programmers to implement their services efficiently. For example, security and privacy play a significant role in all markets globally due to the sensitivity of consumers' privacy. Also, assessing the performance of the IoT services is a key challenge [120]. Most of the identified challenges are reported in the surveys [3], [7], [8], [121], [122]. Moreover, there are some research projects like IoT6 [123], RERUM[1] and RELYonIT[2] that intend to investigate the challenges and shortcomings of the IoT and provide guidelines for solutions. In the following paragraphs, we provide a brief discussion of the key challenges faced in the development and deployment phases of the IoT and relevant research efforts and projects. Table VII presents a summary of the research efforts and projects associated with the IoT challenges under discussion.

A. Availability

Availability of the IoT must be realized in the hardware and software levels to provide anywhere and anytime services for customers. Availability of software refers to the ability of the IoT applications to provide services for everyone at different places simultaneously. Hardware availability refers to the existence of devices all the time that are compatible with the IoT functionalities and protocols. Protocols such as IPv6, 6LoWPAN, RPL, CoAP, etc., should be embedded within the single board resource constrained devices that deliver the IoT functionality. One solution to achieve high availability of IoT services is to provide redundancy for critical devices and services [124]. Moreover, there are some studies on assessing and evaluating the availability of IoT applications at the first stages of designing the system [124], [125]. Such tools can help system designers to make educated decisions to maximize the availability of their system.

[1]http://ict-rerum.eu/
[2]http://www.relyonit.eu/

TABLE VII
Projects and Research Addressing IoT Key Challenges

IoT Challenge	Projects/Protocols	Research
Architecture	IoT-A, IoT@Work, EBBITS, BETaaS, CALIPSO, VITAL, SENSEI	[3], [15], [16], [17], [18], [134], [141], [146], [147]
Availability	-	[124], [125] , [131], [148], [149]
Reliability	RERUM, RELYonIT	[124], [126], [128], [148], [150], [151], [152]
Mobility	IoT6, OpenIoT, APEC IoV	[129], [130], [131], [132], [133]
Performance	SmartSantander, RELYonIT	[67, 78, 79, 81, 95, 102, 104, 111], [120], [153], [154], [155]
Management	OMA Device Management (OMA-DM), LWM2M, NETCONF Light, Kura, MASH Platform	[136], [138], [139], [156], [157], [158], [159]
Scalability	IoT-iCore, IoT6, SENSEI	[97], [141], [142], [160]
Interoperability	IoT-iCore, PROBE-IT, OpenIoT, LinkSmart	[97], [118], [141], [143]
Security and Privacy	IETF SOLACE, BUTLER, Codo, SVELETE	[110], [111], [112], [115], [116], [117], [144], [161], [162], [163]

B. Reliability

Reliability refers to the proper working of the system based on its specification [124]. Reliability aims to increase the success rate of IoT service delivery. It has a close relationship with availability as by reliability, we guarantee the availability of information and services over time. Reliability is even more critical and has more stringent requirements when it comes to the field of emergency response applications [126]. In these systems, the critical part is the communication network which must be resilient to failures in order to realize reliable information distribution. Reliability must be implemented in software and hardware throughout all the IoT layers. In order to have an efficient IoT, the underlying communication must be reliable, because for example by an unreliable perception, data gathering, processing, and transmission can lead to long delays, loss of data, and eventually wrong decisions, which can lead to disastrous scenarios and can consequently make the IoT less dependable [127]. [126] proposes a reliability scheme at the transmission level to minimize packet losses in IoT environments. Providing services to smart devices need reliable service composition. In [128], [129], the authors exploit probabilistic model checking methods to evaluate the reliability and cost of service composition in IoT systems.

C. Mobility

Mobility is another challenge for the IoT implementations because most of the services are expected to be delivered to mobile users. Connecting users with their desired services continuously while on the move is an important premise of the IoT. Service interruption for mobile devices can occur when these devices transfer from one gateway to another. [129] proposes a resource mobility scheme that supports two modes: caching and tunneling to support service continuity. These methods allow applications to access the IoT data in the case of the temporary unavailability of resources. The enormous number of smart devices in IoT systems also requires some efficient mechanisms for mobility management. A feasible approach has been presented in [130]. In this scheme, group mobility is managed by a leader based on some similarity metric that is based on the mobility pattern of devices.

Another mobility management scheme is proposed in [131] in which the mobility of sensor nodes as well as service availability are addressed by providing a distributed service lifecycle management mechanism. This technique controls the lifecycle of web service instances that represent a sensor. Internet of Vehicles (IoV) as an emerging area of the IoT needs a precise attention to the mobility issues. [132] discusses various solutions that support mobility for vehicle-to-vehicle networking. A group mobility mechanism for mobile ad-hoc networks is presented in [133] that is inspired from birds flying in flocks.

D. Performance

Evaluating the performance of IoT services is a big challenge since it depends on the performance of many components as well as the performance of the underlying technologies. The IoT, like other systems, needs to continuously develop and improve its services to meet customers' requirements. The IoT devices need to be monitored and evaluated to provide the best possible performance at an affordable price for customers. Many metrics can be used to assess the performance of the IoT including the processing speed, communication speed, device form factor, and cost.

Performance evaluation of the individual underlying protocols and technologies like BLE [95], IEEE 802.15.4 [95], [111], RFID [120], 6LoWPAN [104], RPL [88], [102], application layer protocols [67], [78], [79], [81], and QoS [134] have been reported in the literature, but the lack of a thorough performance evaluation for IoT applications is still an open issue.

E. Management

The connection of billions or trillions of smart devices presents service providers with daunting issues to manage the Fault, Configuration, Accounting, Performance and Security (FCAPS) aspects of these devices. This management effort necessitates the development of new light-weight management protocols to handle the potential management nightmare that will potentially stem from the deployment of the IoT in the coming years. Managing IoT devices and applications can be an effective factor for growing the IoT deployments [135]. For example, monitoring the M2M communication of the IoT objects is important to ensure all times connectivity for providing on demand services. The Light-weight M2M (LWM2M) [136] is a standard that is being developed by the Open Mobile Alliance to provide interface between M2M devices and M2M Servers to build an application agnostic scheme for the management of a variety of devices. It aims to provide M2M applications with remote management capabilities of machine-to-machine devices, services, and applications. The NETCONF Light protocol [137] which is an IETF effort for the management of constrained devices provides mechanisms to install, manipulate, and delete the configuration of network devices. It is capable of managing a broad range of devices from resource-constrained to resource-rich devices. The MASH [138] IoT Platform is an example of a platform that facilitates the management (monitoring, control, and configuration) of IoT assets anywhere in real-time using an IoT dashboard on smartphones. Maintaining compatibility across the IoT layers also needs to be managed to enhance connectivity speed and to ensure service delivery. In [139], the authors propose a framework for IoT management through the concept of intercepting intermediary in which they execute heavy device management tasks on the edge routers or gateways of constrained networks. The Open Mobile Alliance (OMA) Device Management working group is specifying protocols and mechanisms for the management of mobile devices and services in resource constrained environments.

F. Scalability

The scalability of the IoT refers to the ability to add new devices, services and functions for customers without negatively affecting the quality of existing services. Adding new operations and supporting new devices is not an easy task especially in the presence of diverse hardware platforms and communications protocols. The IoT applications must be designed from the ground up to enable extensible services and operations [140]. A generic IoT architecture has been presented in [141] by introducing an IoT daemon consisting of three layers: Virtual Object, Composite Virtual Object, and Service layer. Presenting these layers featured with automation, intelligence, and zero-configuration in each object guarantees scalability as well as interoperability in IoT environment. In order to deliver scalable services [142], proposed their IoT PaaS platform through virtual vertical service delivery. IoT-iCore[3] is a work under progress that aims to provide a layered framework that offers scalable mechanisms for registration, look-up and discovery of entities, as well as interoperability between objects.

G. Interoperability

End-to-end interoperability is another challenge for the IoT due to the need to handle a large number of heterogeneous things that belong to different platforms. Interoperability should be considered by both application developers and IoT device manufactures to ensure the delivery of services for all customers regardless of the specifications of the hardware platform that they use. For example, most of the smartphones nowadays support common communication technologies such as WiFi, NFC, and GSM to guarantee the interoperability in different scenarios. Also,

[3]http://www.iot-icore.eu/

programmers of the IoT should build their applications to allow for adding new functions without causing problems or losing functions while maintaining integration with different communication technologies. Consequently, interoperability is a significant criterion in designing and building IoT services to meet customers' requirements [143]. Beside variety of protocols, different interpretations of the same standard implemented by different parties presents a challenge for interoperability [144]. To avoid such ambiguities, interoperability testing between different products in a test-bed like ETSI Plugtests would be helpful. PROBE-IT[4] is a research project that aims to ensure the interoperability of validated IoT solutions that conducted interoperability tests like CoAP, 6LoWPAN, and IoT semantic interoperability.

H. Security and Privacy

Security presents a significant challenge for the IoT implementations due to the lack of common standard and architecture for the IoT security. In heterogeneous networks as in the case of the IoT, it is not easy to guarantee the security and privacy of users. The core functionality of the IoT is based on the exchange of information between billions or even trillions of Internet connection objects. One open problem in IoT security that has not been considered in the standards is the distribution of the keys amongst devices [144]. IETF's Smart Object Lifecycle Architecture for Constrained Environments (SOLACE) started some work to overcome this problem. On the other hand, privacy issues and profile access operations between IoT devices without interferences are extremely critical. Still, securing data exchanges is necessary to avoid losing or compromising privacy. The increased number of smart things around us with sensitive data necessitates a transparent and easy access control management in such a way that for example one vendor can just read the data while another is allowed to control the device. In this regard, some solutions have been proposed such as grouping embedded devices into virtual networks and only present desired devices within each virtual network. Another approach is to support access control in the application layer on a per-vendor basis [144].

Remarks: Although a lot of research has been done in the IoT, there is a need for a lot more efforts for it to mature. The increasing attention of governments and industries to this disruptive technology has led to an extensive range of research projects. Some of the challenges like the overall architecture and security have attracted a lot of attention, while others like availability, reliability, and performance still require more attention. Some research studies have been conducted in the laboratories while others are still in the simulation phase. This is natural since these latter challenges need real applications or test-beds based on the current technologies; something that has not happened at a large-scale yet.

Another nascent IoT research thrust is to estimate the network location of smart objects to realize new location and context-aware services. The current methods for location estimation are based on IP. However, Named Data Networking (NDN) is one of the candidates for naming infrastructure in the future Internet [145].

[4]http://www.probe-it.eu/

VII. Big Data Analytics, Cloud and Fog Computing in Support of the IOT

Connecting a large number of physical objects like humans, animals, plants, smart phones, PCs, etc. equipped with sensors to the Internet generates what is called "big data." Big data needs smart and efficient storage. Obviously, connected devices need mechanisms to store, process, and retrieve data. But big data is so huge such that it exceeds the capability of commonly used hardware environments and software tools to capture, manage, and process them within an acceptable slot of time. The emerging and developing technology of cloud computing is defined by the US National Institute of Standards and Technology (NIST) as an access model to an on-demand network of shared configurable computing sources such as networks, servers, warehouses, applications, and services. Cloud services allow individuals and companies to use remote third-party software and hardware components [164]. Cloud computing enables researchers and businesses to use and maintain many resources remotely, reliably and at a low cost. The IoT employs a large number of embedded devices, like sensors and actuators that generate big data which in turn requires complex computations to extract knowledge [165]. Therefore, the storage and computing resources of the cloud present the best choice for the IoT to store and process big data. In the following subsections, we discuss the relation between the IoT and big data analytics, cloud and fog computing.

A. Big Data Analytics in Support of the IoT

What makes *big data* an important asset to businesses is that it makes it possible to extract analytics and consequently knowledge, by which a business can achieve competitive advantage. There are some platforms for big data analytics like Apache Hadoop and SciDB. However, these tools are hardly strong enough for big data needs of IoT [166]. The amount of IoT data generally is too huge to be fed and processed by the available tools. In support of the IoT, these platforms should work in real-time to serve the users efficiently. For example, Facebook has used an improved version of Hadoop to analyze billions of messages per day and offer real-time statistics of user actions [167]. In terms of resources, besides the powerful servers in data centers a lot of smart devices around us offer computing capabilities that can be used to perform parallel IoT data analytic tasks [168].

Instead of providing application specific analytics, IoT needs a common big data analytic platform which can be delivered as a service to IoT applications. Such analytic service should not impose a considerable overhead on the overall IoT ecosystem.

A recent research has proposed such an IoT big data analytics service known as TSaaaS using time series data analytics to perform pattern mining on a large amount of collected sensor data [169]. Their analytic service relies on the Time Series Database service and is accessible by a set of RESTful interfaces. Their evaluations show that TSaaaS can perform pattern searches quicker than the existing systems. They also reported that 0.4% of the original data volume was needed as the overhead space for index storage of the service provider.

One viable solution for IoT big data is to keep track of just the interesting data only. Existing approaches can help in this

field like principle component analysis (PCA), pattern reduction, dimensionality reduction, feature selection, and distributed computing methods [166].

A use-case that illustrates the use of traffic analytics in the context of IoT is presented in Section IX-B.

B. Cloud Computing for the IoT

Cloud computing (CC) offers a new management mechanism for big data that enables the processing of data and the extraction of valuable knowledge from it.

Employing CC for the IoT is not an easy task due to the following challenges:

- *Synchronization*: Synchronization between different cloud vendors presents a challenge to provide real-time services since services are built on top of various cloud platforms.
- *Standardization*: Standardizing CC also presents a significant challenge for IoT cloud-based services due having to interoperate with the various vendors.
- *Balancing*: Making a balance between general cloud service environments and IoT requirements presents another challenge due to the differences in infrastructure.
- *Reliability*: Security of IoT cloud-based services presents another challenge due to the differences in the security mechanisms between the IoT devices and the cloud platforms.
- *Management*: Managing CC and IoT systems is also a challenging factor due to the fact that both have different resources and components.
- *Enhancement*: Validating IoT cloud-based services is necessary to ensure providing good services that meet the customers' expectations.

IoT can utilize numerous cloud platforms with different capabilities and strengths such as ThingWorx, OpenIoT, Google Cloud, Amazon, GENI, etc. For example, Xively (formerly known as Cosm and Pachube) represents one of the first IoT application hosting service providers allowing sensor data to be available on the web. Xively aims to connect devices to applications securely in real-time. Xively provides a Platform as a Service (PaaS) solution for the IoT application developers and service providers. It is able to integrate devices with the platform by ready libraries (such as ARM mbed, Electric Imp and iOS/OSX) and facilitate communication via HTTP(S), Sockets/Websocket, or MQTT [170]. It could also integrate with other platforms using Java, JS, Python, and Ruby libraries. The automated parking lot presented in [171], is a sample of using Xively to implement IoT applications.

Some of the features that made Xively one of the preferred cloud-based service providers for IoT service offerings are [172]:

- Open source, free and easy to use as it exposes accessible Application Programming Interfaces (APIs).
- Interoperability with many protocols, environments and its ability to manage real-time sensors and distribute data in numerous formats such as JSON, XML and CSV.
- Enables users to visualize their data graphically in real-time using a website to monitor activities based on data sensors. Also, it enables users to control sensors remotely by modifying scripts to receive an alert.

TABLE VIII
IOT CLOUD PLATFORMS AND THEIR CHARACTERISTICS

Platform	Gateway	Provision	Assurance	Billing	Application Protocol			
					REST	CoAP	XMPP	MQTT
Arkessa	-	+	+	-	+	-	-	+
Axeda	+	+	+	+	+	-	-	-
Etherios	+	+	+	-	+	-	-	-
LittleBits	-	-	-	-	+	-	-	-
NanoService	+	+	+	-	+	+	-	-
Nimbits	-	-	-	-	+	-	+	-
Ninja Blocks	+	-	-	-	+	-	-	-
OnePlatform	+	+	+	-	+	+	+	-
RealTime.io	+	+	-	-	+	-	-	-
SensorCloud	+	+	-	-	+	-	-	-
SmartThings	+	+	-	-	+	-	-	-
TempoDB	-	-	-	-	+	-	-	-
Thingworx	-	+	+	-	+	-	-	+
Xively	+	+	+	+	+	-	-	+

- Supported by many Original Equipment Manufacturers (OEM) like Arexx, Nanode, OpenGear, Arduino, and mBed.

As another example, Nimbits is an open source Platform as a Service (PaaS) that connects smart embedded devices to the cloud [173]. It also performs data analytics on the cloud, generates alerts, and connects with social networks and spreadsheets. Moreover, it connects to websites and can store, share and retrieve sensors' data in various formats including numeric, text based, GPS, JSON or XML. To exchange data or messages, XMPP is a built-in service in Nimbits. The core of Nimbits is a server that provides REST web services for logging and retrieval of raw and processed data.

Table VIII summarizes some characteristics of several publicly available Cloud platforms for IoT (in the table, "+" stands for support and "−" stands for lack of support) [174]. The evaluation metrics include: supporting gateway devices to bridging the short range network and wide area network, supporting discovery, delivery, configuration and activation of applications and services, providing proactive and reactive assurance of platform, support of accounting and billing of applications and services, and finally support of standard application protocols. All the platforms support sensing or actuating devices, a user interface to interact with devices, and a web component to run the business logic of the application on the cloud. Also, none of them supports the DDS protocol.

C. Fog Computing in Support of the IoT

Fog Computing (a.k.a. cloudlets or edge computing) can act as a bridge between smart devices and large-scale cloud computing and storage services. Through fog computing, it is possible to extend cloud computing services to the edge devices of the network. Because of their proximity to the end-users compared to the cloud data-centers, fog computing has the potential to offer services that deliver better delay performance. It should be emphasized here that, typically there is a significant difference in scale between the fog and the cloud such that the

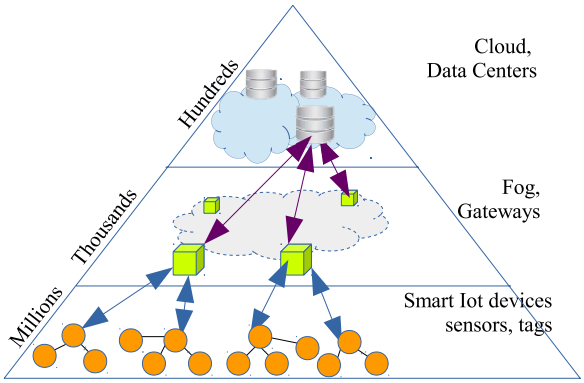

Fig. 24. The role of the cloud and fog resources in the delivery of IoT services.

cloud has massive computational, storage and communications capabilities compared to the fog [175]. Fig. 24 illustrates the roles that the cloud data-centers and the cloudlets (fog computing) play to deliver IoT services to end-users. Mobile network operators are the potential providers of fog computing since they can offer fog services as one of IaaS, PaaS, or SaaS models to the enterprise businesses by providing services at their service network or even cell tower [170].

Fog computing can serve as an optimal choice for the IoT designers for the following features:

- *Location*: Fog resources are positioned between smart objects and the cloud data-centers; thus, providing better delay performance.
- *Distribution*: Since fog computing is based on "micro" centers with limited storage, processing and communication capabilities compared to the cloud, it is possible to deploy many such "micro" centers closer to the end-users as their cost is typically a small fraction compared to cloud data-centers.
- *Scalability*: Fog allows IoT systems to be more scalable such that as the number of end-users increase, the number of deployed "micro" fog centers can increase to cope with the increasing load. Such an increase cannot be achieved by the cloud because the deployment of new data-centers is cost prohibitive.
- *Density of devices*: Fog helps to provide resilient and replicated services.
- *Mobility support*: Fog resources act as a "mobile" cloud as it is located close to the end-users.
- *Real-time*: Fog has the potential to provide better performance for real-time interactive services.
- *Standardization*: Fog resources can interoperate with various cloud provides.
- *On the fly analysis*: Fog resources can perform data aggregation to send partially processed data as opposed to raw data to the cloud data-centers for further processing.

Therefore, fog computing has the potential to increase the overall performance of IoT applications as it tries to perform part of high level services which are offered by cloud inside the local resources.

Remarks: In this section, three complementary elements to the IoT were introduced. In the field of big data analytics in support of the IoT, the conventional analytic tools that rely on offline analysis are no longer interesting. Moreover, the current trend is to increase the computing resources in support of big data analytics through IoT edge devices.

One important aspect of cloud platforms is the ability to interact with different application protocols. A cloud platform may have different customers who use specific application protocols. If the customers wish to use the services of other customers, then the limitation of the cloud which offers just a specific application protocol is a barrier to its expansion. The available cloud platforms hardly support all standard application protocols, while almost all of them support REST. However, one solution is using hybrid clouds. The RESERVOIR project [176] is such a platform that aims to provide an architecture by which cloud providers will be able to join with each other to make a great number of IT solutions. IoTCloud [177] is another project that aims to provide a scalable and high performance cloud platform for IoT application.

Through fog computing, it is proposed to use smart devices like mobile phones or home gateways [168]. However, the field of fog computing needs more attention to resolve other issues like reliability, mobility and security of analytical data on the edge devices [178]. In [179], the authors presented a fog computing model (Edge Cloud) that tries to bring information centric cloud capabilities to the edge. In this model, traditional data center hosted cloud solutions which are great for large-scale general purpose computations and storage are improved by services on the network edge. Using this architecture provides a sort of service delivery with reduced latency and bandwidth while maintaining service resiliency and localization.

VIII. The Need for Better Horizontal Integration Between Application Layer Protocols

IoT devices can be classified into two major categories; namely: resource-constrained and resource-rich devices. We define resource-rich devices as those that have the hardware and software capability to support the TCP/IP protocol suite. On devices that support the TCP/IP protocol suite, IoT applications are implemented on top of a variety of application level protocols and frameworks including REST, CoAP, MQTT, MQTT-SN, AMQP and others. On the other hand, devices that do not have the required resources to support TCP/IP cannot interoperate easily with resource-rich devices that support the TCP/IP suite. For example, microcontroller based appliances and gadgets should have the capability to interoperate with other IoT elements that are TCP/IP enabled. Beyond the interoperability issues between devices that support TCP/IP and those that do not, TCP/IP enabled devices utilize a variety of protocols leading to a myriad of interoperability issues that limit the potential applications of the IoT. This fragmentation between the protocols utilized for communication within and across resource-constrained and resource-rich devices is not foreseen to change in the near future.

This gloomy picture for interoperation between IoT devices calls for a protocol gateway that allows for better horizontal integration between these diverse technologies. Several attempts have been made in the recent literature to address this issue. Paramount amongst these attempts is Ponte [180] which was initially developed as QEST [181]. Ponte offers uniform open

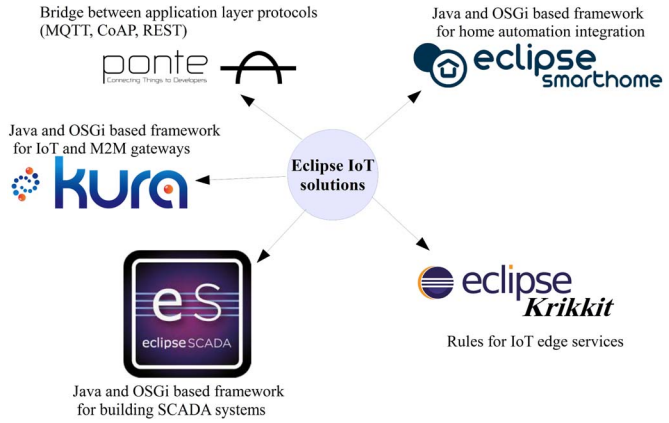

Fig. 25. Eclipse IoT projects.

APIs for the programmer to enable the automatic conversion between the various IoT application protocols such as CoAP and MQTT. Ponte is developed under the Eclipse IoT project [182] which contains other sub projects to ease the development of IoT solutions for the consumers. Other projects are Kura [159], Eclipse SCADA [183], Eclipse SmartHome [184], and Krikkit [185]. Kura as an M2M application platform is supposed to provide a Java/OSGi-based container for M2M applications running in service gateways. The most common requirements of M2M applications that Kura targets to cover are I/O access, data services, watchdog, network configuration and remote management. The SCADA's focus is to provide a way to connect different industrial devices to a common communication system. It also aims to facilitate post-process and visualizing the data. Eclipse suggests its SmartHome project as a framework for building smart home solutions. Integration of different protocols and standards in a heterogeneous environment is one of its main promising targets. Also, it is intended to bring a uniform access to underlying devices and to support different kinds of interactions between them. However, the system needs to be run on platforms that can run an OSGi stack. In order to overcome the problem of postponed stream data processing (store first, analyze later) in IoT solutions, the Krikkit project comes up with a RESTful API that allows a user or developer to acquire the data of interest in edge devices such as sensor gateways. The Krikkit architecture uses publish/subscribe model by which data acquiring rules or policies are registered on edge devices. Fig. 25 summarizes these frameworks.

Beyond the aforementioned projects, there are other academic research efforts that propose partial solutions to the problem or they have been designed for specific applications or protocols or need specific hardware. For example, the authors in [186] propose a Gateway to cover the gap between ZigBee and GPRS protocols to facilitate data transmission between wireless sensor networks and mobile communication networks. This architecture assumes the use of TCP/IP protocols. In [187], a Gateway is proposed which is specific for wireless sensor networks using TCP/IP based devices. The gateway architecture that is proposed by [188] also needs to be run on a computationally powerful system (PC).

The Light-weight M2M (LWM2M) protocol [136] which was cited before in this paper as a device management standard, aims to provide a unified way to deal with devices to manage them remotely. Although this protocol is applicable to Cellular, WiFi and WSN devices, it is limited to those devices that support IP [189].

In [190], instead of gateways, authors propose a solution to integrate smart resource-constrained objects into the Internet using virtual networks. This work can provide an end-to-end communication between devices, but scalability and binding to specific protocols are main challenges.

Authors in [191] present a communication model to support multiple protocols in a medical IoT application. Their purpose is to prevent conflict between the medical wireless transmission systems and increase the throughput of those devices in hospitals and medical environments. They used the Software Defined Radio (SDR) technology as part of their platform to sense and transform the wireless signals in the frequency spectrum. A demo is also presented in [192] in which the SDR technology is used to build a communications infrastructure for IoT applications.

An approach based on software-defined networking is proposed for IoT tasks in [193]. In their research, the authors developed a middleware with a layered IoT SDN controller to manage dynamic and heterogeneous multi-network environments.

From the anecdotal data that we collected so far about the diverse needs of IoT applications and the capabilities of the underlying hardware, it is evident to us that the strategy used by Ponte to bridge the gap between the different IoT protocols is not sufficient and a more intelligent solution is needed. To be specific, while Ponte has the capability to perform any-to-any automatic protocol conversion this conversion comes at a price as the underlying packet communication tends to be more verbose in order for it to be application agnostic. Furthermore, Ponte as many other protocol gateways that have been presented in the literature assumes the underlying devices to be TCP/IP enabled.

While this "one size fits all" approach shields programmers from having to write multiple instances of the same application to support different protocols, the underlying wire-protocol cannot be controlled by the programmer and consequently leading to performance issues and inefficiencies. Yet more importantly, resource-constrained devices are treated as second-class citizens and not considered at all in this solution.

Therefore, we are motivated by the following three main observations to content for the need of a new intelligent IoT gateway:

- Programmers should always be in control and they should have the flexibility to control the wire protocol. IoT devices can be resource-constrained and using application agnostic messaging leads to unnecessary packet exchanges. An intelligent gateway should allow for programmers to control the wire protocol traffic as needed to optimize the performance based on the specific needs of the given application.
- Resource-constrained devices should not be treated as second-class citizens. An intelligent gateway should allow for true interoperability between resource-rich and resource-constrained devices.
- Can the introduction of a protocol gateway into the IoT provide a new opportunity? An intelligent gateway should be opportunistic to create new opportunities out of the gloomy picture caused by the market fragmentation between IoT protocols.

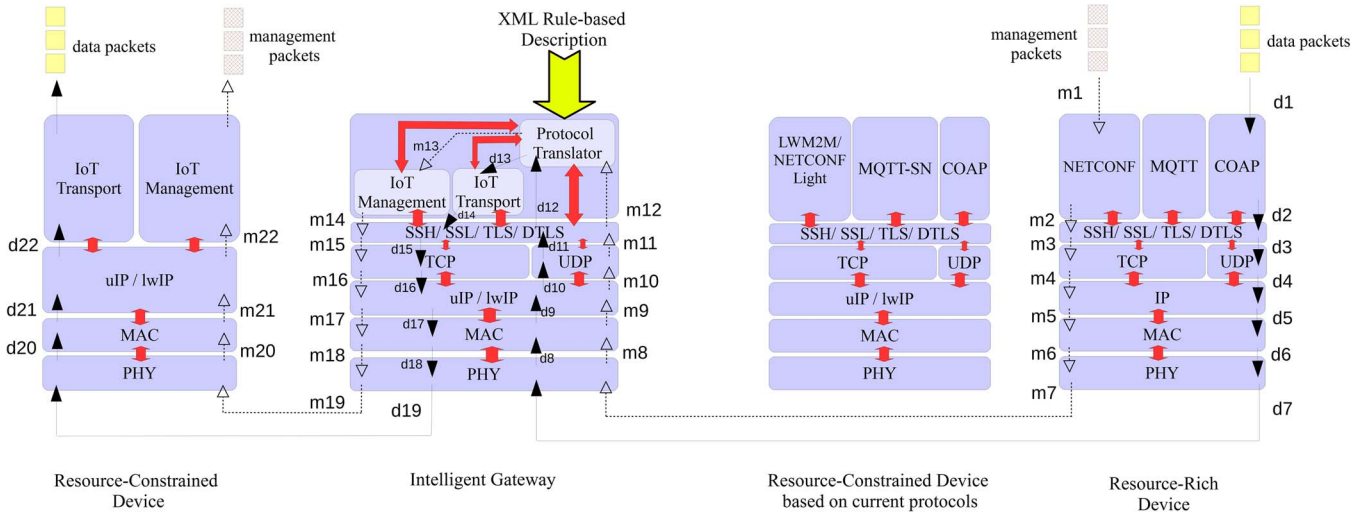

Fig. 26. The Architecture of a rule-base Intelligent IoT gateway.

Based on the aforementioned observations, we believe that there is a need for an intelligent IoT gateway that offers "smart" services that is deeply re-programmable through a rule-based language written by the programmer. It should be emphasized here that our proposal for deeper re-programmability of the IoT gateway through a rule-based language does not conflict with current interoperability and management standardization efforts. Actually, our proposal complements the interoperability effort in IEEE 1905.1 and the management efforts in LWM2M and NETCONF Light.

The concept of the proposed gateway is demonstrated in Fig. 26 [194]. We should emphasize here that the figure illustrates the protocol stack that needs to be installed on resource-constrained devices based on the current technologies vis-à-vis resource-constrained devices that utilize the intelligent gateway. The figure also details the flow sequence of data packets (d1. . .d12, then through the rule-based data-flow logic of the gateway, and finally d13. . .d22) and the flow sequence of management packets (m1. . .m12, then through the rule-based management logic of the gateway, and finally m13. . .m22). The logic of the rules is applied to the received data and management packets (i.e., d12 and m12 in Fig. 26) to generate corresponding data and management packets (i.e., d13 and m13 in Fig. 26). The rules that pertain to autonomic management and data aggregation services can result in the origination and transmission of new data and management packets by the gateway itself. It should be evident from the figure that the intelligent gateway will result in a lighter protocol stack that relies on uIP/lwIP only (no need for TCP/UDP, DTLS, TLS or other security protocols on the resource-constrained device). The "IoT transport" and "IoT Management" are two light weight protocols to encapsulate and decapsulate the data and management packets, respectively. Security services can also be delegated to the gateway so that confidentiality, authenticity, and integrity traffic can be exchanged with the gateway and not directly with resource-constrained devices. Supporting a rule-based language will shield programmers from having to write multiple instances of the same application to support the different wire protocols. Albeit the programmer will have to write rules in a high-level language to describe the transformations that the gateway should perform to translate the wire protocol as needed. The benefit of this approach stems from the ability of the programmer to perform wire protocol optimizations as needed at the cost of having to write high-level rules to describe the required transformations. With the use of standard transformation templates, the wire protocol messaging will be less efficient but in this case programmers do not have to specify rules that are specific to their application. In this later case, the performance of the proposed system will be similar to that of Ponte.

The introduction of a gateway entity within the context of IoT will also instigate the opportunity to utilize the gateway and its deep re-programmability for localized autonomic management of the IoT elements without human intervention. In real deployment scenarios, IoT nodes can be deployed by the thousands or even millions in support of a single application. Thus, having self-management Fault, Configuration, Accounting, Performance and Security (FCAPS) capabilities is a must.

Most IoT applications are low-rate but the large number of IoT devices participating on a single application will result in having the network elements to deal with "mice" flows. A deeply re-programmable gateway can also offer an opportunity to perform data and flow aggregation to limit the number of flows that the network elements have to handle. Thus, resulting in big performance gains and limiting the number of entries in the flow tables of the transport network elements.

We believe that deep re-programmability of the IoT gateway through a rule-based language can put the gateway in a unique position to offer "smart" autonomic management, data/flow aggregation, and protocol adaptation services. The presence of multiple gateways within the IoT can also offer unique benefits to balance the potentially huge IoT load amongst the available gateways.

In order to have an efficient and best-fit solution for protocol conversion, we believe that there is a need for a protocol-friendly mechanism inside the Protocol Translator that can increase the conversion speed. The key point of this mechanism is a name-value index table of data which is carried in the optional headers of the different application protocols. When a packet

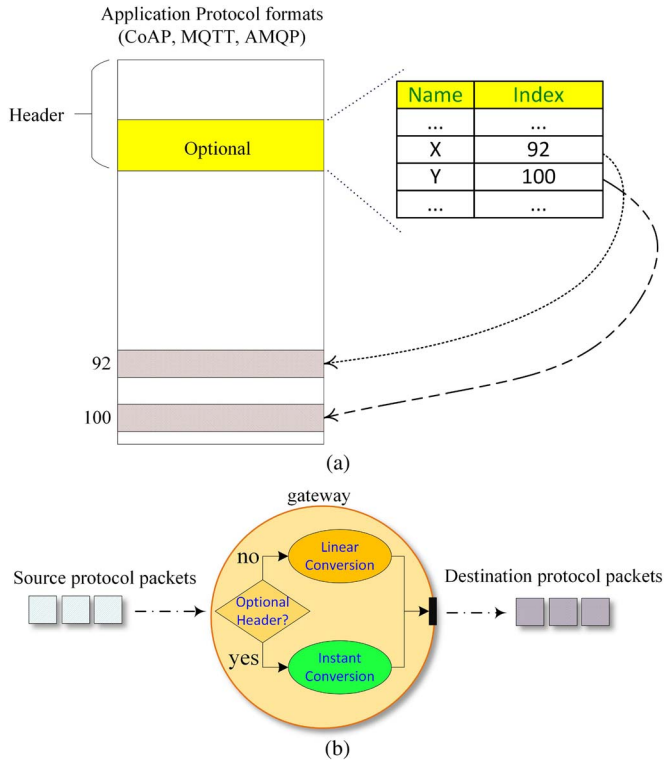

Fig. 27. The mechanism of protocol translator; (a) application protocol headers. (b) Protocol conversion.

arrives at the gateway, the Protocol Translator examines the optional headers. If it finds an acceptable index table there, it then grabs the data instantly from the payload and forms a packet targeting the destination protocol. Fig. 27 demonstrates (a) the optional header of the application protocol, the index table and (b) the conversion mechanism inside the gateway. Since the index table is stored as an optional header, application protocols may not use the index tables. In such cases, the conversion is done in the conventional form and consequently it takes longer time.

In the scenario where a packet consisting of n name-value pairs needs to be converted from a source protocol to a desired protocol in the conventional format, and since data is stored in a linear structure inside the payload of each packet, we need O(n/2) operations to find a data item in the payload before inserting it into the desired protocol. So having n name-value pairs, $O(n^2)$ operations are needed to convert all data inside a packet. This analysis is also true for XMPP in which data are stored in XML tags. Lookup and insertion in XML takes a linear time. Therefore, conversion from/to XMPP takes $O(n^2)$ as well. On the other hand, if the application protocols utilize the index table described above, the conversion time will be reduced to O(n), since the position of each name-value item is readily available in the index table.

IX. Application and Service Use-Cases

In this section, we discuss three application use-cases to point out to the readers how the main protocols discussed in the previous sections fit together to deliver desired IoT application functionality. Then, we also provide two use cases of service analytic capabilities that might be relevant to various IoT applications.

This section will focus on the overall architecture of the desired IoT applications and will not provide detailed code snippets. Readers can find relevant IoT sample source code in [69] including the following projects:

- WSN, which is a simulation of a wireless sensor network in which a sink node has to receive packets from motes to build a local table that identifies the closest anchor node to every non-anchor node in the network. The underlying standards used in this project are IPv6 and RPL.
- Service Discovery, which is an implementation and simulation of a service discovery protocol for Wireless Sensor Networks using multicast DNS (mDNS).
- Cloud Computing, in which several motes send their values (temperature readings) to the Nimbits cloud (http://cloud.nimbits.com) using HTTP REST method.

We made the source code of these projects publically available on GitHub for the readers that prefer to go through practical examples. These examples have been implemented with Contiki/cooja to make them accessible in the classroom by teachers and students. The examples can also be run on actual motes that support Contiki. The guide to use these examples is available at the wiki page of the project.

A. Application Use-Cases

The three applications that we will focus on in this section are a nursing home patient monitoring system, a system for the monitoring and mitigation of eating disorders, and an indoor navigation system for the blind and visually impaired people. The following paragraphs detail the overall architecture of these applications.

1) Nursing Home Patient Monitoring System: In this application use-case, we are interested in collecting the patients' vital sign measurements and delivering it to multiple nursing stations. We are also interested in deploying a light sensor and a door sensor to monitor the activity level of the patients and potentially identify the ones suffering from depression assuming that the patients have private rooms.

To implement this functionality quickly, an application developer can choose the relevant SmartThings or BITalino sensors that utilize ZigBee or Z-wave for communication to collect the senor measurements on the SmartThings platforms and utilize their APIs to build an application the pulls the collected data to the nursing stations.

While the SmartThings approach described above can be quickly implemented, a custom approach that utilizes Phidgets USB sensors in conjunction with a microcontroller or processor based Single Board Computer (SBC) can provide a better option to integrate hardware and software components from different providers. These sensor nodes can utilize WiFi or IEEE 802.15.4 to communicate their measurements. In this scenario, an application developer might first download and install an open source MQTT broker like Mosquitto. Then, an open source implementation of the MQTT protocol like Eclipse Paho might be used to implement a client that runs on the SBC associated with the Phidgets USB sensors collecting the vital signs, light, and door sensor data. The MQTT clients publish the sensor data to the MQTT broker. In turn, the MQTT servers connected to the

nursing stations subscribe to the MQTT broker to fetch messages of interest. If inter-sensor collaboration is needed, a routing protocol like RPL can be utilized between the sensors to enable the multi-hop delivery of data between sensors.

In order to allow doctors to access the collected data remotely, a mobile application can be developed to connect to the MQTT broker to subscribe to messages that have the topics of interest. The broker can be publically exposed on the Internet behind a firewall through an LTE-A connection using Cisco's 819 M2M Gateway.

2) Monitoring and Mitigation of Eating Disorders: Now, let us assume that we want to extend the application to allow patients with essential tremors or Parkinson disease to eat without spilling food. In this scenario, a glove can be equipped with tiny vibrating MEMS motors to counteract the hand movement instability measured by the accelerometers. In this application, the accelerometer sensors and vibrating motors have to communicate with the minimum delay possible to deliver the required functionality. Therefore, the DDS protocol would be the right choice for this scenario to allow for minimum direct communication between the accelerometers and the vibrating MEMS motors without the broker's involvement. In order to integrate this functionality with the nursing stations, a gateway needs to be deployed to translate the DDS messages to MQTT to allow for such integration.

Expanding the capabilities of MQTT based solutions can be easily done as new sensors and mobile/fixed apps can be developed to publish and/or subscribe to messages of interest through the broker. A single broker deployment can suffer from failures or performance bottlenecks in case of a large number of sensor and client app connections. In such cases, a solution that involves multiple brokers is needed and the topology and client assignment is such cases become interesting problems.

3) In-Door Navigation System for the Blind and Visually Impaired People: Let's assume that we want to extend the application further to utilize a constellation of decaWave or Nanotron transceivers to provide Real-time Locating Services (RTLS) to the users. In this case, a user held device can utilize mDNS to connect to a local server, obtain an authentication token to access the RTLS services. The RTLS nodes themselves might utilize DDS for the timely exchange of data packets. The RTLS nodes can then relay their collected data to the local RTLS server in order for it to estimate the current location of the user. The local RTLS server can overlay the location on a floorplan obtained from an Internet connected server to provide tactile navigation information to the users allowing them to avoid obstacles and other physical movement constrains reported earlier by other users of the system.

Fig. 28 provides a block diagram of the three application use-cases discussed in this section. The figure provides a visual illustration of how the application layer protocols discussed in the previous sections fit together to provide the overall IoT application functionality.

B. Service Analytic Use-Cases

Beyond the core functionality of the applications discussed above, application developers might be also interested in collecting service analytics and proactive management of the different entities utilized to deliver the application's functionality. In the following paragraphs we provide two use cases of such service analytic capabilities that might be on interest in typical IoT deployments.

1) Efficient Estimation of the Number of Unique IP Addresses Using a Given Service: Note that IoT services can have millions of users and knowing the number of unique users over a period of time is of interest. Also, storing the unique IP addresses in a relational database can be expensive. Instead, the service can hash the IP address of each packet in its local Invertible Bloom Filter (IBF) [195]. To illustrate the potential benefits of this approach, we conducted an experiment in which we assumed that in a period of time, a stream of N (N = 1000, 2000, and 3000) packets arrive to the service. In Fig. 29 we illustrate the tradeoff between the size of the IBF and the achieved accuracy. The baseline of comparison is a relational table with one thousand records. If we want to store these IPs in a traditional relational table, we need 4∗N bytes of memory, while in the proposed approach, we need much less memory. Table IX shows the memory portion used by the IBF (in the case that we need accuracy at least 0.95) for the different input sizes. In the worst case that we use 2250 bytes to cover N = 1000 IPs, our memory usage is 56% (2250/4000) of the baseline approach. While in realistic scenarios the input size will be much bigger, this scenario illustrates the potential benefit of employing the IBF in the context of the IoT in support of collecting service analytics.

2) Tracking the Frequency of Service Usage by a Given IP: Tracking the usage frequency of IoT services can greatly help in the network and application management tasks. A service can efficiently store the set of IP addresses and their associated access frequencies in the form of key-value pair in its local IBF. To illustrate the benefit of using the IBF in this context, we conducted an experiment in which we simulated a stream of service requests generated by 250 different IP addresses. So the IP addresses and their frequencies are maintained in the IBF. The number of service requests in this experiment reaches to 10 000. Fig. 30 shows the comparison of the accuracy of frequency statistics and the cost of memory usage for the IBF versus a relational data structure. It is interesting that with an IBF of size 50 bytes we are able to have a complete list of IPs and their frequencies. In a relational structure, we need 6 bytes for each IP (4 byte) and frequency (2 byte). Therefore to cover the statistics of all 250 IPs by a conventional structure we need 1500 bytes. Again, while in realistic scenarios the input size will be much bigger; this scenario illustrates the potential benefit of employing the IBF in the context of the IoT in support of collecting service analytics.

X. Concluding Remarks

A. Lessons Learned

In this paper, we reviewed IoT from different angles and here we summarize the lessons learned by this review. First, from the market opportunities perspectives, investment on this new technology is rational for organizations seeking market competitiveness. From the architecture point of view, the layered structure of IoT systems is adopted well by IoT frameworks and

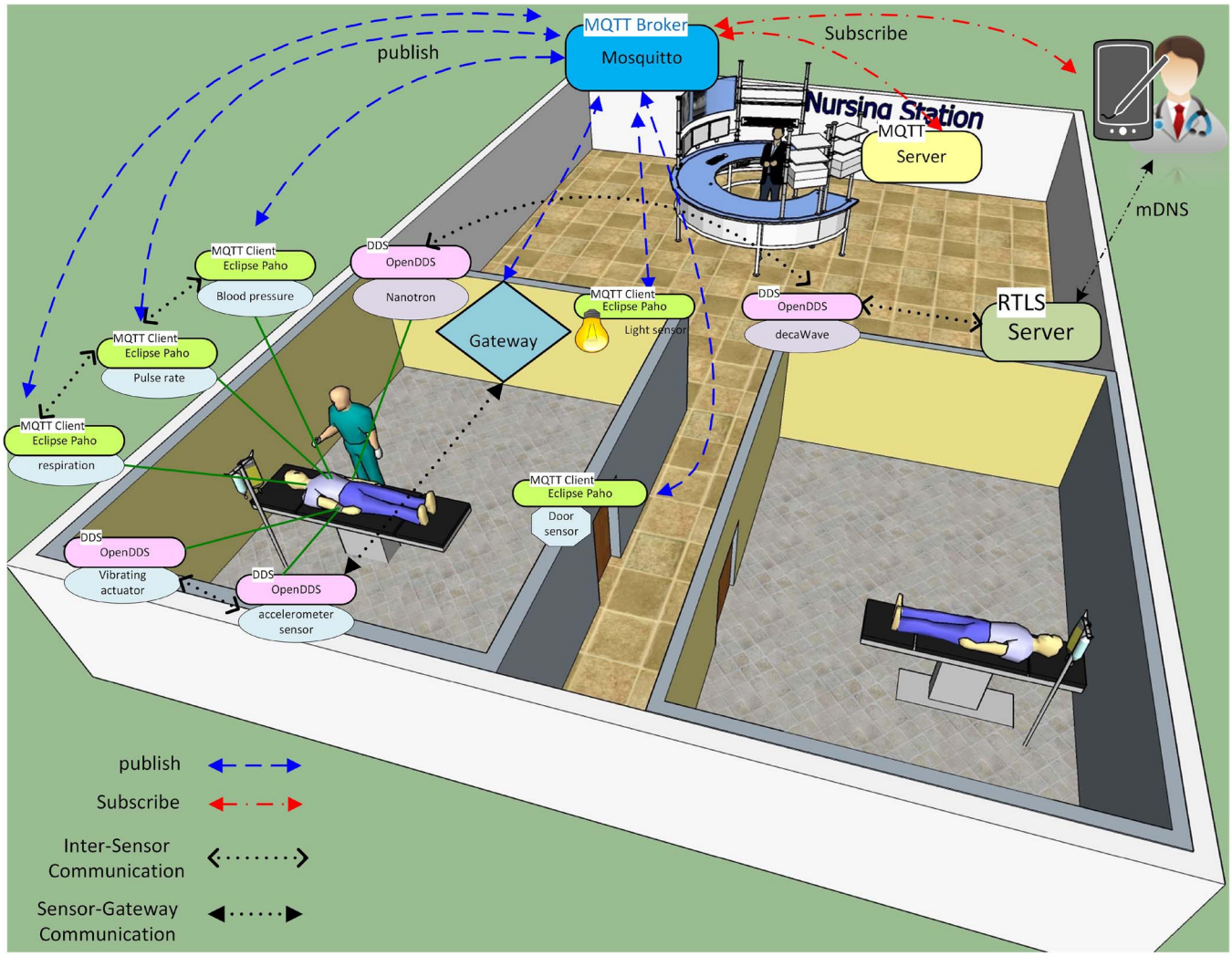

Fig. 28. The integration of IoT protocols in nursing scenario.

TABLE IX
THE IBF MEMORY RATE FOR DIFFERENT INPUT SIZES

N	IBF size for accuracy>0.95 (byte)	Relational table (byte)	Memory portion
1000	550	4000	0.1375
2000	1500	8000	0.1875
3000	2250	12000	0.1875

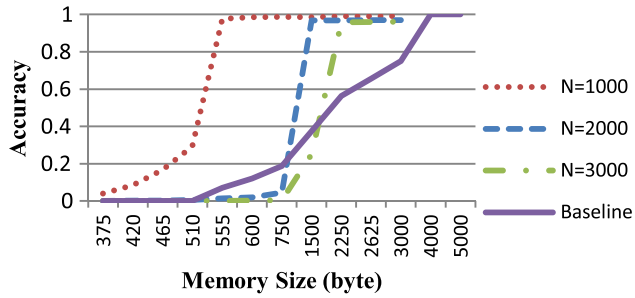

Fig. 29. Tradeoff between accuracy and the IBF size.

research attempts [2], [3], [15]–[20]. However, the number of layers and their scopes are defined differently depending on the underlying infrastructures and technologies. As scalability and interoperability have a great importance in IoT applications, augmenting the architecture with better abstractions can ease these issues. The five-layer architecture [3], [17], [18] present such a model.

Then, we introduced the different IoT elements and their related technologies and tools that are needed to realize an IoT solution. Identification and sensing are the elementary components of a system. Low-weight efficient communication between sensing devices and interoperability between different communication mechanisms are critical problems of IoT. Communication technologies like ZigBee, Bluetooth LE, NFC, RFID, Z-Wave, and LTE-A are among the most attractive technologies to use in IoT and M2M environments.

We dived more into the IoT protocols and standards by reviewing the different protocols and standards in the different layers of an IoT environment. We addressed the main functionality and role of these protocols so the reader can learn the fundamentals of these protocols without having to go through the thousands of pages of specification documents for the different protocols. Optimization of the current protocols is something that requires further development.

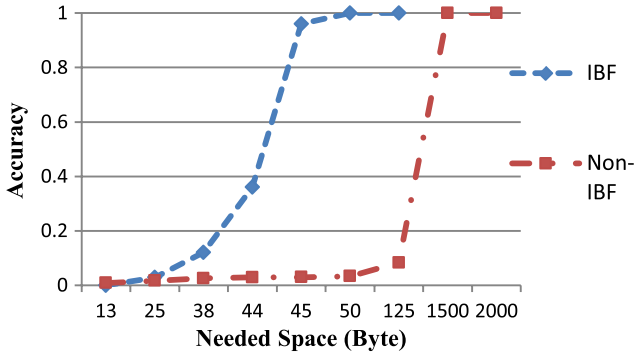

Fig. 30. The IBF versus Relational structure in terms of needed space.

We also reviewed a large number of recent studies to find out the main challenges and open issues in the IoT area. Security and privacy are the top priority for IoT applications followed by performance, reliability and management. We then investigated the consequence of IoT which is Big Data and the need for a new generation of data analytics algorithms and tools that are suitable for IoT big data. We found out that real-time data analytics techniques that are able to shrink the input size would be promising. We also reviewed the role of cloud and fog computing in the IoT ecosystem. At the cloud level, we need platforms to support IoT big data, IoT analytics and availability.

In Section IX, we presented three detailed use-cases that illustrate how the different protocols presented in this survey fit together to deliver new smart IoT services that deliver new functionality to the users while bridging the gap between the divergent IoT protocols and performing opportunistic traffic analytics.

B. Conclusion

The emerging idea of the Internet of Things (IoT) is rapidly finding its path throughout our modern life, aiming to improve the quality of life by connecting many smart devices, technologies, and applications. Overall, the IoT would allow for the automation of everything around us. This paper presented an overview of the premise of this concept, its enabling technologies, protocols, applications, and the recent research addressing different aspects of the IoT. This, in turn, should provide a good foundation for researchers and practitioners who are interested to gain an insight into the IoT technologies and protocols to understand the overall architecture and role of the different components and protocols that constitute the IoT. Further, some of the challenges and issues that pertain to the design and deployment of IoT implementations have been presented. Moreover, the interplay between the IoT, big data analytics, cloud and fog computing has been discussed.

We finally presented the need for new "smart" autonomic management, data aggregation, and protocol adaptation services to achieve better horizontal integration among IoT service. Finally, detailed application-use cases were presented to illustrate typical protocol integration scenarios to deliver desired IoT services.

REFERENCES

[1] D. Evans, "The Internet of things: How the next evolution of the Internet is changing everything," CISCO, San Jose, CA, USA, White Paper, 2011.
[2] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.
[3] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future Internet: The Internet of Things architecture, possible applications and key challenges," in *Proc. 10th Int. Conf. FIT*, 2012, pp. 257–260.
[4] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, Sep. 2013.
[5] P. Lopez, D. Fernandez, A. J. Jara, and A. F. Skarmeta, "Survey of Internet of Things technologies for clinical environments," in *Proc. 27th Int. Conf. WAINA*, 2013, pp. 1349–1354.
[6] D. Yang, F. Liu, and Y. Liang, "A survey of the Internet of Things," in *Proc. 1st ICEBI*, 2010, pp. 358–366.
[7] A. Gluhak *et al.*, "A survey on facilities for experimental Internet of Things research," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 58–67, Nov. 2011.
[8] Z. Sheng *et al.*, "A survey on the IETF protocol suite for the Internet of Things: Standards, challenges, and opportunities," *IEEE Wireless Commun.*, vol. 20, no. 6, pp. 91–98, Dec. 2013.
[9] J. Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east," *IDC iView: IDC Anal. Future*, vol. 2007, pp. 1–16, Dec. 2012.
[10] S. Taylor, "The next generation of the Internet revolutionizing the way we work, live, play, and learn," CISCO, San Francisco, CA, USA, CISCO Point of View, 2013.
[11] J. Manyika *et al.*, *Disruptive Technologies: Advances that Will Transform Life, Business, and the Global Economy*. San Francisco, CA, USA: McKinsey Global Instit., 2013.
[12] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, and J. Wang, "A first look at cellular machine-to-machine traffic: Large scale measurement and characterization," in *Proc. ACM SIGMETRICS Perform. Eval. Rev.*, 2012, pp. 65–76.
[13] D. Floyer, "Defining and sizing the industrial Internet," Wikibon, Marlborough, MA, USA, 2013.
[14] "Commercial building automation systems," Navigant Consulting Res., Boulder, CO, USA, 2013.
[15] S. Krco, B. Pokric, and F. Carrez, "Designing IoT architecture(s): A European perspective," in *Proc. IEEE WF-IoT*, 2014, pp. 79–84.
[16] EU FP7 Internet of Things Architecture Project, Sep. 18, 2014. [Online]. Available: http://www.iot-a.eu/public
[17] Z. Yang *et al.*, "Study and application on the architecture and key technologies for IOT," in *Proc. ICMT*, 2011, pp. 747–751.
[18] M. Wu, T. J. Lu, F. Y. Ling, J. Sun, and H. Y. Du, "Research on the architecture of Internet of Things," in *Proc. 3rd ICACTE*, 2010, pp. V5-484–V5-487.
[19] L. Tan and N. Wang, "Future Internet: The Internet of Things," in *Proc. 3rd ICACTE*, 2010, pp. V5-376–V5-380.
[20] M. A. Chaqfeh and N. Mohamed, "Challenges in middleware solutions for the Internet of Things," in *Proc. Int. Conf. CTS*, 2012, pp. 21–26.
[21] N. Koshizuka and K. Sakamura, "Ubiquitous ID: Standards for Ubiquitous computing and the Internet of Things," *IEEE Pervasive Comput.*, vol. 9, no. 4, pp. 98–101, Oct.–Dec. 2010.
[22] N. Kushalnagar, G. Montenegro, and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, assumptions, problem statement, and goals," Internet Eng. Task Force (IETF), Fremont, CA, USA, RFC4919, vol. 10, Aug. 2007.
[23] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of IPv6 packets over IEEE 802.15. 4 networks," Internet Eng. Task Force (IETF), Fremont, CA, USA, Internet Proposed Std. RFC 4944, 2007.
[24] K. Pilkington, "Revolv teams up with Home Depot to keep your house connected," Centre National d'Etudes des Telecommunications (CNET), San Francisco, CA, USA, CNET—News, 2014. [Online]. Available: http://Ces.Cnet.Com/8301-35306_1-57616921/Revolv-Teams-Up-with-Home-Depot-to-Keep-Your-House-Connected/
[25] "SmartThings | Home automation, home security, and peace of mind," SmartThings, Palo Alto, CA, USA, Sep. 2014. [Online]. Available: http://www.smartthings.com
[26] U. Rushden, *Belkin Brings Your Home to Your Fingertips With WeMo Home Automation System*. Los Angeles, CA, USA: Press Room Belkin, 2012.
[27] R. Want, "An introduction to RFID technology," *IEEE Pervasive Comput.*, vol. 5, no. 1, pp. 25–33, Jan.–Mar. 2006.
[28] R. Want, "Near field communication," *IEEE Pervasive Comput.*, vol. 10, no. 3, pp. 4–7, Jul./Sep. 2011.
[29] R. S. Kshetrimayum, "An introduction to UWB communication systems," *IEEE Potentials*, vol. 28, no. 2, pp. 9–13, Mar./Apr. 2009.

[30] E. Ferro and F. Potorti, "Bluetooth and Wi-Fi wireless protocols: A survey and a comparison," *IEEE Wireless Commun.*, vol. 12, no. 1, pp. 12–26, Feb. 2005.
[31] P. McDermott-Wells, "What is Bluetooth?" *IEEE Potentials*, vol. 23, no. 5, pp. 33–35, Jan. 2005.
[32] "Press releases detail: Bluetooth technology website," Bluetooth Technol. Website, Kirkland, WA, USA, Sep. 2014. [Online]. Available: http://www.bluetooth.com/Pages/Press-Releases-Detail.aspx?ItemID=197
[33] *IEEE Standard for Local and Metropolitan Area Networks—Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*, IEEE Std. 802. 15. 4-2011, 2011.
[34] G. V. Crosby and F. Vafa, "Wireless sensor networks and LTE-A network convergence," in *Proc. IEEE 38th Conf. LCN*, 2013, pp. 731–734.
[35] A. Ghosh, R. Ratasuk, B. Mondal, N. Mangalvedhe, and T. Thomas, "LTE-Advanced: Next-generation wireless broadband technology [Invited Paper]," *IEEE Wireless Commun.*, vol. 17, no. 3, pp. 10–22, Jun. 2010.
[36] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki—A lightweight and flexible operating system for tiny networked sensors," in *Proc. 29th Annu. IEEE Int. Conf. Local Comput. Netw.*, 2004, pp. 455–462.
[37] P. Levis *et al.*, "TinyOS: An operating system for sensor networks," in *Ambient Intelligence*. New York, NY, USA: Springer-Verlag, 2005, pp. 115–148.
[38] Q. Cao, T. Abdelzaher, J. Stankovic, and T. He, "The LiteOS operating system: Towards Unix-like abstractions for wireless sensor networks," in *Proc. Int. Conf. IPSN*, 2008, pp. 233–244.
[39] E. Baccelli, O. Hahm, M. Günes, M. Wählisch, and T. C. Schmidt, "RIOT OS: Towards an OS for the Internet of Things," in *Proc. IEEE Conf. INFOCOM WKSHPS*, 2013, pp. 79–80.
[40] Open Auto Alliance, Oct. 20, 2014. Available: http://www.openautoalliance.net/
[41] X. Xiaojiang, W. Jianli, and L. Mingdong, "Services and key technologies of the Internet of Things," ZTE Commun., Shenzhen, China, vol. 2, p. 011, 2010.
[42] M. Gigli and S. Koo, "Internet of Things: Services and applications categorization," *Adv. Internet Things*, vol. 1, no. 2, pp. 27–31, Jul. 2011.
[43] D. J. Cook, A. S. Crandall, B. L. Thomas, and N. C. Krishnan, "CASAS: A smart home in a box," *Computer*, vol. 46, no. 7, pp. 62–69, Jul. 2013.
[44] N. Komninos, E. Philippou, and A. Pitsillides, "Survey in smart grid and smart home security: Issues, challenges and countermeasures," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1933–1954, 4th Quart. 2014.
[45] E. Finch, "Is IP everywhere the way ahead for building automation?" *Facilities*, vol. 19, no. 11/12, pp. 396–403, 2001.
[46] C. Talcott, "Cyber–physical systems and events," in *Software-Intensive Systems and New Computing Paradigms*, M. Wirsing, J. Banatre, M. Hölzl, and A. Rauschmayer, Eds. New York, NY, USA: Springer Sci. Business Media, 2008, pp. 101–115.
[47] L. Yongfu, S. Dihua, L. Weining, and Z. Xuebo, "A service-oriented architecture for the transportation cyber–physical systems," in *Proc. 31st CCC*, 2012, pp. 7674–7678.
[48] L. Ying and Z. Lingshu, "Novel design of intelligent Internet-of-vehicles management system based on cloud-computing and Internet-of-Things," in *Proc. Int. Conf. EMEIT*, 2011, pp. 3190–3193.
[49] M. Gerla, L. Eun-Kyu, G. Pau, and L. Uichin, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds," in *Proc. IEEE WF-IoT*, 2014, pp. 241–246.
[50] A. Strange, "Toyota, Audi prepping self-driving cars," PC, New York, NY, USA, PCMag.Com: New Product Rev. 2013. [Online]. Available: http://Www.Pcmag.Com/Article2/0,2817,2413841,00.Asp
[51] J. Markoff, "Google cars drive themselves, in traffic," New York Times, New York, NY, USA, 2010. [Online]. Available: http://www.Nytimes.com/2010/10/10/Science/10google.Html?Pagewanted=all&_r=0
[52] A. Del-Colle, "Volvo will test autonomous cars on Sweden's streets—Popular mechanics," Popular Mechanics, Cars, News, New York, NY, USA, 2013.
[53] I. Ungurean, N. C. Gaitan, and V. G. Gaitan, "An IoT architecture for things from industrial environment," in *Proc. 10th Int. COMM*, 2014, pp. 1–4.
[54] C. Wang, Z. Bi, and L. D. Xu, "IoT and cloud computing in automation of assembly modeling systems," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1426–1434, May 2014.
[55] "Radical-7 breakthrough measurements. Radical monitor," Masimo Corp., Irvine, CA, USA, Data Sheet Radical-7, 2013.
[56] C. Nay, "Sensors remind doctors to wash up," IBM Res., Armonk, NY, USA, 2013.
[57] K. Michaelsen, J. L. Sanders, S. M. Zimmer, and G. M. Bump, "Overcoming patient barriers to discussing physician hand hygiene: Do patients prefer electronic reminders to other methods?" *Infection Control*, vol. 34, no. 9, pp. 929–934, Sep. 2013.
[58] S. Jain *et al.*, "A low-cost custom HF RFID system for hand washing compliance monitoring," in *Proc. IEEE 8th ASICON*, 2009, pp. 975–978.
[59] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, "A survey on smart grid communication infrastructures: Motivations, requirements and challenges," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 5–20, 1st Quart. 2013.
[60] T. Gea, J. Paradells, M. Lamarca, and D. Roldan, "Smart cities as an application of Internet of things: Experiences and lessons learnt in Barcelona," in *Proc. 7th Int. Conf. IMIS Ubiquitous Comput.*, 2013, pp. 552–557.
[61] J. Jin, J. Gubbi, S. Marusic, and M. Palaniswami, "An information framework for creating a smart city through Internet of Things," *IEEE Internet Things J.*, vol. 1, no. 2, pp. 112–121, Apr. 2014.
[62] P. Barnaghi, W. Wang, C. Henson, and K. Taylor, "Semantics for the Internet of Things: Early progress and back to the future," *Proc. IJSWIS*, vol. 8, no. 1, pp. 1–21, Jan. 2012.
[63] T. Kamiya and J. Schneider, "Efficient XML Interchange (EXI) Format 1.0," World Wide Web Consortium, Cambridge, MA, USA, Recommend. REC-Exi-20110310, 2011.
[64] Z. Shelby, K. Hartke, C. Bormann, and B. Frank, "Constrained Application Protocol (CoAP).draft-ietf-core-coap-18," Internet Eng. Task Force (IETF), Fremont, CA, USA, 2013.
[65] C. Bormann, A. P. Castellani, and Z. Shelby, "CoAP: An application protocol for billions of tiny Internet nodes," *IEEE Internet Comput.*, vol. 16, no. 2, pp. 62–67, Mar./Apr. 2012.
[66] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 2000.
[67] W. Colitti, K. Steenhaut, N. De Caro, B. Buta, and V. Dobrota, "Evaluation of constrained application protocol for wireless sensor networks," in *Proc. 18th IEEE Workshop LANMAN*, 2011, pp. 1–6.
[68] C. Lerche, K. Hartke, and M. Kovatsch, "Industry adoption of the Internet of Things: A constrained application protocol survey," in *Proc. IEEE 17th Conf. ETFA*, 2012, pp. 1–6.
[69] IoT Code Recipes: RPL, mDNS and REST, Jul. 8, 2014. [Online]. Available: http://github.com/mehdimo/IoTCodeRecipes
[70] D. Locke, "MQ telemetry transport (MQTT) v3. 1 protocol specification," IBM developerWorks, Markham, ON, Canada, Tech. Lib., 2010. [Online]. Available: Http://Www.Ibm.Com/Developerworks/Webservices/Library/Ws-Mqtt/Index.Html
[71] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, "MQTT-S—A publish/subscribe protocol for wireless sensor networks," in *Proc. 3rd Int. Conf. COMSWARE*, 2008, pp. 791–798.
[72] P. Saint-Andre, "Extensible messaging and presence protocol (XMPP): Core," Internet Eng. Task Force (IETF), Fremont, CA, USA, Request for Comments: 6120, 2011.
[73] M. T. Jones, "Meet the Extensible Messaging and Presence Protocol (XMPP)," IBM developerWorks, Markham, ON, Canada, 2009.
[74] P. Waher and Y. Doi, *Efficient XML Interchange (EXI) Format*, Std. XEP-0322, 2013.
[75] "OASIS Advanced Message Queuing Protocol (AMQP) Version 1.0," Adv. Open Std. Inf. Soc. (OASIS), Burlington, MA, USA, 2012.
[76] Data distribution services specification, V1.2, Object Manage. Group (OMG), Needham, MA, USA, Apr. 2, 2015. [Online]. Available: http://www.omg.org/spec/DDS/1.2/
[77] C. Esposito, S. Russo, and D. Di Crescenzo, "Performance assessment of OMG compliant data distribution middleware," in *Proc. IEEE IPDPS*, 2008, pp. 1–8.
[78] D. Thangavel, X. Ma, A. Valera, H. Tan, and C. K. Tan, "Performance evaluation of MQTT and CoAP via a common middleware," in *Proc. IEEE 9th Int. Conf. ISSNIP*, 2014, pp. 1–6.
[79] N. De Caro, W. Colitti, K. Steenhaut, G. Mangino, and G. Reali, "Comparison of two lightweight protocols for smartphone-based sensing," in *Proc. IEEE 20th SCVT*, 2013, pp. 1–6.
[80] M. Laine and K. Säilä, "Performance evaluation of XMPP on the Web," Aalto Univ. Tech. Rep., Aalto, Finland, 2012.
[81] J. L. Fernandes, I. C. Lopes, J. J. P. C. Rodrigues, and S. Ullah, "Performance evaluation of RESTful web services and AMQP protocol," in *Proc. 5th ICUFN*, 2013, pp. 810–815.
[82] A. J. Jara, P. Martinez-Julia, and A. Skarmeta, "Light-weight multicast DNS and DNS-SD (lmDNS-SD): IPv6-based resource and service discovery for the web of things," in *Proc. 6th Int. Conf. IMIS Ubiquitous Comput.*, 2012, pp. 731–738.
[83] R. Klauck and M. Kirsche, "Chatty things—Making the Internet of Things readily usable for the masses with XMPP," in *Proc. 8th Int. Conf. CollaborateCom*, 2012, pp. 60–69.

[84] S. Cheshire and M. Krochmal, "Multicast DNS," Internet Eng. Task Force (IETF), Fremont, CA, USA, Request for Comments: 6762, 2013.

[85] M. Krochmal and S. Cheshire, "DNS-based service discovery," Internet Eng. Task Force (IETF), Fremont, CA, USA, Request for Comments: 6763, 2013.

[86] J. Vasseur *et al.*, "RPL: The IP routing protocol designed for low power and lossy networks," Internet Protocol for Smart Objects (IPSO) Alliance, San Jose, CA, USA, 2011.

[87] T. Winter *et al.*, "RPL: IPv6 routing protocol for low-power and lossy networks," Internet Eng. Task Force (IETF), Fremont, CA, USA, Request for Comments: 6550, 2012.

[88] T. Clausen, U. Herberg, and M. Philipp, "A critical evaluation of the IPv6 routing protocol for low power and lossy networks (RPL)," in *Proc. IEEE 7th Int. Conf. WiMob*, 2011, pp. 365–372.

[89] M. R. Palattella *et al.*, "Standardized protocol stack for the Internet of (important) things," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1389–1406, 3rd Quart. 2013.

[90] J. Ko *et al.*, "Connecting low-power and lossy networks to the Internet," *IEEE Commun. Mag.*, vol. 49, no. 4, pp. 96–101, Apr. 2011.

[91] J. W. Hui and D. E. Culler, "Extending IP to low-power, wireless personal area networks," *IEEE Internet Comput.*, vol. 12, no. 4, pp. 37–45, Jul./Aug. 2008.

[92] R. Frank, W. Bronzi, G. Castignani, and T. Engel, "Bluetooth low energy: An alternative technology for VANET applications," in *Proc. 11th Annu. Conf. WONS*, 2014, pp. 104–107.

[93] J. Decuir, "Introducing Bluetooth smart: Part 1: A look at both classic and new technologies," *IEEE Consum. Electron. Mag.*, vol. 3, no. 1, pp. 12–18, Jan. 2014.

[94] E. Mackensen, M. Lai, and T. M. Wendt, "Bluetooth low energy (BLE) based wireless sensors," in *IEEE Sens.*, 2012, pp. 1–4.

[95] M. Siekkinen, M. Hiienkari, J. K. Nurminen, and J. Nieminen, "How low energy is Bluetooth low energy? Comparative measurements with ZigBee/802.15.4," in *Proc. IEEE WCNCW*, 2012, pp. 232–237.

[96] E. C. Jones and C. A. Chung, *RFID and Auto-ID in Planning and Logistics: A Practical Guide for Military UID Applications*. Boca Raton, FL, USA: CRC Press, 2011.

[97] D. Minoli, *Building the Internet of Things With IPv6 and MIPv6: The Evolving World of M2M Communications*. New York, NY, USA: Wiley, 2013.

[98] J. Grasso, "The EPCglobal network: Overview of design, benefits, and security," EPCglobal Inc, Position Paper, vol. 24, 2004.

[99] M. Hasan, E. Hossain, and D. Niyato, "Random access for machine-to-machine communication in LTE-Advanced networks: Issues and approaches," *IEEE Commun. Mag.*, vol. 51, no. 6, pp. 86–93, Jun. 2013.

[100] D. Niyato, P. Wang, and D. I. Kim, "Performance modeling and analysis of heterogeneous machine type communications," *IEEE Trans. Wireless Commun.*, vol. 13, no. 5, pp. 2836–2849, May 2014.

[101] C. Gomez and J. Paradells, "Wireless home automation networks: A survey of architectures and technologies," *IEEE Commun. Mag.*, vol. 48, no. 6, pp. 92–101, Jun. 2010.

[102] N. Accettura, L. A. Grieco, G. Boggia, and P. Camarda, "Performance analysis of the RPL routing protocol," in *Proc. IEEE ICM*, 2011, pp. 767–772.

[103] E. Ancillotti, R. Bruno, and M. Conti, "RPL routing protocol in advanced metering infrastructures: An analysis of the unreliability problems," in *Proc. SustainIT*, 2012, pp. 1–10.

[104] B. Cody-Kenny *et al.*, "Performance evaluation of the 6LoWPAN protocol on MICAz and TelosB motes," in *Proc. 4th ACM Workshop Perform. Monitoring Meas. Heterogeneous Wireless Wired Netw.*, 2009, pp. 25–30.

[105] B. Enjian and Z. Xiaokui, "Performance evaluation of 6LoWPAN gateway used in actual network environment," in *Proc. ICCECT*, 2012, pp. 1036–1039.

[106] B. B. Olyaei, J. Pirskanen, O. Raeesi, A. Hazmi, and M. Valkama, "Performance comparison between slotted IEEE 802.15.4 and IEEE 802.1 lah in IoT based applications," in *Proc. IEEE 9th Int. Conf. WiMob*, 2013, pp. 332–337.

[107] J. Y. Maina, M. H. Mickle, M. R. Lovell, and L. A. Schaefer, "Application of CDMA for anti-collision and increased read efficiency of multiple RFID tags," *J. Manuf. Syst.*, vol. 26, no. 1, pp. 37–43, Jan. 2007.

[108] E. Vahedi, R. K. Ward, and I. F. Blake, "Performance analysis of RFID protocols: CDMA versus the standard EPC Gen-2," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 4, pp. 1250–1261, Oct. 2014.

[109] C. Withanage, R. Ashok, C. Yuen, and K. Otto, "A comparison of the popular home automation technologies," in *Proc. IEEE ISGT Asia*, 2014, pp. 600–605.

[110] I. Bagci, M. Pourmirza, S. Raza, U. Roedig, and T. Voigt, "Codo: Confidential data storage for wireless sensor networks," in *Proc. IEEE 9th Int. Conf. MASS*, 2012, pp. 1–6.

[111] S. Raza, S. Duquennoy, J. Höglund, U. Roedig, and T. Voigt, "Secure communication for the Internet of Things—A comparison of link-layer security and IPsec for 6LoWPAN," *Security Commun. Netw.*, vol. 7, no. 12, pp. 2654–2668, Dec. 2012.

[112] S. Raza *et al.*, "Securing communication in 6LoWPAN with compressed IPsec," in *Proc. Int. Conf. DCOSS*, 2011, pp. 1–8.

[113] M. Srivatsa and L. Liu, "Securing publish-subscribe overlay services with EventGuard," in *Proc. 12th ACM Conf. Comput. Commun. Security*, 2005, pp. 289–298.

[114] A. B. Corman, P. Schachte, and V. Teague, "QUIP: A protocol for securing content in peer-to-peer publish/subscribe overlay networks," in *Proc. 13th Australasian Conf. Comput. Sci.*, 2007, vol. 62, pp. 35–40.

[115] S. Raza, H. Shafagh, K. Hewage, R. Hummen, and T. Voigt, "Lithe: Lightweight secure CoAP for the Internet of Things," *IEEE Sens. J.*, vol. 13, no. 10, pp. 3711–3720, Oct. 2013.

[116] MQTT NIST Cyber Security Framework, Jun. 15, 2014. [Online]. Available: https://www.oasis-open.org/committees/download.php/52641/mqtt-nist-cybersecurity-v1.0-wd02.doc

[117] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: Real-time intrusion detection in the Internet of Things," *Ad Hoc Netw.*, vol. 11, no. 8, pp. 2661–2674, Nov. 2013.

[118] *IEEE Standard for a Convergent Digital Home Network for Heterogeneous Technologies*, IEEE Std. 1905. 1-2013, pp. 1–93, 2013.

[119] Y. Liu, Y. Meng, and J. Huang, "Gemini: A green deployment scheme for Internet of Things," in *Proc. 22nd WOCC*, 2013, pp. 338–343.

[120] D. Uckelmann, "Performance measurement and cost benefit analysis for RFID and Internet of Things implementations in logistics," in *Quantifying the Value of RFID and the EPCglobal Architecture Framework in Logistics*. New York, NY, USA: Springer-Verlag, 2012, pp. 71–100.

[121] J. A. Stankovic, "Research directions for the Internet of Things," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 3–9, Feb. 2014.

[122] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, "A vision of IoT: Applications, challenges, and opportunities with china perspective," *IEEE Internet Things J.*, vol. 1, no. 4, pp. 349–359, Aug. 2014.

[123] S. Ziegler *et al.*, *IoT6-Moving to an Ipv6-Based Future IoT*. New York, NY, USA: Springer-Verlag, 2013.

[124] D. Macedo, L. A. Guedes, and I. Silva, "A dependability evaluation for Internet of Things incorporating redundancy aspects," in *Proc. IEEE 11th ICNSC*, 2014, pp. 417–422.

[125] I. Silva, R. Leandro, D. Macedo, and L. A. Guedes, "A dependability evaluation tool for the Internet of Things," *Comput. Electr. Eng.*, vol. 39, no. 7, pp. 2005–2018, Oct. 2013.

[126] N. Maalel, E. Natalizio, A. Bouabdallah, P. Roux, and M. Kellil, "Reliability for emergency applications in Internet of Things," in *Proc. IEEE Int. Conf. DCOSS*, 2013, pp. 361–366.

[127] J. Kempf, J. Arkko, N. Beheshti, and K. Yedavalli, "Thoughts on reliability in the Internet of Things," in *Proc. Interconnecting Smart Objects Internet Workshop*, 2011, pp. 1–4.

[128] L. Li, Z. Jin, G. Li, L. Zheng, and Q. Wei, "Modeling and analyzing the reliability and cost of service composition in the IoT: A probabilistic approach," in *Proc. IEEE 19th ICWS*, 2012, pp. 584–591.

[129] F. Ganz, R. Li, P. Barnaghi, and H. Harai, "A resource mobility scheme for service-continuity in the Internet of Things," in *Proc. IEEE Int. Conf. GreenCom*, 2012, pp. 261–264.

[130] H. Fu, P. Lin, H. Yue, G. Huang, and C. Lee, "Group mobility management for large-scale machine-to-machine mobile networking," *IEEE Trans. Veh. Technol.*, vol. 63, no. 3, pp. 1296–1305, Mar. 2014.

[131] T. Elsaleh, A. Gluhak, and K. Moessner, "Service continuity for subscribers of the mobile real world Internet," in *Proc. IEEE ICC Workshops*, 2011, pp. 1–5.

[132] Z. Zhu, L. Zhang, and R. Wakikawa, "Supporting mobility for Internet cars," *IEEE Commun. Mag.*, vol. 49, no. 5, pp. 180–186, May 2011.

[133] S. Misra and P. Agarwal, "Bio-inspired group mobility model for mobile ad hoc networks based on bird-flocking behavior," *Soft Comput.*, vol. 16, no. 3, pp. 437–450, Mar. 2012.

[134] R. Duan, X. Chen, and T. Xing, "A QoS architecture for IOT," in *Proc. Int. Conf. 4th iThings/CPSCom*, 2011, pp. 717–720.

[135] M. A. Rajan, P. Balamuralidhar, K. P. Chethan, and M. Swarnahpriyaah, "A self-reconfigurable sensor network management system for Internet of Things paradigm," in *Proc. ICDeCom*, 2011, pp. 1–5.

[136] OMA Lightweight M2M, Sep. 28, 2014. [Online]. Available: http://technical.openmobilealliance.org/Technical/technical-information/release-program/current-releases/oma-lightweightm2m-v1-0-2

[137] V. Perelman, M. Ersue, J. Schönwälder, and K. Watsen, "Network Configuration Protocol Light (NETCONF Light)," Internet Eng. Task Force (IETF), Fremont, CA, USA, Network, 2012.

[138] MASH IoT Platform, Youtube Channel, Sep. 28, 2014. [Online]. Available: http://www.youtube.com/user/MASHPlatform

[139] F. Van den Abeele, J. Hoebeke, I. Moerman, and P. Demeester, "Fine-grained management of CoAP interactions with constrained IoT devices," in *Proc. IEEE NOMS*, 2014, pp. 1–5.

[140] D. Uckelmann, M. Isenberg, M. Teucke, H. Halfar, and B. Scholz-Reiter, "Autonomous control and the Internet of Things: Increasing robustness, scalability and agility in logistic networks," *Unique Radio Innovation for the 21st Century*, pp. 163–181, 2010.

[141] C. Sarkar, S. N. A. U. Nambi, R. V. Prasad, and A. Rahim, "A scalable distributed architecture towards unifying IoT applications," in *Proc. IEEE WF-IoT*, 2014, pp. 508–513.

[142] F. Li, M. Voegler, M. Claessens, and S. Dustdar, "Efficient and scalable IoT service delivery on cloud," in *Proc. IEEE 6th Int. Conf. CLOUD*, 2013, pp. 740–747.

[143] A. Dunkels, J. Eriksson, and N. Tsiftes, "Low-power interoperability for the IPv6-based Internet of Things," in *Proc. 10th Scandinavian Workshop Wireless ADHOC*, Stockholm, Sweden, 2011, pp. 10–11.

[144] I. Ishaq *et al.*, "IETF standardization in the field of the Internet of Things (IoT): A survey," *J. Sens. Actuator Netw.*, vol. 2, pp. 235–287, 2013.

[145] L. Zhang *et al.*, "Named data networking," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 66–73, 2014.

[146] S. K. Datta, C. Bonnet, and N. Nikaein, "An IoT gateway centric architecture to provide novel M2M services," in *Proc. IEEE World Forum WF-IoT*, 2014, pp. 514–519.

[147] A. P. Castellani *et al.*, "Architecture and protocols for the Internet of Things: A case study," in *Proc. 8th IEEE Int. Conf. PERCOM Workshops*, 2010, pp. 678–683.

[148] I. Silva, L. A. Guedes, P. Portugal, and F. Vasques, "Reliability and availability evaluation of wireless sensor networks for industrial applications," *Sensors*, vol. 12, no. 1, pp. 806–838, 2012.

[149] D. G. Costa, I. Silva, L. A. Guedes, F. Vasques, and P. Portugal, "Availability issues in wireless visual sensor networks," *Sensors*, vol. 14, no. 2, pp. 2795–2821, Feb. 2014.

[150] G. Wang, S. Chen, H. Lu, and M. Lin, "Pa-GFDP: An algorithm enhancing reliability of WSNs," *WSEAS Trans. Comput.*, vol. 11, no. 12, pp. 445–454, Dec. 2012.

[151] A. Dâmaso, N. Rosa, and P. Maciel, "Reliability of wireless sensor networks," *Sensors*, vol. 14, pp. 15 760–15 785, 2014.

[152] E. Z. Tragos *et al.*, "Enabling reliable and secure IoT-based smart city applications," in *Proc. IEEE Int. Conf. PERCOM Workshops*, 2014, pp. 111–116.

[153] F. Shaoshuai, S. Wenxiao, W. Nan, and L. Yan, "MODM-based evaluation model of service quality in the Internet of Things," *Procedia Environ. Sci.*, vol. 11, pp. 63–69, 2011.

[154] X. Che and S. Maag, "A passive testing approach for protocols in Internet of Things," in *Proc. IEEE Int. Conf. IEEE Cyber, Phys. Soc. Comput. GreenCom, iThings/CPSCom*, 2013, pp. 678–684.

[155] W. Pöttner, H. Seidel, J. Brown, U. Roedig, and L. Wolf, "Constructing schedules for time-critical data delivery in wireless sensor networks," *Proc. ACM TOSN*, vol. 10, no. 3, p. 44, Apr. 2014.

[156] G. Colistra, V. Pilloni, and L. Atzori, "Task allocation in group of nodes in the IoT: A consensus approach," in *Proc. IEEE ICC*, 2014, pp. 3848–3853.

[157] R. Lan-Lan, M. Luo-Ming, Q. Xue-Song, and Z. Jie, "Integrated management model for terminal devices in pervasive communication networks," in *Proc. 2nd ICCAE*, 2010, pp. 249–253.

[158] P. Castillejo, J. Martínez, L. López, and G. Rubio, "An Internet of Things approach for managing smart services provided by wearable devices," *Int. J. Distrib. Sens. Netw.*, 2013.

[159] Kura—OSGi-Based Application Framework for M2M Service Gateways, eclipse, Ottawa, ON, Canada, Sep. 25, 2014. [Online]. Available: http://www.eclipse.org/proposals/technology.kura/

[160] S. Ziegler, C. Crettaz, and I. Thomas, "IPv6 as a global addressing scheme and integrator for the Internet of Things and the cloud," in *Proc. 28th Int. Conf. WAINA*, 2014, pp. 797–802.

[161] X. Wang, J. Zhang, E. M. Schooler, and M. Ion, "Performance evaluation of attribute-based encryption: Toward data privacy in the IoT," in *Proc. IEEE ICC*, 2014, pp. 725–730.

[162] F. Bao, I. Chen, and J. Guo, "Scalable, adaptive and survivable trust management for community of interest based Internet of Things systems," in *Proc. IEEE 11th ISADS*, 2013, pp. 1–7.

[163] S. L. Keoh, S. S. Kumar, and H. Tschofenig, "Securing the Internet of Things: A Standardization Perspective," *IEEE Internet Things J.*, vol. 1, no. 3, pp. 265–275, Jun. 2014.

[164] B. Rao, P. Saluia, N. Sharma, A. Mittal, and S. Sharma, "Cloud computing for Internet of Things and sensing based applications," in *Proc. 6th ICST*, 2012, pp. 374–380.

[165] R. Bryant, R. H. Katz, and E. D. Lazowska, "Big-data computing: Creating revolutionary breakthroughs in commerce, science, and society," Comput. Commun. Consortium (CCC), Washington, DC, USA, 2008.

[166] C. Tsai, C. Lai, M. Chiang, and L. T. Yang, "Data mining for Internet of Things: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 77–97, 1st Quart. 2014.

[167] D. Borthakur *et al.*, "Apache Hadoop goes realtime at Facebook," in *Proc. 2011 ACM SIGMOD Int. Conf. Management Data*, 2011, pp. 1071–1080.

[168] A. Mukherjee, H. S. Paul, S. Dey, and A. Banerjee, "ANGELS for distributed analytics in IoT," in *Proc. IEEE WF-IoT*, 2014, pp. 565–570.

[169] X. Xu, S. Huang, Y. Chen, K. Browny, I. Halilovicy, and W. Lu, "TSAaaS: Time series analytics as a service on IoT," in *Proc. IEEE ICWS*, 2014, pp. 249–256.

[170] D. C. Verma and P. Verma, *Techniques for Surviving Mobile Data Explosion (Chapter 8)*. New York, NY, USA: Wiley, 2014.

[171] K. Yang *et al.*, "Park-a-lot: An automated parking management system," *Comput. Sci. Inform. Technol.*, vol. 1, no. 4, pp. 276–279, 2013.

[172] C. Doukas, *Building Internet of Things With the ARDUINO*, CreateSpace Independent Pub. Platform, Dougherty, GA, USA, 2012.

[173] Nimbits, Sep. 25, 2014. [Online]. Available: http://www.nimbits.com/

[174] O. Mazhelis and P. Tyrvainen, "A framework for evaluating Internet-of-Things platforms: Application provider viewpoint," in *Proc. IEEE WF-IoT*, 2014, pp. 147–152.

[175] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st Edition MCC Workshop Mobile Cloud Comput.*, 2012, pp. 13–16.

[176] B. Rochwerger *et al.*, "The reservoir model and architecture for open federated cloud computing," *IBM J. Res. Develop.*, vol. 53, no. 4, pp. 535–545, Jul. 2009.

[177] G. C. Fox, S. Kamburugamuve, and R. D. Hartman, "Architecture and measured characteristics of a cloud based Internet of Things," in *Proc. Int. Conf. CTS*, 2012, pp. 6–12.

[178] H. Madsen, G. Albeanu, B. Burtschy, and F. L. Popentiu-Vladicescu, "Reliability in the utility computing era: Towards reliable fog computing," in *Proc. 20th IWSSIP*, 2013, pp. 43–46.

[179] H. Chang, A. Hari, S. Mukherjee, and T. V. Lakshman, "Bringing the cloud to the edge," in *Proc. IEEE Conf. INFOCOM WKSHPS*, 2014, pp. 346–351.

[180] Ponte—M2M Bridge framework for REST developers, Eclipse, Ottawa, ON, USA, Sep. 25, 2014. [Online]. Available: http://eclipse.org/proposals/technology.ponte/

[181] M. Collina, G. E. Corazza, and A. Vanelli-Coralli, "Introducing the QEST broker: Scaling the IoT by bridging MQTT and REST," in *Proc. IEEE 23rd Int. Symp. PIMRC*, 2012, pp. 36–41.

[182] Eclipse IoT, Sep. 25, 2014. [Online]. Available: https://projects.eclipse.org/projects/iot

[183] Eclipse SCADA, Sep. 25, 2014. [Online]. Available: http://projects.eclipse.org/projects/technology.eclipsescada

[184] Eclipse SmartHome, Sep. 25, 2014. [Online]. Available: http://eclipse.org/proposals/technology.smarthome/

[185] Eclipse Krikkit, Sep. 25, 2014. [Online]. Available: http://eclipse.org/proposals/technology.krikkit/

[186] Q. Zhu, R. Wang, Q. Chen, Y. Liu, and W. Qin, "IOT gateway: Bridging wireless sensor networks into Internet of Things," in *Proc. IEEE/IFIP 8th Int. Conf. EUC*, 2010, pp. 347–352.

[187] S. Guoqiang, C. Yanming, Z. Chao, and Z. Yanxu, "Design and implementation of a smart IoT gateway," in *Proc. IEEE Int. Conf. IEEE Cyber, Phys. Soc. Comput. GreenCom, iThings/CPSCom*, 2013, pp. 720–723.

[188] L. Wu, Y. Xu, C. Xu, and F. Wang, "Plug-configure-play service-oriented gateway-for fast and easy sensor network application development," in *Proc. SENSORNETS*, 2013, pp. 53–58.

[189] J. Vermillard, "M2M, IoT, device management: One protocol to rule them all?" EclipseCon, San Francisco, CA, USA, 2014.

[190] I. Ishaq, J. Hoebeke, I. Moerman, and P. Demeester, "Internet of Things virtual networks: Bringing network virtualization to resource-constrained devices," in *Proc. IEEE Int. Conf. GreenCom*, 2012, pp. 293–300.

[191] X. Wang, J. T. Wang, X. Zhang, and J. Song, "A multiple communication standards compatible IoT system for medical usage," in *Proc. IEEE FTFC*, 2013, pp. 1–4.

[192] Y. H. Lin, Q. Wang, J. S. Wang, L. Shao, and J. Tang, "Wireless IoT platform based on SDR technology," in *Proc. IEEE Int. Conf. IEEE Cyber, Phys. Soc. Comput. GreenCom, iThings/CPSCom*, 2013, pp. 2245–2246.
[193] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, "A software defined networking architecture for the Internet-of-things," in *Proc. IEEE NOMS*, 2014, pp. 1–9.
[194] A. Al-Fuqaha, A. Khreishah, M. Guizani, A. Rayes, and M. Mohammadi, "Towards better horizontal integration among IoT services," *IEEE Commun. Mag.*, Aug. 2015, to be published.
[195] M. T. Goodrich, and M. Mitzenmacher, "Invertible bloom lookup tables," in *Proc. 49th Annu. Allerton*, 2011, pp. 792–799.

Ala Al-Fuqaha (S'00–M'04–SM'09) received the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Missouri-Columbia, Columbia, MO, USA, and the University of Missouri-Kansas City, Kansas City, MO, USA, in 1999 and 2004, respectively. He is currently an Associate Professor and the Director of the NEST Research Laboratory, Department of Computer Science, Western Michigan University, Kalamazoo, MI, USA. He served as the Principal Investigator (PI) or Co-PI on multiple research projects funded by NSF, Qatar Foundation, Cisco, Boeing, AVL, Stryker, Wolverine, Traumasoft, and Western Michigan University. His research interests include wireless vehicular networks (VANETs), cooperation and spectrum access etiquettes in cognitive radio networks, smart services in support of the Internet of Things, management and planning of software-defined networks (SDN), intelligent services for the blind and the visually impaired, QoS routing in optical and wireless networks, and performance analysis and evaluation of high-speed computer and telecommunication networks.

Dr. Al-Fuqaha has served as a Technical Program Committee Member and a reviewer of many international conferences and journals. He is currently serving on the Editorial Board of John Wiley's *Security and Communication Networks* journal, John Wiley's *Wireless Communications and Mobile Computing* journal, *EAI Transactions on Industrial Networks and Intelligent Systems*, and *International Journal of Computing and Digital Systems*. He was the recipient of the Outstanding Researcher Award at the College of Engineering and Applied Sciences, Western Michigan University, in 2014.

Mohsen Guizani (S'85–M'89–SM'99–F'09) received the B.S. (with distinction) and M.S. degrees in electrical engineering and the M.S. and Ph.D. degrees in computer engineering from Syracuse University, Syracuse, NY, USA, in 1984, 1986, 1987, and 1990, respectively. He is currently a Professor at the Department of Computer Science and Engineering, Qatar University (QU), Doha, Qatar. Previously, he served as the Associate Vice President of Graduate Studies at QU in 2011–2014; the Chair of the Department of Computer Science, Western Michigan University, in 2002–2006; and the Chair of the Department of Computer Science, University of West Florida, in 1999–2002. He also served in academic positions at the University of Missouri-Kansas City, University of Colorado-Boulder, Syracuse University, and Kuwait University.

His research interests include wireless communications and mobile computing, computer networks, cloud computing, cyber security, and smart grid. Dr. Guizani is a Senior Member of ACM and a member of the IEEE Communications Society, IEEE Computer Society, and ASEE. He currently serves on the Editorial Boards of several international technical journals and the Founder and Editor-in-Chief of Wiley's *Wireless Communications and Mobile Computing* (http://www.interscience.wiley.com/jpages/1530-8669/). He is the author of nine books and more than 400 publications in refereed journals and conferences (with an h-index=30 according to Google Scholar). He guest edited a number of special issues in IEEE journals and magazines. He also served as a Member, Chair, and General Chair of a number of conferences. He was the Chair of the IEEE Communications Society Wireless Technical Committee (WTC 2009–2010) and the Chair of the Transmission, Access and Optical Systems (TAOS 2007–2009). He served as the IEEE Computer Society Distinguished Speaker from 2003 to 2005. He received the Best Research Award from two institutions.

Mehdi Mohammadi (S'14) received the B.S. degree in computer engineering from Kharazmi University, Tehran, Iran, in 2003 and the M.S. degree in computer engineering (software) from Sheikhbahaee University (SHBU) Isfahan, Iran, in 2010. He is currently working toward the Ph.D. degree with the Department of Computer Science, Western Michigan University (WMU), Kalamazoo, MI, USA. His research interests include Internet of Things, future internet, software-defined networking, cloud computing, data mining, and natural language processing.

Mr. Mohammadi has served as a Co-Investigator in a research project that performs automatic human language translation funded by Unitec Institute of Technology, Auckland, New Zealand. He also serves as a reviewer for Wiley's *Security and Wireless Communication Networks* journal and a Technical Program Committee (TPC) Member of IoT-NAT 2015 conference. He received a Graduate Doctoral Assistantship from the WMU Libraries Information Technology in 2013. He was the recipient of four travel grants from the National Science Foundation to attend the GENI conference.

Mohammed Aledhari (S'14) received the B.E. degree in computer science from the University of Anbar, Ramadi, Iraq, in 2003 and the M.S. degree in computer science from the University of Basrah, Basra, Iraq, in 2010. He is currently working toward the Ph.D. degree with the Department of Computer Science, Western Michigan University, Kalamazoo, MI, USA. He is currently a Research Assistant at the Center for High Performance Computing and Big Data (CHPCBD), Western Michigan University, to design and implement a novel data-aware transfer protocol for big genomic data. Prior to joining CHPCBD, he served as a Research Assistant at the Computer Networks, Embedded Systems and Telecommunications (NEST) Laboratory. He also served as a Microcontrollers and PLC Embedded Software Engineer at the Iraqi Drilling Company (IDC) from January 2005 to August 2011. His current research interests include Internet of Things, networks, big data, cloud computing, machine learning, microcontrollers, and robots. Mr. Aledhari was a recipient of many awards from the Iraqi Oil Minister for his contributions to the field of microcontrollers. He also was a recipient of the High Education Minister Award in Iraq for holding the highest GPA in his undergraduate program.

Moussa Ayyash (M'98–SM'12) received the B.Sc. degree in electrical and communications engineering from Mu'tah University, Maw'tah, Jordan, in 1997, the M.Sc. degree in electrical and communications engineering from the University of Jordan, Amman, Jordan, in 1999, and the Ph.D. degree in electrical and computer engineering from Illinois Institute of Technology, Chicago, IL, USA, in 2005. He is currently an Associate Professor at the Department of Information Studies, Chicago State University, Chicago. He is the Director of the Center of Information and National Security Education and Research. His current research interests span digital and data communication areas, wireless networking, visible light communications, Internet of Things, and interference mitigation. Dr. Ayyash is a member of the IEEE Computer and Communications Societies and a member of the Association for Computing Machinery.