The Internet of Things (IoT): a survey of techniques, operating systems, and trends

Elham Ali Shammar and Ammar Thabit Zahary
Department of Information Technology,
Faculty of Computer and Information Technology, Sana'a University, Sana'a, Yemen

Received 18 December 2018
Revised 24 March 2019
3 June 2019
Accepted 8 August 2019

Abstract

Purpose – Internet has changed radically in the way people interact in the virtual world, in their careers or social relationships. IoT technology has added a new vision to this process by enabling connections between smart objects and humans, and also between smart objects themselves, which leads to anything, anytime, anywhere, and any media communications. IoT allows objects to physically see, hear, think, and perform tasks by making them talk to each other, share information and coordinate decisions. To enable the vision of IoT, it utilizes technologies such as ubiquitous computing, context awareness, RFID, WSN, embedded devices, CPS, communication technologies, and internet protocols. IoT is considered to be the future internet, which is significantly different from the Internet we use today. The purpose of this paper is to provide up-to-date literature on trends of IoT research which is driven by the need for convergence of several interdisciplinary technologies and new applications.

Design/methodology/approach – A comprehensive IoT literature review has been performed in this paper as a survey. The survey starts by providing an overview of IoT concepts, visions and evolutions. IoT architectures are also explored. Then, the most important components of IoT are discussed including a thorough discussion of IoT operating systems such as Tiny OS, Contiki OS, FreeRTOS, and RIOT. A review of IoT applications is also presented in this paper and finally, IoT challenges that can be recently encountered by researchers are introduced.

Findings – Studies of IoT literature and projects show the disproportionate importance of technology in IoT projects, which are often driven by technological interventions rather than innovation in the business model. There are a number of serious concerns about the dangers of IoT growth, particularly in the areas of privacy and security; hence, industry and government began addressing these concerns. At the end, what makes IoT exciting is that we do not yet know the exact use cases which would have the ability to significantly influence our lives.

Originality/value – This survey provides a comprehensive literature review on IoT techniques, operating systems and trends.

Keywords Internet of Things, IoT, Middleware, Ubiquitous Computing, Context Aware Systems, embedded devices

Paper type Literature review

1. Introduction

Today, the internet is dominated by Web 2.0, but the researchers have been working on another aim of making the internet behave more intelligently, which is referred to as Web 3.0 or Semantic Web. In the Semantic Web, the web content becomes more understandable by machines and it can be processed and shared by the machines themselves. The machines and search engines behave more intelligently without the need for human intervention (Whitmore *et al.*, 2015). Many things in our daily lives have become wirelessly connected with miniatures and low-powered wireless devices such as radio-frequency identification (RFID) tags, near-field communication (NFC), Electronic Product Code (EPC), mobile devices and GPS, along with the development of many technologies such as sensors, actuators, embedded computing, machine-to-machine (M2M) communication and cloud computing. It was expected that there will be 7 trillion wireless devices to serve 7bn people (1,000 devices/person). This ultra-huge number of connected devices or things and the new techniques have introduced a paradigm, commonly referred to as the Internet of Things (IoT) (Razzaque *et al.*, 2016). In IoT, every vital object in our daily life such as the wallet, watch, refrigerator, car, etc., will be

Library Hi Tech
Vol. 38 No. 1, 2020
pp. 5-66

0737-8831
DOI 10.1108/LHT-12-2018-0200

connected with each other and with the internet. Anyone would be able to track his/her belongings from anywhere, anytime and any network. Those objects would send alerts to the users, in any event, to keep them safe. Hence, this revolution will change the way people think, live and work (Singh *et al.*, 2014).

IoT started in 1998, and the term IoT was originally coined by Kevin Ashton in 1999. IoT is an expansion of Mark Weiser's vision of ubiquitous computing (UbiComp), which aims to produce a global network that supports UbiComp and context awareness among devices. Establishing IoT is based on the proliferation of wireless sensor network (WSN), mobile computing (MobiComp), UbiComp and information technologies. Ambient intelligence is one of the key components of IoT. UbiComp and context awareness are the main requirements of ambient intelligence, in which everyday devices would be able to understand their environment, interact with each other and with humans and make the decisions. IoT application scenarios require applications to demonstrate their adaptability to very diverse contexts with different available resources and possibly changing deployment environments over time. Since the origin of IoT, it has been evolving in a tremendous way and it is still an emerging trend for researchers in both academia and industry (Miorandi *et al.*, 2012; Porkodi and Bhuvaneswari, 2014; Whitmore *et al.*, 2015; Zhang *et al.*, 2011).

The National Intelligence Council and McKinsey Global Institute announced that everyday things like food packages, furniture, paper documents, etc., will represent internet nodes by 2025. They highlighted the future that would be created by combining the technologies that interact with the human environment. The information technology and embedded technology would have a potential impact on the revolution of using smart objects, which play a main role in the IoT vision (Asghar *et al.*, 2015; Atzori *et al.*, 2010). It is impractical, in a UbiComp environment like IoT, to make standards and enforce everyone to adhere to them. Having a very wide network of things and a large number of events that can be automatically generated by those things, coupled with heterogeneous devices/technologies/applications in the IoT, brings new challenges in the application development and makes the existing challenges in UbiComp much more tricky (Razzaque *et al.*, 2016).

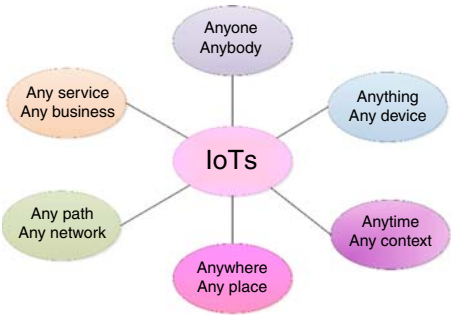
The heterogeneous nature and the increasing popularity of IoT and the rapid growth of IoT articles pose challenges to provide an appropriate synthesizing of previous IoT literature. IoT's research studies mostly come with different points of concentration, which hinder the understanding and the accumulation of knowledge in the existing literature. Moreover, the exponential growth of IoT research aggravates the problem, leading to an invincible obstacle in conducting a systematic review of IoT literature. Liu *et al.* (2017) conducted a systematic literature review on prior IoT research through a combination of both quantitative and qualitative approaches. The researchers aimed to address the aforementioned challenges by reviewing 1,065 articles on IoT obtained from the International Statistical Institute Web of Science through a combination of quantitative citation analysis using HistCite and qualitative content analysis by qualifying the knowledge progression chronologically.

This survey will provide up-to-date literature on IoT research trends driven by the need for the convergence of several interdisciplinary technologies and new applications. This survey can contribute to the future development and research on IoT. This paper is organized as follows: Section 2 will explore the IoT concepts and visions. IoT evolution will be presented in Section 3. Section 4 will explore the IoT architecture. IoT components will be discussed thoroughly in Section 5, including the review of the most recent IoT operating systems (OSs). IoT enabling technologies and trends will be provided in Section 6. Section 7 and Section 8 will take a glance in IoT standardizations and IoT protocols. IoT cybersecurity and quality of service (QoS) will be introduced in Sections 9 and 10, respectively. Several application domains in IoT will be reviewed in Section 11. Section 12 will provide the challenges that face the IoT vision and open research issues. Finally, the paper will be concluded in Section 13.

2. IoT concept and visions

In 1999, the concept of IoT was first proposed by Kevin Ashton as interconnected objects that can be uniquely identified with RFID technology. Nevertheless, the exact definition of IoT is still in the formation phase which depends on the researchers' points of view (Bandyopadhyay and Sen, 2011; Li *et al.*, 2015; Perera *et al.*, 2014). Even if the traditional concept of the internet will vanish, the internet infrastructure itself will not disappear. The internet will keep its vital role as a global backbone for sharing and disseminating information around the world. The concept of smart interconnected objects, which constitute widespread computing environments, will be enabled by integrating electronics into everyday physical objects and allowing those objects to seamlessly, smoothly and intelligently integrate into the resulting global physical infrastructure. In IoT, the connection will not be merely between end-user devices, but also between the physical objects that communicate with each other and/or humans in order to provide a particular service. The traditional methods of networking, computing and administration would also be changed (Miorandi *et al.*, 2012). The outstanding goal of IoT is to create a better world for the human beings, where things around us know what we love, want and need, thus taking the proper action accordingly without explicit instructions from us (Perera *et al.*, 2014). To summarize, the IoT is the next generation of the internet, by which any physical objects can be connected, accessed and identified over the internet (Li *et al.*, 2015). Figure 1 shows the different aspects of the IoT concept.

Large differences in IoT insights emerge from the fact that IoT includes miscellaneous stakeholders, businesses, research studies and standardizations. Everybody is trying to interpret and define the IoT according to his/her specific needs, concerns and backgrounds (Atzori *et al.*, 2010; Singh *et al.*, 2014). The two words "internet" and "things", when they come together, carry the meaning that offers a fascinating level of innovation in the world of information and communication technology (ICT). The first one moves the focus toward an "Internet-oriented vision" of IoT, whereas the second one moves the attention to the second vision, "Things-oriented vision." Both are integrated to form a common framework in which a huge number of heterogeneous things/objects are connected to each other (Atzori *et al.*, 2010; Bandyopadhyay and Sen, 2011). These visions involve a massive amount of data that would be generated and collected from vast numbers of sensors. Therefore, we will have an enormous amount of information, maybe redundant, that must be addressed meaningfully. This concept moves the attention to the third vision, "Semantic-oriented vision." With this vision, the huge generated data would be managed and processed to get better representations and understanding and to help in making the proper decision. Figure 2 shows the IoT vision from the aforementioned perspectives: internet-oriented, things-oriented (smart things and sensors) and semantic-oriented (knowledge)



Sources: Perera *et al*. (2014), Razzaque *et al*. (2016)

Figure 1.
IoT concept aspects

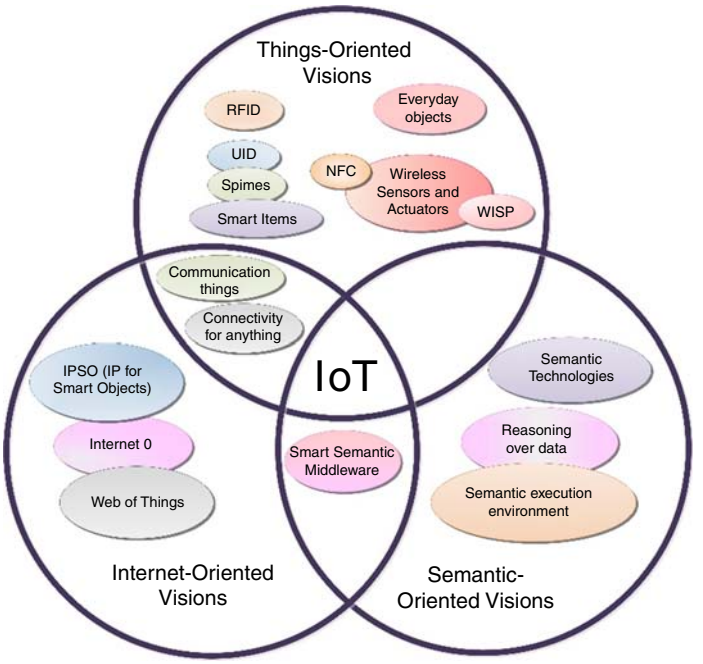


Figure 2.
IoT paradigm as a result of the convergence of different visions

Sources: Atzori *et al*. (2010), Bandyopadhyay and Sen (2011), Singh *et al*. (2014)

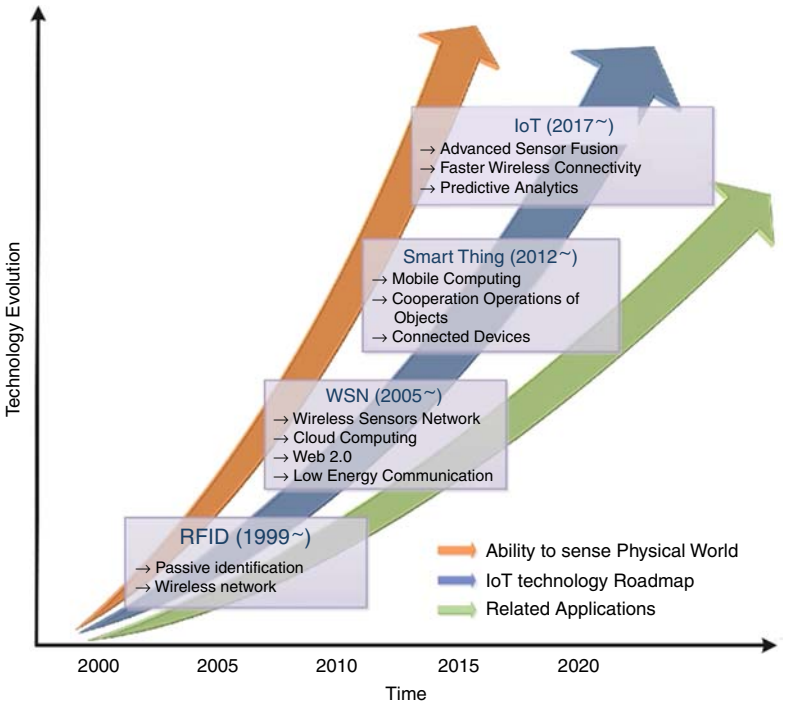
perspectives. The key concepts, techniques and standards are mentioned and categorized with the reference to these three visions. From this illustration, it seems clear that the convergence of the three main visions produces the novel IoT paradigm (Atzori *et al.*, 2010; Bandyopadhyay and Sen, 2011; Singh *et al.*, 2014).

3. IoT evolution

The evolution of IoT can be illustrated through several stages as shown in Figure 3. The term IoT was initially introduced to refer to interoperable and uniquely identifiable connected objects using RFID technology, which is increasingly used for identification in retail, logistics, pharmaceutical production and various industries. Later on, researchers connected IoT with further technologies such as sensors, actuators, mobile devices and GPS devices (Da Xu *et al.*, 2014). A number of technologies are involved in IoT, like WSNs, intelligent sensing, RFID, barcodes, NFC, low power, cloud computing, wireless communications and others, which expand the original concept of IoT into ambient intelligence and autonomous control. The developments of previously mentioned technologies bring new technologies to IoT (Li *et al.*, 2015).

4. IoT architecture

Currently, the internet is using TCP/IP protocol layered stack, which was introduced long time ago for communication between network hosts. Nevertheless, IoT makes it difficult for the TCP/IP protocol because IoT will connect everything and everyone. For the exchange of information, billions of objects would be connected. IoT will include an increasing number of interconnected sensors and smart devices and an extremely wide range of technologies. Architecture solutions will need to include all those diverse devices and technologies. IoT architecture solutions should



Source: Li *et al*. (2015)

Figure 3.
Evolution of IoT

also be flexible to meet the needs of identification (RFID tags), intelligent devices and smart objects (hardware and software solutions). Moreover, the architecture solutions will require to combine volume of data from various sources, identify relevant features, interpret data, show relationships, compare data with useful historical information and extract knowledge to support making decisions. Furthermore, architecture solutions must be open and they must not limit users to use fixed end-to-end connection. The new proposed architecture needs to address these factors and many others such as scalability, interoperability, reliability, QoS, etc. Single reference architecture cannot be used as a blueprint for all applications; thus, heterogeneous reference architectures in IoT have to coexist (Chen *et al.*, 2014; Khan *et al.*, 2012).

Most of the IoT architecture-related works have been categorized into four types of architectures, as stated in Porkodi and Bhuvaneswari (2014): the WSNs, the European Union's (EU) SENSEI projects, Internet of Things Architecture (IoT-A) and cloud architecture. Each project is trying to design a common architecture based on the analysis of the needs of researchers and industry. Nevertheless, this may not be the best choice for every application domain. No single architecture has been converged into a reference model. The choice of IoT architecture itself is a big challenge, paving the way for the development of a new architecture and the modification of the current architectures. Since IoT must be able to connect billions or trillions of heterogeneous objects over the internet, there is a crucial need for a flexible layered architecture (Al-Fuqaha *et al.*, 2015; Porkodi and Bhuvaneswari, 2014; Singh *et al.*, 2014).

IoT layered architecture consists of several layers ranging from the data acquisition layer at the bottom to the application layer at the top and the internet layer in between to serve as a common media for communication. The layered architecture would provide features that can meet the requirements of diverse industries, enterprises, institutes, societies, governments

(Bandyopadhyay and Sen, 2011), etc. From the bunch of proposed architecture solutions, the most widespread suggested models are the three-layer architecture and the five-layer architecture (Al-Fuqaha *et al.*, 2015). The three-layer architecture contains the following layers: application, network and perception layers (Al-Fuqaha *et al.*, 2015; Yang *et al.*, 2011; Zhang *et al.*, 2012). The five-layer architecture adds two additional layers to the three-layer architecture. The five-layer architecture includes business layer, application layer, middleware layer, network layer and perception layer (Khan *et al.*, 2012).

4.1 The three-layer architecture

At the inception of IoT, the accepted proposed architecture was the three-layer architecture, which consists of three layers: perception, network and application. The perception layer serves as the nerve endings such as people's eyes, ears, nose and throat. Its main functions are identifying things and collecting information. The perception layer consists of RFID tags, sensors, GPS, cameras, etc. The network layer is the core of the IoT, which serves as a person's brain. It is used to transmit and cope with the information from the perception layer. The network layer is established on the basis of the current mobile telecommunication, the internet backbone and various information networks. Its main feature is the transmission of information over long distances. The network layer consists of wired and wireless communication networks, the internet and different private networks. In addition, it includes cloud computing, platforms and expert systems, which intelligently process the massive information. The application layer is the interface between IoT and users like humans, organizations and other systems. It is a connection between IoT technologies and sector professional technologies. The main function of the application layer is to discover the services and to deliver them. The application layer also recognizes wide intelligent applications by providing various solutions. It is connected with the needs of the industry to achieve the intelligent applications of IoT (Yang *et al.*, 2011; Zhang *et al.*, 2012).

4.2 The five-layer architecture

The five-layer architecture is the most widely used architecture solution for IoT applications. It adds two additional layers in addition to the three layers of three-layer architecture. The two added layers are the middleware layer and the business layer. The perception layer on the five-layer architecture has the same tasks of the perception layer on the three-layer architecture. The network layer or transmission layer looks like the transport layer of the OSI model. It collects the data captured by the perception layer to send them to the internet. The transmission medium can be wireless or wired and the used technologies can be WiFi, ZigBee, Bluetooth, infrared, 3G, UMTS, etc. The middleware layer receives the information from the network layer and stores it in the database. It performs UbiComp and information processing and takes the proper decisions based on the results. The application layer performs the final presentation of data depending on the desired application. The last layer is the business layer that is responsible for the management of the whole IoT system including the services and applications. The business layer builds the graphs, flowcharts and business models based on the data received from the application layer. This layer will also help in anticipating and identifying future actions and business strategies based on the analysis of the results (Aazam *et al.*, 2014; Khan *et al.*, 2012).

The architectures that borrow their layers and concepts from network stacks, such as the three-layer model, do not comply with real IoT environments because the network layer, for example, does not cover all the basic technologies that transfer data on the IoT platform. Moreover, these models are designed to handle certain types of communication media like WSNs. The SOA-based architecture has a layer called "Service Composition" that takes rather a significant part of the device's time and energy to connect with other devices and

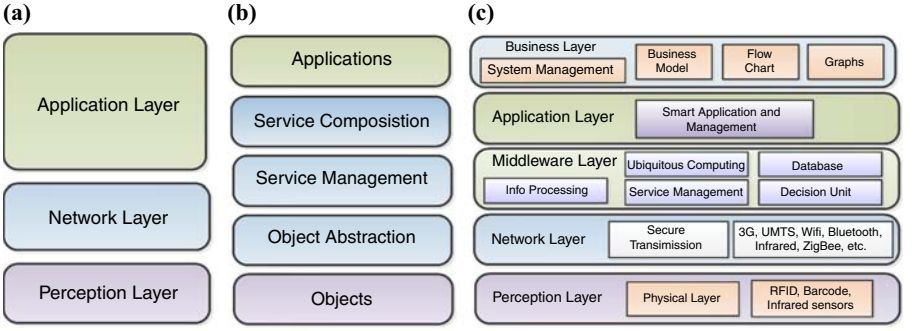
integrate the required services. This problem is solved in the five-layer model. The control mechanisms of accessing data in the application layer are managed on the business layer, which is hosted on powerful devices; thus, it adheres to the simplicity of IoT architecture. Therefore, the five-layer architecture is the most widely used architecture solution for IoT applications (Al-Fuqaha *et al.*, 2015). Figure 4 shows the most common IoT architectures.

However, in recent studies, other architecture solutions for IoT have been proposed, which add more abstraction to the IoT architecture. Zhang *et al.* (2012) described a new extent six-layer architecture, as shown in Figure 5. This model consists of six layers: the coding layer, the information acquisition layer, the information access layer, the network layer, the information integration layer and the application service layer. The coding layer gives the objects their ID numbers to be easily identifiable in the IoT whole cycle. The information acquisition layer has the same tasks of the perception layer on the three-layer architecture and the five-layer architecture. The information access layer is used to transmit the data obtained from the lower layers to the network layer. The network layer in this architecture performs the same work of the network layer on the previous models. The information integration layer used to manage and control the huge and uncertain data in the network in real time. Then the data are reorganized, filtered, merged and converted to a content service in the SOA. Therefore, it provides a good service interface to the application service layer. The application service layer integrates the service and constitutes the application services for various industries.

Sarkar *et al.* (2014) proposed a layered distributed architecture for IoT, which is called Distributed Internet-like Architecture for the Internet of Things (DIAT), as shown in Figure 6. This architecture provides different levels of abstraction to address issues such as interoperability, heterogeneity and scalability. Some features are integrated into DIAT such as intelligence, automation and zero configuration. The authors have shown that DIAT fulfills the main characteristics and objectives of IoT architecture through a real case study test. The functionalities of IoT infrastructure are classified into three layers: the virtual object layer, responsible for object virtualization, the composite virtual object layer, responsible for service composition and execution, and the service layer (SL), responsible for service creation and management.

5. IoT components

IoT has many diverse components at each layer on the IoT architecture. Those components can be grouped into the following categories.



Notes: (a) Three-layer IoT architecture; (b) SOA-based IoT architecture (Al-Fuqaha *et al.*, 2015); (c) five-layer IoT architecture (Khan *et al.*, 2012)

Figure 4.
IoT architecture

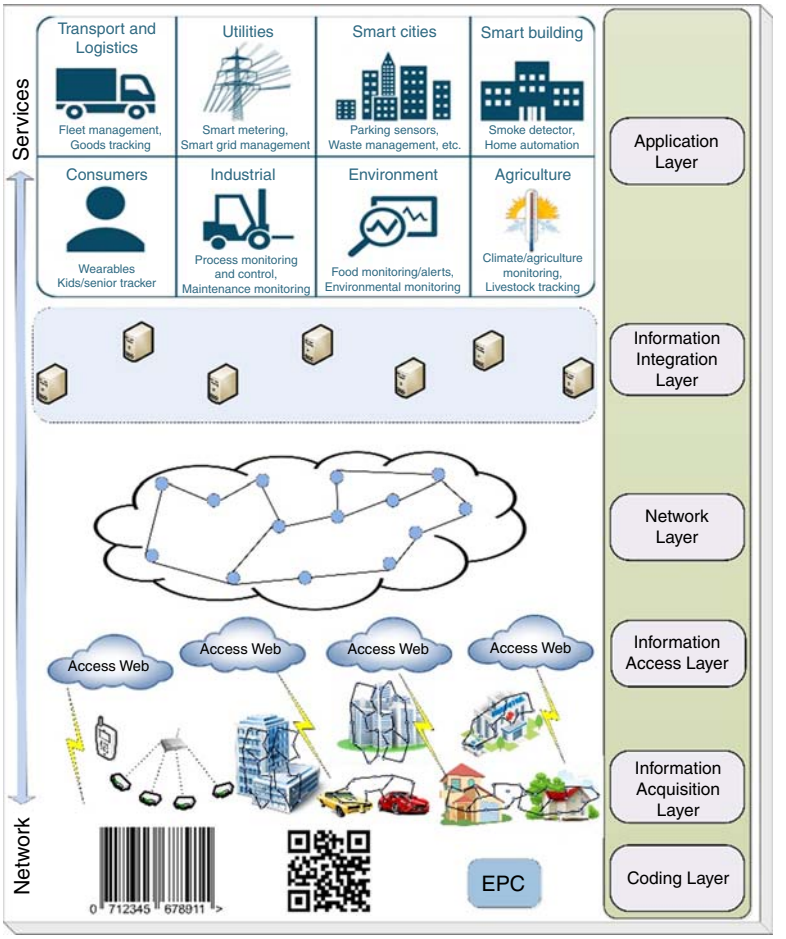


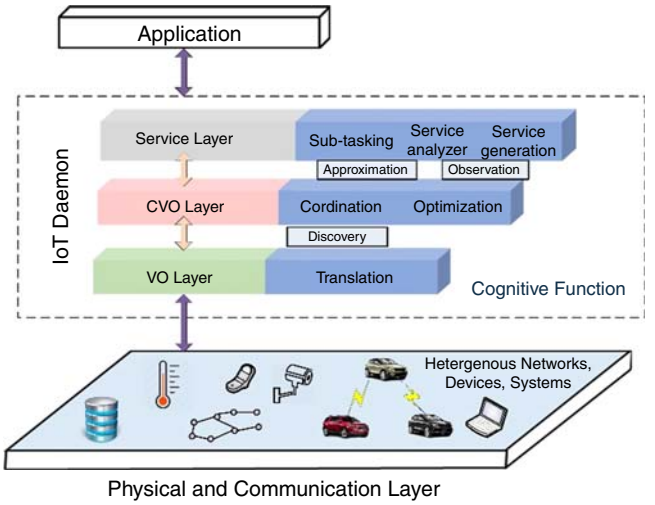
Figure 5.
The extent six-layer architecture of IoT

Source: Zhang *et al*. (2012)

5.1 Hardware

Most of the hardware upon which the IoT is built already exists and is used pervasively. The most important hardware infrastructure comprises RFID, NFC and sensor networks (Al-Fuqaha *et al.*, 2015).

5.1.1 Radio-frequency identification. RFID is a short-range communication technology. It is one of the main factors in the embedded communication technology. RFID systems consist of many RFID tags, one or more tag readers, antenna, information management software (processing center) and database. The RFID tag (sender) represents a label or a chip attached to provide the identity of the objects, and communicates with the RFID reader via radio-frequency electromagnetic fields. Tags may contain different data, but the most commonly used data model for IoT is the EPC, which is a globally unique identifier (UID) for the objects (even people or animals). Those UIDs ensure that objects can be tracked using RFID tags. RFID tags can be active, passive or semi-passive/active. Active tags are powered by the battery, whereas passive ones do not have their own battery, but rather use the energy of electromagnetic radiation emitted by the tag reader. Semi-passive/active tags



Source: Sarkar *et al*. (2014)

Figure 6.
DIAT architecture for the IoT

have their own battery in addition to the energy waves emitted by the tag reader. The RFID reader (receiver) sends a query signal to the tag. This query signal looks for the possible existence of tags in the surrounding area. The reader then receives a response signal from the tag, which is then transferred to the database. The database is connected to the processing center to identify the objects based on the response signals within the range of 10 cm to 200 m. The antenna is used for transmitting data between the tags and the tags readers. RFID systems can be used in monitoring objects in real time, without having to exist on the line of sight, which allows mapping between the real world and the virtual world. RFID is not a new technology designed especially for IoT. RFID utility of tracking objects was established and applied to the areas of supply chain management, logistics, retailing, aviation, public utilities, security, food safety and e-health. It is generally understood that RFID tracking capability is a precursor of IoT. RFID benefits can be extended by making the data accessible remotely through the internet. The use of RFID has been delegated by organizations such as Walmart and the US Department of Defense (Al-Fuqaha *et al.*, 2015; Asghar *et al.*, 2015; Atzori *et al.*, 2010; Whitmore *et al.*, 2015).

5.1.2 Near-field communication. NFC is a short-range communication standard. Using NFC, devices can communicate wirelessly with each other when touched or placed near each other. It is a new technology that is built on the RFID standard and is often integrated into smartphones to share data when brought together. NFC devices are able to make connections with passive unpowered NFC tags that contain a UID tied to each object and associated with the tag. The most common use of NFC is in the contactless payment system, which is similar to those currently used in electronic ticket, smart card, credit card and smart posters. NFC is also used in Android Beam while transferring the file using Bluetooth (Porkodi and Bhuvaneswari, 2014; Whitmore *et al.*, 2015).

5.1.3 Wireless sensor network. WSNs will take an important part in IoT technology. WSNs can collaborate with RFID systems to better track the state of things, their location, movement, temperature, etc. WSNs also can increase awareness of a particular environment and act as an additional bridge between the physical and the virtual world. Moreover, WSNs have been introduced in many application scenarios such as e-health, environmental monitoring, intelligent transportation systems (ITS) and monitoring of industrial and

military systems. WSNs consist of a very huge number of sensing nodes that are connected to each other via multi-hop wireless technology. The sensor nodes usually report the results of their sensing to special nodes called sinks, which are small in number or as in most cases only one. Whereas sensors sense the state of the object or environment, actuators perform actions that affect the object or environment. Actuators can affect the environment by producing light, sound, radio waves or even smell. Because of these capabilities, IoT objects can communicate with people. This combination of sensors and actuators has produced sensors/actuators networks (SANET), which can enable objects to be simultaneously aware of their environment and to interact with people, which are targets for IoT SANET. A network that uses the sensors to detect the presence of carbon monoxide in the room and the actuators to produce high noise that alarms people of harmful gas existence is an example of this combination (Atzori *et al.*, 2010; Whitmore *et al.*, 2015).

5.1.4 Integrating WSN and RFID. To provide a full picture of the location and status of moving objects, which empowers IoT implementation in industrial services, WSN and RFID can be integrated. This integration would increase the deployment of services in other IoT applications such as healthcare and smart systems (smart rehabilitation, smart city or smart transportation) and would help in making decisions in complex systems. RFID sensor tag differs from a normal RFID tag, as sensor-enabled RFID can take the actions according to the data collected by the sensor in addition to the tracking and monitoring functions as in normal RFID tags (Li *et al.*, 2015; Porkodi and Bhuvaneswari, 2014). Table I makes a comparison between the features of RFID, WSN and radio-frequency identification sensor networks (RSN). Currently, many solutions have been proposed in this field such as wireless identification and sensing platforms project, which is implemented at Intel's Labs (Atzori *et al.*, 2010).

Another research is performed by Wang, Da Xu, Bi and Xu (2014). In their work, they focused on the adoption of RFID in WSN and the development of a new algorithm for data cleaning to effectively eliminate data redundancy. The researchers discussed some essential challenges related to the integration of WSN and RFID like power consumption, redundant data and time delay. The researchers developed a five-layer system architecture to integrate WSN and RFID systems. The Bluetooth and ZigBee technologies are chosen as a communication protocol for WSN networks to meet the requirements of a large number of sensor nodes, wide areas and low cost. The researchers also presented an improved cross-redundant algorithm (ICRDC) to eliminate redundant data in the integrated WSN. The effectiveness of the ICRDC algorithm was verified by simulation and comparison. Although the feasibility of RFID and WSN integration has been verified in this study, much work is needed to address the challenges of integrating WSN and RFID before making the system applicable to more challenging situations and being commercialized.

Characteristic	RFID	WSN	RSN
Standard	ISO18000	IEEE 802.15.4	None
Size	Very small	Small	Small
Power	Harvested	Battery	Harvested
Lifetime	Indefinite	< 3 years	Indefinite
Range (m)	10	100	Harvested
Sensing	No	Yes	Yes
Communication	Asymmetric	Peer to peer	Asymmetric
Processing	No	Yes	Yes

Source: Atzori *et al.* (2010)

Table I.
Comparison of RFID systems, wireless sensor networks and RFID sensor networks

5.2 Software

Even if IoT may rely heavily on hardware infrastructure that already existed, new software should be written to support interoperability among many heterogeneous devices and to process the huge volume of generated data (Whitmore *et al.*, 2015).

5.2.1 Middleware. To provide a solution to some issues, for example, interoperability, heterogeneity, dependability and security, which are frequently encountered in IoT, a middleware layer is employed. A middleware is a software layer that exists between the application and the network OS and provides abstraction between the applications and the IT infrastructure to hide the technological details. Therefore, application developers will focus more on the development of IoT applications (Chaqfeh and Mohamed, 2012; Perera *et al.*, 2014). IoT will include a huge number of heterogeneous devices that generate a vast amount of diverse data. IoT middleware would help in aggregating those diverse devices and huge data in a way that will enable the developers to create and deploy new IoT services without having to write different codes for each type of devices or data format (Whitmore *et al.*, 2015). The middleware software has become increasingly important in recent years because of its central role in facilitating the development of new services and the integration of old technologies into new ones. This simplification frees the programmer from the burden of exact knowledge of the various sets of techniques adopted by the lower layers. Hiding the details of different technologies is essential to make the programmer free from considering issues not related directly to his/her focus (Atzori *et al.*, 2010).

A middleware layer provides common application services and facilitates application development by integrating heterogeneous computing, communication and devices and supporting interoperability within various applications and services. Several solutions have been proposed and implemented in middleware research domains. Nevertheless, those proposals are mostly introduced for WSNs without considering RFID, M2M communications and Supervisory Control and Data Acquisition (SCADA), which are core elements in the IoT vision (Razzaque *et al.*, 2016). In spite of the considerable research efforts in the field of IoT and intelligent environments, no middleware has been standardized yet. Probably no single solution can be adapted for all environments (Chaqfeh and Mohamed, 2012).

The proposed middleware solutions vary widely in their design methods (event-based and database-based), programming abstractions level (local or node level, global or network level) and implementation domains (WSNs, RFID, M2M and SCADA) (Razzaque *et al.*, 2016). The SOA-based solution deemed to be the most promising one. In SOA-based middleware, the functionalities of each device would be offered as standard services with performing the discovery and invocation of new functionalities from other services synchronously. The integration of SOA-based middleware has the features of reducing the effort and cost required to recognize new business scenarios and eliminating the need for hardware drivers or third-party solutions (Chaqfeh and Mohamed, 2012).

Most of the proposed middleware solutions use the SOA approach, but they are not the only provided solution. Other solutions, developed especially for a specific scenario, are also introduced. One brilliant project is Fosstrak (Fosstrak.github, 2018), which focuses particularly on the management of RFID-based applications and implements the interfaces specified in the EPC specification. It is an open-source RFID infrastructure that provides services related to RFID management such as data collection, data filtering, data interpretation, data dissemination, tag writing, tag identifier management, privacy and fault and configuration management. These functions are provided to the application layer to facilitate the deployment of RFID-related services. Another solution, introduced in the e-SENSE project, also does not follow the SOA approach and focuses on issues related to capturing ambient intelligence via WSNs (Atzori *et al.*, 2010).

The researchers have suggested different classifications for middleware solutions depending on the core concept, requirements and design approach. Azzarà *et al.* (2013) first, introduced an overview of state-of-the-art middleware solutions that were targeting WSNs. Then, they introduced the network protocols, targeting the IoT scenarios, for next-generation middleware. They classified the IoT middleware based on their core concepts and design patterns into publish-oriented/subscribe-oriented design, service-oriented design, database-oriented design and virtual machine-oriented design. In addition, the overall architecture of the ICSI WSN middleware was presented by exploring its core components and IoT enabling solutions.

Chaqfeh and Mohamed (2012) provided a classification of middleware solutions according to the perspective scopes and IoT requirements. The classification includes three categories: RFID and sensor networks, robotics systems and semantic web and web services. The authors also reviewed the technical challenges that face the developers of designing IoT middleware solutions. A list of challenges on the surveyed approaches is highlighted to illustrate how each scope can support middleware solutions on the IoT.

Another classification has been introduced by Ngu *et al.* (2017). The authors classified the existing IoT middleware architectures into three classes: service-based, cloud-based and actor-based middlewares. The service-based or service-oriented architecture (SOA) middleware allows developers or users to add or deploy a variety of IoT devices as services. The second type, the cloud-based solution, enables users to connect, collect and interpret the collected data easily. The potential use cases can be identified and programmed in advance, but cloud-based solution limits the users to the type and number of IoT devices they can deploy. The third type, the actor-based framework, focuses on the open plug and play IoT architecture; therefore, a wide range of IoT devices can be viewed as reusable actors and distributed in the network. The authors also conducted a comprehensive analysis of the challenges and the enabling technologies in the development of IoT middleware such as the heterogeneity of IoT devices and the main requirements of adaptability, composition and security issues.

Razzaque *et al.* (2016) grouped the existing middleware solutions based on their design approaches into seven types: event-based, service-oriented, VM-based, agent-based, tuple spaces, database-oriented and application-specific design. The authors presented a comprehensive view of the field by comparing different middleware solutions based on IoT requirements like scalability, security, availability, reliability, real-time support and context awareness.

Some middlewares use a combination of different design approaches, for example, many SOA middleware solutions, such as Servilla and SOCRADES, also employ the virtual machine approach in their design and development. The ICSI project, which was created by the European Commission, adopted ICSI WSN middleware solution. ICSI WSN introduced state-of-the-art advances by integrating two middleware designs: publish/subscribe model and a virtual machine-based model. Such a novel architecture gives greater flexibility to the middleware, which makes it suitable for a wide range of applications. Typically, a hybrid approach outperforms the individual approach by taking advantage of multiple approaches (Azzarà *et al.*, 2013; Razzaque *et al.*, 2016). Table II shows the classifications and the examples of middleware on the aforementioned papers.

5.2.2 Internet of Things operating system. IoT devices are small in size, massive in number, have less memory and use less power. They come in two types: high-end devices and low-end devices. High-end devices, such as smartphones and Raspberry Pi, can run traditional OSs like Linux because they have more processing power and energy. It has been mentioned that more than 70 percent of IoT devices are using Linux, the dominant OS in the market for IoT gateway devices. However, low-end devices are very resource constrained; hence, they cannot run traditional OSs. Therefore, there is a need for a *de facto* standard OS specific for resource-constrained IoT devices to support the various applications and

Reference	Classifications	Examples
Azzarà *et al.* (2013)	Publish/Subscribe-oriented design	Mires, MQTT-S, LooCI
	Service-oriented design	DPWS, SensorsMW, Linksmart, SENSEI
	Database-oriented design	TinyDB
	Virtual machine-based design	VITRO
Chaqfeh and Mohamed (2012)	Semantic web and web services	SOA approach, UBIWARE, Triple space based, Task Computing Framework
	RFID and sensor networks	GSN, Fosstrak, TinyREST
	Robotics systems	Play/Stage, Robotic Operating System (ROS)
Ngu *et al.* (2017)	Service-based IoT middleware	Hydra, Global Sensor Networks (GSN)
	Cloud-based IoT middleware	Google Fit, Xively, Paraimpu
	Actor-based IoT middleware	Calvin, Node-RED, Ptolemy Accessor Host
Razzaque *et al.* (2016)	Event based	Hermes, EMMA, GREEN, RUNES, PRISMA, SensorBus, Mires
	Service oriented	Hydra, SenseWrap, MUSIC, TinySOA, SOCRADES, SensorsMW, SENSEI, ubiSOAP, Servilla, KASOM, CHOReOS, MOSDEN, Xively, CarrIoTs, Echelon
	VM based	Maté, VM, Melete, MagnetOS, SwissQM, TinyReef, DVM
	Agent based	Impala, Smart messages, ActorNet, Agilla, Ubiware, UbiROAD, AFME, MAPS, MASPOT, TinyMAPS
	Tuple space	LIME, TinyLIME, TeenyLIME, TS-Mid, A3-TAG
	Database oriented	SINA, COUGAR, IrisNet, Sensation, TinyDB, GSN, KSpot+, HyCache
	Application specific	AutoSec, Adaptive Middleware, MiLAN, TinyCubus, MidFusion

Table II. Middleware classifications and examples

operational requirements over a heterogeneous network (HetNet). Large-scale IoT software requires an appropriate OS to be built upon for better development, deployment and maintenance (Hahm *et al.*, 2016; Musaddiq *et al.*, 2018; Sabri *et al.*, 2017).

IoT devices usually operate with batteries and have a memory of 100 KB. They are equipped with 8-bit microcontrollers, actually very less than what the current Windows/Unix/Mac-based desktops and laptops have. The limitations of IoT devices call for the need of an efficient, lightweight, portable and flexible system with low RAM and ROM footprints. Windows 8.1, Linux, Arrayant, ARM, IFTTT and others are in the race of designing IoT OSs (Gaur and Tahiliani, 2015).

According to the VDC research report, it was expected that the market of IoT and embedded OS will grow from less than $1.5bn in 2014 to $1.7bn in 2019 at an annual growth of 2.9 percent (CAGR). Major vendors are concentrating more on the real-time operating system (RTOS) and fully functional operating system (FFOS) driven by small footprint OSs. With RTOS and FFOS markets in parallel, the embedded OS industry has been relatively stable over the past few years. Currently, due to Linux maturity and the need to reassess value creation and revenue generation, IoT OS market is experiencing a fast paradigm shift (Sabri *et al.*, 2017).

IoT OSs come in two types: open source and closed source. The open-source OSs include Contiki, RIOT, FreeRTOS, TinyOS, OpenWSNnuttX, eCos, mbedOS, L4microkerenel family, uClinux, Android and Brillo and others. The closed source OSs include ThreadX, QNX, VxWorks, Wind River Rocket, PikeOS, embOS, Nucleus RTOS, Sciopta, LiteOS Hawei and others (Hahm *et al.*, 2016). No single IoT OS deemed to be the dominant one because IoT is a new field of specialization even if the idea is old. IoT specialists are few and most researchers, developers and even companies are in the learning stage (Sabri *et al.*, 2017). Some of the most recently used OS will be provided in the next subsections.

TinyOS: TinyOS is an open-source, BSD-licensed OS, primarily developed for low-power wireless devices like the ones used in WSNs, UbiComp and PANs. TinyOS is designed for extremely constrained 8-bit and 16-bit platforms. TinyOS is considered the most powerful, energy-efficient and innovative OS. It has more flexibility in supporting different types of sensor applications. TinyOS features make it the preferred OS for WSNs (Amjad *et al.*, 2016; Hicham *et al.*, 2017).

TinyOS is written in NesC, a dialect of C. TinyOS was initially developed in 2000 at the University of California, Berkeley. The research project was started and used only by researchers. Currently, TinyOS development has been converted to GitHub, where the researchers can contribute to its development. Now, there are about 35,000 downloads for this free OS per year. TinyOS has made its way in well-known computing projects such as Cisco and Xen smart grid systems. Since implementing new applications in distributed sensor nodes using real hardware is more expensive and much more time consuming, there is a need for virtual environments for execution. Researchers can now simulate different sensing applications and OSs using virtual environments like TOSSIM (basic TinyOS simulator), Power TOSSIM, Viptos, Qualnet and TOSSF (Amjad *et al.*, 2016).

TinyOS uses a single-level file system to run a single file per node at any given time. However, a single-level file system cannot handle a large number of files at one time. TinyOS 1.x utilizes a microfile system called Matchbox, which provides an interface for reading, writing, deleting and renaming files. There are also other file system implementations from third parties like TinyOS FAT16, which supports SD cards in order to reduce the overall power consumption for sensor nodes. The FAT file system has a portable implementation that permits a node to store a huge volume of data (Musaddiq *et al.*, 2018). TinyOS also implements an efficient log-structured flash (ELF) file system that provides low-power operation, better memory efficiency and dedicated support for popular types of sensor files (Dai *et al.*, 2004).

Contiki operating system: Contiki OS is a flexible, lightweight and highly portable OS. It is one of the most widely used open-source OSs. Contiki OS was developed by Adam Dunkels in 2003 at the Swedish Institute of Computer Science. Contiki OS is written in C and is available on GitHub under the BSD license. It was initially developed for resource-constrained embedded systems around the uIP stack (Dunkels *et al.*, 2004; Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Hicham *et al.*, 2017). Contiki OS has evolved over time to be more general OS used in many areas like WSNs, IoT and retro computing. The typical Contiki OS configuration consumes 40 or 60 KB of ROM and 2 KB of RAM. Contiki supports a wide range of limited resources hardware including 8-bit AVR MCU platforms, 16- and 20-bit MSP430 MCU platforms and 32-bit ARM Cortex M3 MCU platforms (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Hicham *et al.*, 2017). Contiki OS also supports the ESB platform that uses the MSP430 microcontroller with 2 KB of RAM and 60 KB of ROM running at 1 MHz. The microcontroller has the ability to selectively reprogram parts of the on-chip flash memory (Amjad *et al.*, 2016; Dunkels *et al.*, 2004; Sabri *et al.*, 2017).

Contiki OS uses Coffee Contiki File System (CFS) (Tsiftes *et al.*, 2009), which is a virtual file system that provides an interface to different file systems. The Coffee file system supports the file system of sensor devices based on flash (EEPROM). Each file system uses CFS API to read, write and extract files in a mechanism similar to the API of Portable Operating System Interface for Unix (POSIX). On Coffee CFS, file size must be reserved in advance, which introduces a delay in real-time applications if they require more than the reserved size (Musaddiq *et al.*, 2018).

Contiki supports C language mainly, but some parts of Contiki OS like protothreads utilize macro-based abstraction, which requires the developers to take into account certain

limitations on the type of language features they can use. There are other run-time environments that enable development in languages such as Java and Python (Hahm *et al.*, 2016). Contiki OS and its various applications can be simulated on Cooja and MSPSim Netsim simulators (Amjad *et al.*, 2016).

FreeRTOS: At first, FreeRTOS was developed by Richard Barry in 2002. Now, it is maintained and distributed by Real-Time Engineers Ltd. FreeRTOS kernel is a market-leading RTOS. The main feature of FreeRTOS is the provision of standard solutions for microcontrollers and small microprocessors. FreeRTOS is deployed in various industrial/commercial environments, and it becomes the basis of many research projects. Unlike many other RTOS systems, FreeRTOS is simple, small, portable and easy to use. Thus, FreeRTOS has been supported by a large community and moved to a big number of MCUs including hardware available on open testbeds like IoT-LAB. FreeRTOS supports two privilege modes of operation: kernel mode and user mode. For the kernel to be protected from failures, FreeRTOS has FreeRTOS memory-protection unit (MPU). FreeRTOS is written in C programming language and is available under a modified GPL license, on which only the kernel has to remain open source and the commercial applications can be made closed source. There are several forms of FreeRTOS code base, for example, SafeRTOS, which focuses on safety, and OpenRTOS, which removes all references to GPL. Since its development, FreeRTOS has become one of the most widely used open-source RTOS for constrained devices (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Hicham *et al.*, 2017).

C is the programming language used for FreeRTOS, which enables the users to integrate any C++ application seamlessly. FreeRTOS does not define any portable driver model or MCU peripheral abstraction interfaces. Instead, FreeRTOS works with the provided vendor board support packages. Although testing and debugging the system rely on third-party solutions, the design of FreeRTOS makes it possible to integrate the testing and debugging into the existing development processes (Hahm *et al.*, 2016).

FreeRTOS utilizes the Super Lean FAT file system (FreeRTOS+FAT SL) that provides a low memory footprint. FreeRTOS+FAT12/FAT16/FAT32 is a DOS/Windows-compatible embedded file system with the main target of reducing flash footprint (< 4 KB) and RAM footprint (< 1 KB). Nevertheless, the FAT file system is unsuitable for IoT particularly because it was not designed to be used in flash memory (Musaddiq *et al.*, 2018).

RIOT: RIOT is an open-source lightweight OS designed for devices with low power consumption. The core of RIOT is largely inherited from FireKernel, which was originally developed for sensor networks. RIOT is free software released under the GNU License (LGPL). RIOT was developed primarily by the Free University of Berlin, the National Institute for Research in Computer Science and Automation (INRIA) and the Applied Sciences University of Hamburg (HAW Hamburg) (Hicham *et al.*, 2017). It is developed into two separate parts: hardware-dependent code and hardware-independent code with well-defined interfaces. With this design, the hardware-specific code would be configured independently to utilize the underlying platform without changing the kernel libraries (Baccelli *et al.*, 2018; Hahm *et al.*, 2016). RIOT has an efficient cross-platform code and supports a wide range of IoT hardware (8, 16 and 32-bit MCUs) and the development and maintenance of several network stacks (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Hicham *et al.*, 2017).

RIOT supports a friendly API for developers because RIOT microkernel architecture is written entirely in American National Standards Institute (ANSI) C language and supports multithreading. POSIX compliance is already partially available and, in the near future, it is planned for full POSIX compliance. Since RIOT is written entirely in C language, libraries and applications can be implemented in C++ language. RIOT utilizes the GNU Compiler Collection (GCC) in the latest versions, and its source code is available on GitHub

under LGPLv2. Contiki Cooja can be used in simulating some platforms (Hahm *et al.*, 2016; Hicham *et al.*, 2017).

To differentiate IoT OSs, there are also some key characteristics, concepts or requirements, which are described in the next subsections.

Architecture: The architecture of IoT OS kernel can be monolithic, modular microkernel or layered. Monolithic kernel has less expensive modules interaction and smaller memory footprint. The performance in monolithic is high because the control between the kernel and the userspace need not be passed as in the case of a microkernel. However, the monolithic kernel has a problem, which makes it a poor choice for the OS of low-end IoT devices. When the kernel code becomes long and complex, it becomes hard to be understood, modified, or configured. In microkernel architecture, the kernel size is significantly reduced because the kernel provides only the minimal functions, and all the other functions are provided by servers running at the user level. Hence, the OS functionalities can be extended and the failure in one user-level server will not crash the kernel itself. Modules in microkernel can be loaded into memory only when needed and can be configured individually. A microkernel architecture is also robust against bugs in the components. Due to the small size of the kernel and the few numbers of context switches, the microkernel is the preferred design option for many embedded OSs. Nevertheless, the microkernel is poor in performance because of the user-to-kernel boundary crossing and poor memory performance in comparison to monolithic. The third approach is the layered architecture. Layered architecture is more reliable, more manageable and less complex than a monolithic kernel, but it is less modular than microkernel (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Musaddiq *et al.*, 2018; Sabri *et al.*, 2017).

TinyOS follows a monolithic component-based architecture with no userspace. The components, or software modules, wrap around hardware and stick together to form a statically linked core. Each component represents a service, and the unused services (components) can be excluded from the application, which will reduce the size of the code. This architecture is efficient for memory constraints sensor networks (Gaur and Tahiliani, 2015; Hicham *et al.*, 2017). Similar to TinyOS, Contiki OS follows monolithic modular architecture. Applications modules are autonomous and can be linked to the kernel at boot time. The kernel provides core services only while the other services can be added if needed; therefore, memory footprint would be reduced and boot time would be decreased. At run-time, all processes share the same memory space and privileges with the core system. However, the kernel may crash due to the modules that contain bugs (Dunkels *et al.*, 2004; Hahm *et al.*, 2016; Musaddiq *et al.*, 2018). On the contrary, both FreeRTOS and RIOT follow a microkernel architecture (Gaur and Tahiliani, 2015).

Scheduling and real-time support: Another critical part of any OS is the scheduler. Scheduler affects other important properties that depend on the scheduling algorithm such as real-time capabilities, programming model, energy efficiency, system performance and response time. The schedulers must support multitasking and real-time capabilities, and they should be energy efficient. There are two types of schedulers: preemptive and cooperative. In the preemptive scheduler, when the CPU time is assigned to any task, the other tasks must put themselves in the cooperative mode. A preemptive scheduler works well in RTOS. To support real-time capabilities, the scheduler must respect strict time limits, that is the tasks should be accomplished within the specified time limits (Gaur and Tahiliani, 2015; Sabri *et al.*, 2017).

In TinyOS, the earlier form of task scheduler uses first in, first out (FIFO) preemptive scheduling to schedule threads with high priority execution. Since there is a wide range of tasks that must be handled by the OS, the FIFO policy is not valid for real-time tasks. Therefore, multiple scheduling techniques have been developed such as real-time scheduling

(RTS), earliest deadline first scheduling, priority-based soft RTS, scheduling policy, adaptive double-ring scheduling and co-routine scheduling (Amjad *et al.*, 2016; Gaur and Tahiliani, 2015; Musaddiq *et al.*, 2018). Contiki OS is based on a cooperative scheduling approach and FIFO fashion for events queuing and processing (Dunkels *et al.*, 2004; Hahm *et al.*, 2016).

FreeRTOS has preemptive multitasking scheduler and it supports real-time capabilities for low-end IoT devices. The execution of FreeRTOS follows a priority-based round robin, which is triggered by a periodic timer tick interrupt. Thus, FreeRTOS guarantees the execution of a higher priority task in any given period of time by dividing the execution time between the tasks if they are given the same priority (Musaddiq *et al.*, 2018). Since version 7.3.0, released on October 31, 2012, the scheduler also supports a tickless mode. In order to meet real-time guarantees, it is ensured that FreeRTOS uses only unavoidable operations from a critical section or interrupt. FreeRTOS supports blocking and nonblocking insert and remove functions using deep copy queues, and it uses queues for inter-process communication (IPC) (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Musaddiq *et al.*, 2018).

The RIOT kernel uses a fixed priority preemptive scheduler with O (1) operations, allowing for soft real-time capabilities. RIOT supports the tickless scheduler and the scheduling policy used in RIOT facilitates RTS. If the event requires an action by a high priority thread, the high priority thread runs until the event is handled and the lower priority threads are preempted (Baccelli *et al.*, 2018).

Programming model: The programming model determines how an application developer can model the program. Parallelism, memory hierarchy and concurrency are factors that influence the choice of the programming model. In IoT OSs, the programming models are divided into two types: event-driven and multithreaded models. In the event-driven programming model, every task has to be triggered by an external event such as an interrupt. This programming model is mostly accompanied by a shared-stack model and a simple event loop scheduler. Nevertheless, if multiple events occur, the event-driven concurrency model introduces certain complexities. The task in an event-driven model cannot be blocked during run-time and, at the same time, some time-critical tasks need to be executed first. Thus, the OS needs multiple event handlers for better work. Real-time performance is poor on the event-driven approach, whereas in the multithreaded programming model, each task would be able to run in its own thread context and the communication between the tasks would be enabled by using an IPC APIs (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Sabri *et al.*, 2017).

TinyOS follows an event-driven concurrency model because it uses NesC for development in the synchronized environment, which is a component-based and event-based programming language (Gaur and Tahiliani, 2015). Since building applications in TinyOS relied on an event-based programming model, previous versions of TinyOS do not provide multithreading support (Hicham *et al.*, 2017). In version 2.1 of TinyOS, partial support of multithreading is provided via TinyOS Thread (TOSThread), which uses a cooperative threading approach. Even if TOSThread provides a preemptive behavior to the TinyOS, it increases the computational complexity. To overcome this drawback on TOSThread and provide a preemptive implementation in an easy way, TinyOS preemptive original (TOS-PRO) approach was introduced. TOS-PRO provides an increase of flexibility for scheduling without introducing additional complexity in TinyOS (Hicham *et al.*, 2017; Lindgren *et al.*, 2012; Sabri *et al.*, 2017).

Contiki OS follows an event-driven programming model with lightweight pseudo-threading. It requires the Contiki process to explicitly return control back to the scheduler (Dunkels *et al.*, 2004; Hahm *et al.*, 2016). In the event-driven model, the event handlers continuously wait for internal or external events. The memory stack is allocated to the process by the kernel, and the run-to-completion mechanism is followed by the

event handler. All the processes share the same stack effectively and utilize limited memory efficiently (Hahm *et al.*, 2016). Contiki OS supports a novel, hybrid, lightweight and stackless threading mechanism, called protothread, which supports both event-driven and multithreading models. Protothread also takes advantage of the multithreaded model without increasing multiple stack overhead. Protothread provides concurrency and avoids CPU monopolization with preemptive multithreading. Protothread also facilitates an event-based model by utilizing a conditional locking wait statement that enables the program to execute a blocking wait without the need for an additional stack for every protothread. Nevertheless, the provision of synchronization between protothreads is not possible (Hahm *et al.*, 2016; Musaddiq *et al.*, 2018).

FreeRTOS follows a multithreading-based approach, in which the process can be interrupted and the scheduler can switch between threads (Musaddiq *et al.*, 2018). The RIOT programming model also supports multithreading that is partially compatible with POSIX, in which RIOT follows a concept of classical multithreading with an IPC memory passing between threads. RIOT provides the basic functionality of multithreading such as context switching, scheduling, IPC and synchronization primitives like mutex. All other components, such as device drivers, network stack components or application logic, are kept separate from the kernel. Typically, the interaction between these components is carried out through the minimalistic core API provided by the kernel (Baccelli *et al.*, 2018; Hicham *et al.*, 2017; Necula *et al.*, 2005).

Memory management: IoT devices are more resource constrained than other connected devices, especially in memory which holds both RAM and flash memory. IoT devices have a few kilobytes of memory, a million times less than standard machines like laptops and smartphones. Therefore, IoT OSs have to provide smaller memory footprint values that depend on the hardware platform architecture, compiler settings and most substantially OS configurations that include the size of both kernel and run-time libraries (Sabri *et al.*, 2017).

Memory management concerns with providing an abstraction to programming and it depends on the support of the underlying platform and application type. Memory management involves memory allocation/de-allocation, caching, virtual memory, logical–physical address mapping and memory protection. Many IoT OSs do not have a memory management unit (MMU) and floating-point unit because the small and simple kernel is the primary goal in IoT devices. Memory allocation can be static or dynamic. The static memory allocation is simpler than the dynamic memory allocation, but the flexibility of run-time memory allocation can be obtained with a dynamic approach (Gaur and Tahiliani, 2015).

In 2014, The Internet Engineering Task Force (IETF) in the RFC-7228, standardized a classification of these devices based on the memory capacity into three subcategories: Class 0, Class 1 and Class 2. Class 0 devices have low-level resources, with $<<10$ KB of RAM and $<<100$ KB flash. Class 1 devices have average-level resources, with _10 KB of RAM and _100 KB flash, which allow advanced features and more applications than rudimentary motes. Class 2 devices have more resources, but they are still more resource constrained in comparison to high-end IoT devices and conventional internet hosts (Hahm *et al.*, 2016; Sabri *et al.*, 2017).

Earlier versions of TinyOS support only compile-time static memory allocation due to the limited available space. TinyOS uses NesC language, which does not support dynamic memory allocation; therefore, program states and memory are defined at compile time, and run-time allocation failure and memory fragmentation are prevented. In the previous versions of TinyOS, memory safety was not available (Amjad *et al.*, 2016; Musaddiq *et al.*, 2018). However, incremental revisions and new updates in TinyOS provide improved features such as memory safety and memory safety checks like Safe TinyOS (Cooprider *et al.*, 2007), Untrusted Extensions (UTOS) (Regehr *et al.*, 2006) and CCured

(Necula *et al.*, 2005). Furthermore, dynamic memory-like capabilities in TinyOS can be introduced using a component called TinyAlloc via an interface called MemAlloc. Moreover, additional memory management capability is provided through a mechanism called TinyPaging, which makes use of flash storage to provide additional space (Amjad *et al.*, 2016; Musaddiq *et al.*, 2018). In TinyOS, there are no heap, function pointers or virtual memory concepts of dynamic memory allocation. In version 2.1 and beyond, safe TinyOS has a MPU. TinyOS devices belong to Class 0 according to the standard IETF classification (Gaur and Tahiliani, 2015; Hicham *et al.*, 2017).

Contiki OS is primarily designed for static allocation and it contains a few libraries that facilitate memory management like *memb and mmem*. Contiki OS uses third-party dynamic allocation modules, which implement the standard C malloc API. The Contiki C library provides functions for allocating and deallocating memory-like memb *macro(), memb_alloc()* and *memb_free()* functions, which are used for memory declaration, allocation and de-allocation, respectively. Nevertheless, the dynamic allocation may cause a stack overflow and require more space. There is no MPU in Contiki OS. Contiki OS devices belong to Class 0 and Class 1 of IETF standardized classification (Hahm *et al.*, 2016; Musaddiq *et al.*, 2018; Sabri *et al.*, 2017).

In FreeRTOS, the kernel allocates memory dynamically for every event. The *malloc ()* and *free ()* functions, which are used for dynamic allocation and de-allocation, respectively, are undesirable in the real-time OS because dynamic memory allocation has typically deterministic run-times, suffers from memory fragmentation and needs extra code space. To solve these problems, FreeRTOS introduced pvPortMalloc() and vPortFree() functions, which provide three heap implementations, Heap_1, Heap_2 and Heap_3, to allocate memory depending on the system design. FreeRTOS devices belong to Class 1 and Class 2 of IETF standardized classification (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Musaddiq *et al.*, 2018).

RIOT OS is modular based to be able to ensure minimum memory usage. The system configurations can be customized to meet the special specifications. The size of the kernel is minimized; thus, the kernel would require only a few hundred bytes of RAM. Modules dependencies are reduced to an absolute minimum (Hicham *et al.*, 2017). RIOT OS supports both static and dynamic memory allocation. Static memory allocation is used only within the kernel. Therefore, constant intervals for kernel tasks, such as scheduler run, IPC and timer operations, would be enforced to enable RIOT to achieve the unavoidable requirements. However, RIOT does not have a MMU or a floating-point unit. RIOT devices belong to Class 1 and Class 2 of IETF standardized classification (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016).

Networking: The main feature of IoT is that the devices can be connected to each other and to the internet; therefore, IoT devices usually have one or more network interfaces. IoT uses a wide range of low-power radio technologies such as Bluetooth Low Energy (BLE), Institute of Electrical and Electronics Engineers (IEEE) 802.15.4 (ZigBee), Z-wave, Sigfox, Neul and NB-IoT, in addition to different wired technologies like Ethernet. To allow seamless efficient communication through the internet on those different wired and wireless technologies, an open standard is needed. IoT stack should be lightweight, reliable, internet-enabled and flexible; hence, it can be adjusted with minimal changes to meet the requirements of a wide range of IoT applications. Supporting IPv6 is compulsory in IoT systems to make the object uniquely identified in this extremely huge network. Mechanisms such as Constrained Application Protocol (CoAP), 6LoWPAN, Low-power WPAN over IPV6, and RPL, IPv6 Routing Protocol for Low-power and Lossy Networks are designed for low-power systems. Header compression and the inclusion of minimum features are utilized to help in keeping the protocols applicable to IoT (Gaur and Tahiliani, 2015; Sabri *et al.*, 2017).

TinyOS uses Berkeley low-power internet stack (BLIP), which includes some protocols, such as IPv6, ICMPv6, TCP and UDP, and supports CoAP, 6LoWPAN and RPL. The OS uses the

Hydro routing protocol to ensure reliable unicast communication within IPv6 subnetwork with lossy links (Musaddiq *et al.*, 2018). However, in Contiki, there are two different network stacks that can be used: the popular microIP (uIP) stack and the more lightweight Rime stack. MicroIP (uIP) was first developed as a standalone stack and then merged into Contiki after version 0.9. uIP stack protocols are developed especially for wireless, low-power, memory-constrained and real-time sensor devices. uIP supports multiple protocols such as 6LoWPAN, IPv4, IPv6, IPv6 neighbor discovery, IPv6 multicasting, RPL, TCP and UDP. A separate module, called Queuebuf, is used for network buffer management by allocating packet buffers from a static pool of memory. Rime stack is essentially developed for sensor network applications. It includes a set of lightweight custom protocols and a set of distributed programming abstractions. Contiki OS also adopts another application layer protocol called Micro eXtensible Messaging and Presence Protocol (uXMPP), which is used in web services like SMS, publish–subscribe and basic authentication mechanisms (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016).

FreeRTOS uses network stacks provided by third party for internet connectivity because it does not have its own network stack. The leading third party is the Real Time Engineers Ltd, which offers an official FreeRTOS+TCP add-on. FreeRTOS+TCP provides an Ethernet-based IPv4 stack with support for TCP and UDP protocols. Another third-party networking stack is lwIP, which is port of third-party embedded network stacks and is based on IPv6 and other less power-consuming protocols like 6LoWPAN, CoAP, etc. FreeRTOS also supports FreeTCPIP, which is based on uIP stack. FreeTCPIP simplifies the TCP and UDP operations for low-end IoT devices; nonetheless, FreeTCPIP is still under development (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Musaddiq *et al.*, 2018).

RIOT has a modular internet stack. Modules can be maintained separately, which makes a modular stack more flexible than a layered stack. RPL and 6LoWPAN protocols are provided with the support of IPv6, TCP and UDP (Gaur and Tahiliani, 2015). The RIOT system has various network stacks including the gnrc stack, which is the RIOT implementation of full 6LoWPAN stack, OpenWSN, which is a port of the 6TiSCH stack, and CCN-lite, which is a port of the information-centric networking stack (Hahm *et al.*, 2016).

Power management: When designing an IoT OS, energy efficiency should be considered. Reducing power consumption is essential for IoT battery-powered devices as it increases product life and saves money. Unlike traditional network protocols, IoT protocols have to be designed with energy efficiency in mind. In wireless systems, the radio transceiver is the most power-consuming component; therefore, for the IoT low-power devices to be energy efficient, the radio is kept off as much as possible by using low-power protocols like LoRaWan, RPL and 6LoWPAN (Sabri *et al.*, 2017).

During sensing, processing and transmission, the IoT node experiences idle–busy time, which is too short to perform conventional context switching. TinyOS uses software thread integration (STI) for energy saving, by which the processor can utilize idle–busy time in accomplishing other useful tasks. Thus, the processor can enhance battery life by switching to lower power mode sooner (Musaddiq *et al.*, 2018). TinyOS also applied energy-aware target tracking (EATT) algorithm, which employees clustering and data aggregation techniques (Sarna and Zaveri, 2010). By using energy tracking optimization, the number of CPU cycles can be reduced. Nevertheless, this mechanism is unsuitable for mobile devices and it may introduce additional memory usage (Musaddiq *et al.*, 2018).

Contiki kernel itself does not provide any explicit power-saving mechanism; instead, it uses an application-specific energy conservation mechanism. The event scheduler discloses the size of the event queue to help the application determine the system shutdown time. Then, the application provides a power-saving mode by monitoring the event queue size. When the scheduled event queue is empty, the application can place the CPU in sleep mode until it is waked up by an interrupt (Dunkels *et al.*, 2004; Gaur and Tahiliani, 2015; Musaddiq *et al.*, 2018).

Contiki also has a default mechanism, called ContikiMAC protocol, which makes the devices run in low-power mode and still enables them to receive and transmit radio messages. ContikiMAC protocol uses an efficient wake-up mechanism, with which the idle radio duty cycle is only 0.6 percent because this mechanism has a wake-up frequency of 8 Hz (Sabri *et al.*, 2017).

In an event-driven system, tasks or threads will spend some of their time waiting for interruption or the expiration time. In FreeRTOS, this waiting state is called a blocked state. If all the tasks are in a blocked state, FreeRTOS creates and runs a task called idle task hook function. The idle task is given the lowest priority, and the idle hook function gets called only when there is no higher priority task available. The processor can go into power-saving mode when it is idle. This mechanism may be useful in some scenarios, but if the tick frequency is too high, the processor will waste energy and time getting in and out of the idle mode. Therefore, a tickless idle technique was introduced in order to provide a suitable mechanism for energy saving. Using tickless idle technique, FreeRTOS would be able to stop the periodic tick interrupt during idle intervals to allow the processor to remains in a deep sleep mode until either a higher priority external event or kernel interruption occurs. However, this technique introduces a run-time overhead (Gaur and Tahiliani, 2015; Musaddiq *et al.*, 2018; Sabri *et al.*, 2017).

To reduce power consumption, RIOT uses tickless scheduler that works without any periodic events. When there are no pending tasks, the system enters into the sleep mode by switching to the idle thread and remains in this mode until an external or kernel-generated interruption occurs (Baccelli *et al.*, 2018; Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Sabri *et al.*, 2017).

Hardware platform: The degree of heterogeneity in today's internet devices is not very high. On the contrary, a vast heterogeneity and variety of hardware and communication technologies have been developed in IoT because of the wide range of use cases and scenarios. IoT low-end devices are based on different microcontroller architectures and families, which include an 8-bit processor architecture like Intel 8,051 and Atmel AVR, a 16-bit processor architecture such as TI MSP430 and a 32-bit processor architecture such as ARM Cortex-M and MIPS32. A 64-bit architecture may also appear in the future. Therefore, a generic IoT OS should support the different microcontroller architectures and families (Sabri *et al.*, 2017).

TinyOS supports various sensors such as MICA, MICA2, MICAz, TelosB/TMote Sky, Intel® Mote2, eyes, tinynode, IRIS, shimmer, TI CC2430 (test), AVR, MSP430 and px27ax (Amjad *et al.*, 2016; Sabri *et al.*, 2017), whereas Contiki runs on TelosB/Tmote Sky, ARM7, ARM Cortex-M, PIC32,x86,6502 platforms (Amjad *et al.*, 2016; Dunkels *et al.*, 2004; Sabri *et al.*, 2017). The FreeRTOS is developed for Atmel AVR, MSP430, ARM, x86, 8,052, Cortus and Renesasc platforms (Hahm *et al.*, 2016; Sabri *et al.*, 2017), whereas RIOT is developed for AVR, MSP430, ARM7, ARM Cortex-M, x86 and TI MSP430 platforms (Gaur and Tahiliani, 2015; Hahm *et al.*, 2016; Hicham *et al.*, 2017).

The comparison between the four OSs, depending on the previously discussed criteria, is shown in Table III.

5.2.3 Storing and processing data

Big data: The huge volume of data produced from IoT ecosystem has played a prominent role in the big data area. Big data analytics would become a big challenge in IoT because a massive amount of data would be collected from different sensors in different places. The International Data Corporation report showed that the big data market will reach over $125bn by 2019. The benefit of big data analytics is to reveal trends, unseen patterns, hidden relationships and new information from the huge collected data, and consequently reveal knowledge, through which a business can achieve a competitive advantage. There are some platforms for big data analytics such as Apache Hadoop and SciDB. Since the amount of data in IoT is generally too huge, these tools are not strong enough for IoT needs. Furthermore, in IoT environment, the platforms must work in real time to serve the users more efficiently (Al-Fuqaha *et al.*, 2015; Marjani *et al.*, 2017; Xu *et al.*, 2014).

Criteria	TinyOS	Contiki OS	FreeRTOS	RIOT
License	Open-source BSD	Open-source BSD	Under a modified GPL	GNU General Public License (LGPLv2)
Architecture	Monolithic	Monolithic (modular)	Microkernel RTOS	Microkernel RTOS
Programming model	Event-driven, thread	Event driven, protothread	Multithreading	Multithreading
Scheduling	Cooperative	Cooperative	Preemptive, optional tickless	Preemptive
Real time	Not supported	Partially supported	Supported	Supported
Developing language	NesC	C	C	C, C++
Memory management	Static/Dynamic	Static/Dynamic	Dynamic	Static/Dynamic
Hardware platform	MICA, MICA2, MICAz, TelosB/TMote Sky, Intel-Mote2, eyes, tinynode, IRIS, shimmer, TI CC2430 (testing), AVR, MSP430, px27ax	MSP430, Atmel AVR, ESB, TelosB/Tmote Sky, ARM7, ARMCortex-M, PIC32, x86,6502	MSP430, AVR, ARM, x86,8052, Renesas	AVR, MSP430, ARM7, ARM Cortex-M, x86, TI MSP430
Network stacks	BLIP	microIP (uIP), Rime	Free TCP/IP, lwIP	gnrc stack, 6TiSCH stack
Network protocols	Zigbee, 6LoWPAN, RPL, CoAP, MQTT	Zigbee, BLE, 6LoWPAN, RPL, CoAP, MQTT	Zigbee, 6LoWPAN, RPL, CoAP, MQTT	Zigbee,182.15.4, BLE, 6LoWPAN, RPL, LPWAN, MQTT, CoAP
Communication key concept	TinyLPL, MultiMAC, TinyRPL, QURPL, HDRTP, STCP, ERTP, RCRT	ContikiMAC, X-MAC, CSMA-MAC ContikiRPL, RERBDI, BRPL, Contiki has uIP, uXMPP	FreeRTOS MAC, 6LoWPAN Nanostack	RPL
Simulators	TOSSIM (Basic TinyOS Simulator), Power TOSSIM, Viptos, Qualnet, TOSSF, etc.	Cooja, MSPSim Netsim	QEMU, Keil	Cooja
Energy consumption	STI, TOSSTI, EATT	No explicit power-saving mechanism, Microcontroller is put in sleep mode	Hook function, tickles idle	Tickless scheduler
Concurrency	Yes	Yes	Yes	Yes
File system	Single level (ELF, Matchbox)	Coffee	Super Lean FAT File System	ConstFS static, DevFS device, FAT, FatFs integration, littlefs, SPI flash file system
Multi-thread support	Partial (through tiny threads)	Yes	Yes	Yes
Targeted device class	Class 0	Class 0, Class 1	Class 1, Class 2	Class 1, Class 2

Table III.
IoT OSs

(*continued*)

Criteria	TinyOS	Contiki OS	FreeRTOS	RIOT
MCU	8-bit and 16-bit	8-bit, 16-bit, 20-bit and 32-bit MCUs	12-bit, 16-bit and 32-bit MCUs	8-bit, 16-bit, 32-bit MCUs
Security	TinySec	ContikiSec	FreeTEE (Trusted Execution Environment)	Libraries such as tweetNaCl, micro-ecc
Portability	Yes	Highly portable	Yes	Yes

Sources: Baccelli *et al.* (2018), Casado and Tsigas (2009), Gaur and Tahiliani (2015), Hahm *et al.* (2016), Hicham *et al.* (2017), Karlof *et al.* (2004), Musaddiq *et al.* (2018), Sabri *et al.* (2017), Riot-os (2018)

Table III.

IoT requires a common platform for big data analytics that can be delivered as a service to IoT applications instead of providing application-specific analytics. This analytic service also should not cause significant overhead on the entire IoT ecosystem. Some recent research studies have been proposed as IoT big data analytics services such as TSaaaS, which uses time series data analytics to perform pattern mining on a massive sensor collected data (Al-Fuqaha *et al.*, 2015; Xu *et al.*, 2014). The evaluation by some research studies showed that TSaaaS can achieve pattern mining faster than the existing systems. However, it was reported that 0.4 percent of the original data volume was needed for the index storage of the service provider, which introduced a space overhead. Keeping track of the interesting data only would be the applicable solution for IoT big data. Current approaches can assist in this case such as pattern reduction, dimensionality reduction, principal component analysis, feature selection and distributed computing methods (Al-Fuqaha *et al.*, 2015; Tsai *et al.*, 2014). More importantly, a centralized infrastructure is needed to support storage and analytics, which forms the IoT middleware layer. Cloud-based storage solutions have become increasingly popular, and the cloud-based analytics and visualization platforms will be widespread in the upcoming years (Gubbi *et al.*, 2013).

A novel meta-based model for integrating IoT architecture objects is proposed in Marjani *et al.* (2017). The main goal of this semi-automatic approach is to provide support for the appropriate decisions for complex businesses and architecture management with the development of IT evaluation systems. Figure 7 shows both IoT architecture and big data analytics. As shown in this figure, the sensor layer contains all the IoT objects and sensor devices that are connected to each other through a wireless network such as RFID, WiFi, ultra-wideband, ZigBee and Bluetooth. IoT gateway is used to allow IoT devices to be connected to various webs and to the internet. The upper layer contains big data analytics, the cloud, in which a massive amount of data received from IoT devices and sensors is stored and accessed through big data analytic applications. Big data analytic applications have API management and dashboard to help in the interaction with the processing engine.

Data mining: It seems impossible to connect everything on Earth together over the internet, but IoT will dramatically change our lives in the foreseeable future. For many, massive data created or captured by IoT are very useful and valuable information. Undoubtedly, data mining will play vital role in making this kind of system smart enough to provide services and more convenient environments. One of the most important questions that arises is the following: how can we convert data generated or captured by IoT into knowledge to provide a more suitable environment for people? This place is where data mining and knowledge discovery in databases technologies come into play (Tsai *et al.*, 2014). Machine learning and statistical methods are used in data mining techniques for both problem-specific and general data analytics. In IoT, the data are different from the normally

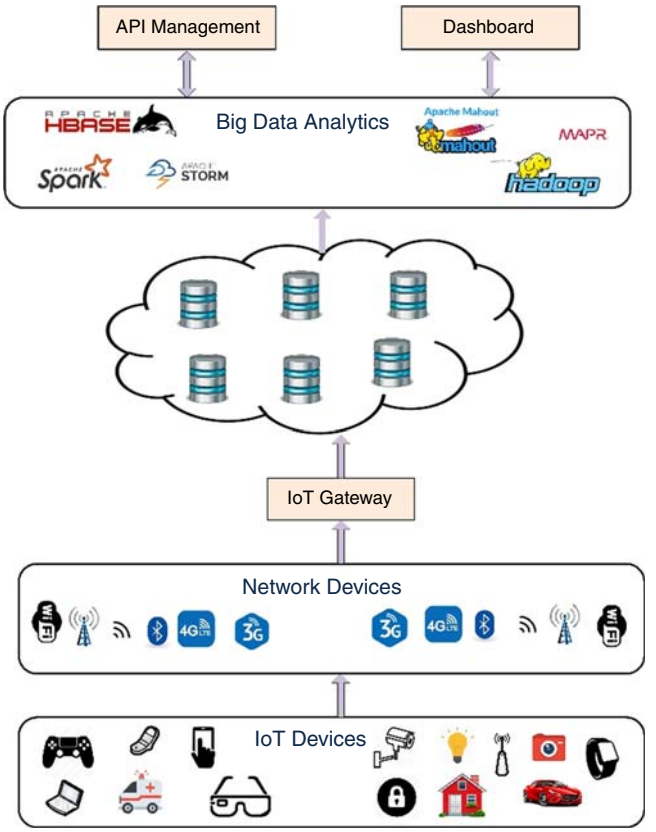


Figure 7.
IoT architecture and big data analytics

Source: Marjani *et al*. (2017)

collected big data because IoT big data are collected from various objects and sensors with heterogeneity, variety, noise and rapid growth (Marjani *et al.*, 2017). Much research is focusing on the use or the development of effective data mining techniques for IoT (Tsai *et al.*, 2014). The researchers in Bin *et al.*, 2010; Cantoni *et al.*, 2006; Keller, 2011; Masciari, 2007; Tsai *et al.*, 2014 assured that data mining algorithms can be used to make IoT smarter, thus providing more intelligent services.

Compressed sensing (CS): The desired compression ratio of data in IoT is very important, which cannot be obtained through the current ways without the introduction of unacceptable distortions. Moreover, for most data compression solutions in IoT, three major issues must be resolved: reliability, sensitivity and resolution. Lately, there is an extensive investigation on an emerging theory called CS, in which data or signals can be efficiently sampled and precisely reconstructed with much fewer samples than the Nyquist theory. CS is a low-cost data acquisition system. It is an important system for collecting and processing data at the end node of IoT-based information systems. The CS changes the game rule of data acquisition in information systems by utilizing the information of *a priori* data sparsity. Li *et al.* (2013) studied the use of CS in information acquisition in IoT and WSNs from the perspective of compressed data sampling, accurate reconstruction and robust transmission, aiming to increase the network capacity and reduce data redundancy,

energy consumption and computation costs. The common duty of IoT end node is to send perceived data to the determined node or fusion center. The researchers demonstrated that CS can be a powerful data acquisition tool to save energy and communication resources in information systems and networks.

Cloud computing: Lately, the already connected devices have arrived 9bn, which exceeded the number of people in the world. By 2020, those connected devices are expected to reach 24bn. Due to this increase in the connected devices number, the amount of data will be massive and huge. There is a crucial need of not only storing the data efficiently but also extracting the valuable knowledge, which requires a high processing ability that does not exist on the resource-constrained low-end IoT devices. Due to this, the processing and calculation need to be available on a rental basis (Aazam *et al.*, 2014). Cloud computing provides a new management mechanism for big data that enables the processing of data and the extraction of valuable knowledge. It also provides the virtual infrastructure that integrates storage devices, analytics tools, client delivery, monitoring devices and visualization platforms. The cost-based model of cloud computing will allow users to access applications on demand from anywhere and will support end-to-end service provision for users and businesses (Al-Fuqaha *et al.*, 2015; Gubbi *et al.*, 2013).

IoT and cloud computing need to be merged together on a novel IT paradigm (Botta *et al.*, 2014). The integration between IoT and cloud computing is not simple, and it has some major problems that need to be dealt with. This integration is referred to as the Cloud of Things (CoT) in some references. On one hand, the CoT will increase business opportunities because the cloud deals with business perspective as well as data and resources. On the other hand, it will increase the chance for the attackers, especially in hybrid clouds. Therefore, security and privacy, particularly identity protection, are very important. In addition, HetNet with different types of services and data will be involved in CoT; therefore, the network must be able to support these different types of data according to their needs for QoS support (Aazam *et al.*, 2014).

The cloud and IoT have evolved independently. In the future, it is expected that there will be many common advantages arising from their integration. Each technique will benefit from the features of the other one. On one side, the IoT can compensate its technological limitations (such as storage, processing and energy) by taking advantage of the virtually limitless capabilities and resources of the cloud. On the other side, the cloud can benefit from IoT by expanding its scope to deal with real-world things in a more dynamic and distributed way. Cloud would be able to deliver new services in a large number of real-life scenarios. Basically, the cloud acts as an intermediate layer between things and applications, hiding all the complexity and functionality needed to implement the applications. This framework will affect the development of applications in the future. Gathering, processing and transmission of information will introduce new challenges that must be addressed in a multi-cloud environment (Botta *et al.*, 2014). The integral characteristics of IoT and Cloud, which inspire the Cloud IoT paradigm, are shown in Table IV.

IoT	Cloud
Pervasive (things are placed everywhere)	Ubiquitous (resources are usable from everywhere)
IoT is real-world things	Cloud is virtual resources
Computational capabilities are limited	Computational capabilities are virtually unlimited
Storage capabilities is either limited or no storage	Storage capabilities are virtually unlimited
IoT is a source of big data	Cloud is intended to manage big data
Uses the internet as a point of convergence	Uses the internet for service delivery

Sources: Babu *et al.* (2015), Botta *et al.* (2014)

Table IV.
Characteristics of IoT and cloud

The Cloud IoT model provides new scenarios for smart services and applications based on cloud extension on IoT devices. Cloud IoT's new services are shown in Table V.

A number of IoT cloud providers are now in the market to take advantage of IoT-based services. Ray (2016) provided and summarized a detailed explanation about the existing IoT cloud service providers, such as KAA, Etherios, Xively, Exosite, Arkessa, Axeda, Oracle IoT cloud, IBM IoT, AerCloud and GroveStreams, in some application domains and their advantages and disadvantages as shown in Table VI.

Cloud computing will represent the backbone of the IoT; thus, QoS is used to differentiate between cloud service providers. QoS is very important because of the competitiveness and the openness of the cloud computing market. For the cloud service providers to succeed in a competitive market, they have to deliver superior services that meet customer expectations. QoS is used to represent, measure and compare the quality of cloud providers' services in order to make everything clear between cloud stakeholders. Nevertheless, measuring the quality of cloud computing services is difficult because they are intangible, unlike normal goods or products. Moreover, cloud computing services are provided via the internet – web based – with no direct interaction between the users and the cloud service providers. Inspired by SERVQUAL and the e-service quality model, Zheng *et al.* (2014a) proposed a quality model for cloud services, named CLOUDQUAL. SERVQUAL and the e-service quality models were unable to provide objective quality measurements; therefore, they cannot be applied to cloud services without reconstruction. To overcome their weaknesses, CLOUDQUAL was proposed. CLOUDQUAL model has six dimensions of quality and five quality metrics that target general cloud services. The six quality dimensions include availability, usability, reliability, flexibility, security and responsiveness. While the usability remains subjective and non-negotiable, the other dimensions are objective and negotiable. The quality metrics include the five objective dimensions. To demonstrate the effectiveness of CLOUDQUAL, the researchers conducted empirical case studies on three well-known storage clouds: Microsoft Windows Azure Blob Storage (Azure Blob), Amazon S3 and AliyunOpen Storage Service. Moreover, the researchers validated the model with three criteria: consistency, correlation and discriminative power. The results showed that CLOUDQUAL can assess and differentiate service quality.

In the same direction, the same authors of the previous paper introduced a mixed approach, based on the "game of chicken", for cloud service negotiation in Zheng *et al.* (2014b). Cloud customers and cloud providers have different QoS preferences that might cause a conflict. Moreover, the service-level agreement cannot be accessed without negotiation. The use of automated negotiation, in which agents negotiate on behalf of their

Table V.
CloudIoT's new services

CloudIoT's services	Acronym	Jobs
Sensing as a Service	SaaS	Provides ubiquitous access to sensor data
Sensing and Actuation as a Service	SAaaS	Allows implementation of automatic control logic in the cloud
Sensor Event as a Service	SEaaS	Sends messaging services triggered by sensor events
Data as a Service	DaaS	Provides ubiquitous access to any kind of data
Database as a Service	DBaaS	Enables ubiquitous database management
Ethernet as a Service	EaaS	Provides ubiquitous layer-2 connectivity to remote devices
Sensor as a Service	SenaaS	Enables ubiquitous management of remote sensors
Identity and Policy Management as a Service	IPMaaS	Enables ubiquitous access to policy and identity management functionalities
Video Surveillance as a Service	VSaaS	Provides ubiquitous access to recorded video

Sources: Babu *et al.* (2015), Botta *et al.* (2014)

Application domain	IoT cloud platforms	Advantages	Disadvantages
Application development	KAA	Supports NoSQL and big data applications	Supports less hardware modules
	Carriots	Supports triggering-based applications	Less user-friendly design
	Temboo	Supports Choreos-based applications	Not appropriate for intensive resources applications
Device management	SeeControl IoT	Supports push/pull-based devices	Visualization does not reach the desired level
	SensorCloud	Can manage large pool of sensor devices	Difficulty in serving open-source devices
	Etherios	Devices and third-party software specialized clouds are enabled	Restrict developers on selected devices
	Xively	Can integrate with devices easily	Few Notification services are provided
System management	Ayla's IoT cloud fabric	Facilitates mobile application development	Appropriate only for large-scale developers
	thethings.io	Device neutrality	Dependent on a third party Lacks of a self-sustainable
	Exosite	Facilitates system development	Lacking of Big data provision
Heterogeneity management	Arrayent Connect TM	Flexible to use	Lagging behind trigger-based services
	Open remote	Supports open cloud service	Too expensive for developers
Data management	Arkessa	Enabled enterprise's design side	not good enough visualization apps
	Axeda	M2M-based data management	Dependent on a third party Lacks of a self-sustainable
	Oracle IoT cloud	Supports database	Because of size Constraint, there is a lack of connectivity of open-source devices
	Nimbits	Developers adopt it smoothly	Slow processing of real-time query
	ThingWorx	Facilitates building data intensive application	Can attach limited number of devices
Analytics	InfoBright	Big data processing and knowledge grid architecture	Lacks of statistical services
	Jasper Control Center	Enables rule-based behavior patterns	Less appropriate for automation services
Deployment management	Echelon	Industrial perspective	Beginners' development scenario is not supported enough
Monitoring management	AerCloud	M2M scalable services	Not suitable for developers
	ThingSpeak	Enables public cloud with trigger facility	Less number of devices can be connected synchronously
Visualization	Plotly	Support for best IoT visualization tools	Storage facility amount is limited
	GroveStreams	Enables seamless event monitoring	Lacks of statistical services
Research	Microsoft research Lab of Things	Proper for home automation	Lacks of IoT supported APIs
	IBM IoT	Enables device identity	Application prototyping is difficult

Source: Ray (2016)

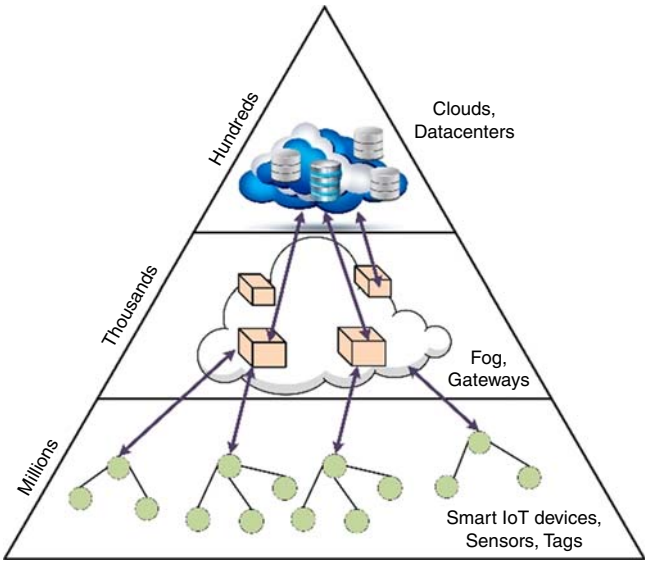
Table VI. Application domain-specific IoT clouds

human counterparts, has been studied for many years in electronic commerce and artificial intelligence (AI). It is considered the most elastic approach to obtain services and products. To create a proposal, two negotiating strategies, concession negotiating strategy and trade-off negotiating strategy, can be adopted by agents. Regarding the utility, a trade-off negotiation approach can overcome a concession one. However, if the information is

incomplete, which may cause miscalculation, a concession negotiation approach can overcome a trade-off approach regarding the success rate. To make a balance between the two approaches, the authors suggested a mixed approach based on the "game of chicken" for cloud service negotiation. In this approach, if the party's counterpart used a trade-off approach, it would be better to use a concession approach, and if the party's counterpart used a concession approach, it would be better to use a trade-off approach. If the party was unsure of its counterpart strategy, it would be better to adopt a mixed approach of trade-off and concession negotiation strategies. The suggested approach can balance between a utility and success rate and can outperform both approaches, as it performs a higher utility than a concession approach while having fewer failures than a trade-off approach. The authors conducted extensive simulations. They first tested the effect of different standards on the outcome of the negotiations and then performed the Monte Carlo simulation. The results showed the effectiveness of the mixed approach when the party has no knowledge of the counterpart's strategy. It should be noted that the mixed approach operates under incomplete information and therefore applies to real negotiations, where information is generally incomplete. Nevertheless, the functions of the nonlinear utilities adopted by them in simulations are general functions, which may not be able to represent specific preferences of agents. Even with this limitation, key findings and paper conclusions are not affected.

Fog computing: Over the past decades, there has been considerable interest in the IoT research, but most of the research has been focused on IoT system aspects such as networking, systems and middleware and cloud support for the IoT. Because of the substantial nature of IoT, which includes widely distributed and heterogeneous devices and computing resources and diverse perception–action cycles, application development is still a complex process. Virtualizing things using cloud services offers a well-defined device abstraction and a common programming framework to facilitate development. However, the greater impact of network failure and unnecessary latency would be introduced because of the centralized cloud-based middleware. Moreover, in many situations and IoT applications, pushing raw data directly into the cloud is not necessary always. Therefore, filtering, aggregating and analyzing raw data are needed to reduce storage and communication costs before uploading data to the cloud. Researchers have proposed fog computing to solve cloud computing shortcomings. Unlike cloud-based virtualization, the fog computing paradigm takes advantage of computing, storage and network resources at the edge of the network to increase the capabilities of the cloud. The processing elements, which operate on various devices such as network gateways and IoT edge devices, provide a mechanism for moving the processing closer to the edge of the network. Since computing resources in fog computing are distributed closer to the edge of the network, that is closer to users and things, the fog computing model can be a better choice for building applications for IoT (Al-Fuqaha *et al.*, 2015; Chiang and Zhang, 2016; Giang *et al.*, 2015).

Fog computing can work as a bridge between smart devices and large-scale cloud computing. Because of its proximity to the end users, in comparison to the cloud data centers, fog computing has the ability to offer services with better delay performance. In addition, fog computing has the ability to improve the overall performance of IoT applications because fog tries to perform part of high-level services, provided by the cloud, in the local resources. Figure 8 illustrates the roles played by fog computing and the cloud data centers to deliver IoT services to the end users. Fog computing can be provided by the mobile network operators because they can offer fog services as one of the SaaS, IaaS or PaaS models to the enterprise businesses at their service network or even cell tower. As an architecture, fog computing supports a growing variety of applications, which include the applications provided in the IoT, fifth-generation wireless systems (5G) and embedded AI (Al-Fuqaha *et al.*, 2015; Chiang and Zhang, 2016).



Source: Al-Fuqaha *et al*. (2015)

Figure 8.
The role of the cloud and fog resources in the delivery of IoT services

To summarize, fog computing differs from cloud computing on some measures. The cloud resides on the internet, whereas the fog is located on the edge of the network. The cloud has a centralized geographical distribution, whereas the fog has a distributed geographical distribution. For the client to be connected to the cloud, he/she needs multiple hopes, whereas in fog computing, the client needs a single hope usually. Therefore, data transmission latency to the offloaded server is reduced in fog computing as compared to cloud computing. For the IoT/cyber-physical systems (CPS) real-time applications, the fog is more suitable than the cloud because fog computing has lower delay jitter. Moreover, since the core network bandwidth is freed up in fog nodes, using fog computing improves the efficiency of the network. Typically, there is a significant difference in data volume between the fog and the cloud. The cloud has massive computational, storage and communication capabilities in comparison to the fog (Al-Fuqaha *et al.*, 2015; Chiang and Zhang, 2016; Munir *et al.*, 2017).

In most literature, fog and edge computing are used interchangeably as a synonym. The researchers in Munir *et al.* (2017) explained the similarities and differences between fog and edge computing. When the goal of edge computing is to provide cloud computing capabilities on the edge of the cellular network, it is called mobile-edge computing (MEC). Edge computing like fog computing pushes computing power, data, services and application away from a centralized node, but the fog has more decentralized and distributed control than the edge. Edge computing is considered as single vendor or proprietary that affects the market adoption as opposed to the fog that is more open. Moreover, in the edge computing paradigm, the radio-access network is typically a cellular network, whereas the radio-access network in the fog can be WLAN, WiMAX and/or cellular (Munir *et al.*, 2017).

Cloudlets and fog computing also are used as a synonym in some literature. Nevertheless, the cloudlets depend on dedicated devices treated as "data center in a box," which have capabilities similar to a data center, but on a lower scale. Cloudlets, like MEC and fog computing, exist closer to the consumer. The cloudlets allow end devices to offload computing to the cloudlet devices in a way similar to a data center. The cloudlet devices are running a virtual machine capable of providing resources to end devices and users in real

time over a WLAN network. The cloudlets only support WiFi as an access mechanism that offers high bandwidth to the end devices, which causes lack of supporting constrained devices like IoT devices. However, fog computing supports non-IP based protocols like BLE and Zigbee. This allows fog computing to connect to a wider range of end devices as well as provide protocol translation (Dolui and Datta, 2017). Dolui and Datta (2017) introduced a comparison between fog computing, MEC, and cloudlets.

Fog interfaces with cloud, other fogs, things and end users are shown in Figure 9. The fog–cloud interfaces will be needed to support fog–cloud collaboration in order to provide end-to-end services. Since different fog nodes or systems may collaborate with each other to jointly support an application, fog-to-fog interfaces are needed. Fog-to-thing/user interfaces would be used to enable fog to provide services to a wide range of end users and devices with widely various capabilities (Chiang and Zhang, 2016).

Giang *et al.* (2015) proposed a distributed dataflow (DDF) programming model for IoT that utilizes the computing infrastructures across the fog and the cloud. To evaluate their framework, the researchers implemented a DDF framework based on Node-RED (Distributed Node-RED or D-NR), which is a visual programming tool that uses a flow-based model for building IoT applications. They demonstrated that their approach facilitates the development process and can be utilized to create different IoT applications that work efficiently in the fog. Munir *et al.* (2017) proposed a fog-centric architecture, IFCIoT, that provides a better performance, energy efficiency, low latency, scalability and improved localized accuracy for the IoT/CPS applications. To achieve better performance for these applications and to support real-time requirements, the authors also proposed a reconfigurable fog-node architecture that can be adapted to the workload running at a given time. Moreover, the authors conducted an ITS as an application use case to illustrate the mapping of the proposed fog. Furthermore, they studied the proposed IFCIoT architecture on other potential consumer applications such as environmental monitoring, localized weather maps, real-time agricultural data control and analytics and smart cities.

5.2.4 Visualization. Visualization plays a critical role in IoT applications because it allows the user to interact with the environment easily. With the ongoing advances in touchscreen technology, the use of tablets and smartphones has turned out to be extremely intuitive. For a non-expert person to take full advantage of the IoT revolution, attractive and easy to understand visualization has to be created. More information can be given in a way that is

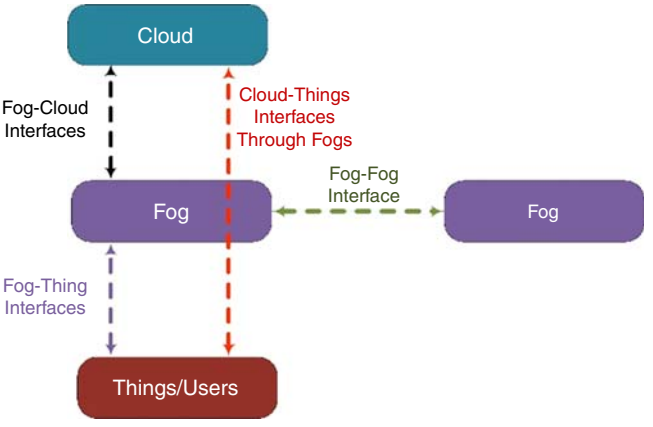


Figure 9.
Fog interfaces

Source: Chiang and Zhang (2016)

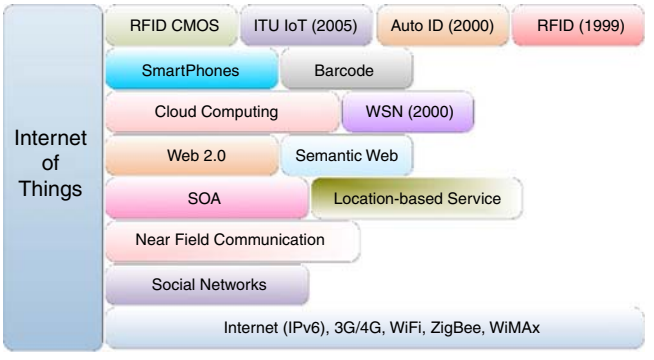
useful to the consumers because of the transition from 2D to 3D screens. This would help in extracting knowledge from raw data and making the proper decisions based on the end-user needs (Gubbi *et al.*, 2013; Singh *et al.*, 2014). New visualization schemes in a 3D landscape for representing heterogeneous sensors data have to be developed (Porkodi and Bhuvaneswari, 2014). In this context, Jeong *et al.* (2015) presented an interactive framework, called AVIoT, for visualizing and composing IoT in indoor environments such as home or small office. The basic building blocks of this framework are virtual sensors and actuators that abstract physical objects and their virtual behaviors over their physical networks. The physical objects behaviors are abstracted and programmed through visual composing tools on the web, allowing the average consumers to easily monitor and identify the behavior without even knowing the underlying physical connections. The users' test is conducted to evaluate the usability of the visual authoring, which demonstrated that the visual authoring was understandable, easy to use and also preferred to typical text-based script programming.

6. IoT enabling technologies and trends

The main idea of IoT is to make everyday objects such as vehicles, refrigerators, medical equipment and general consumer goods equipped with tracking and sensing capabilities. When this vision becomes completed, "things" will also have more sophisticated networking and processing capabilities that will enable them to perceive their surroundings and interact with people (Whitmore *et al.*, 2015). Establishing IoT concept in the real world is possible through the integration of various technologies (Atzori *et al.*, 2010). Figure 10 summarizes the enabling technologies of IoT. Some of these technologies and trends are introduced in the next subsections, whereas some are introduced throughout the paper.

6.1 Identification, sensing and communication technologies

Identification is critical for IoT success. It enables IoT to name and match services with its request; therefore, billions of devices can be uniquely identified and remotely controlled over the internet. All the objects, already connected or going to be connected, must be identified by their unique identification, functionalities and location. On IoT identification, several methods are available such as ubiquitous codes (uCode) and EPC. Addressing methods include IPv4 and IPv6. In the current IPv4, a group of coexistence sensors can be identified geographically, but not individually. IPv6 can identify things globally. Furthermore, the internet mobility attributes in IPv6 may mitigate some of the device identification problems; nevertheless, the heterogeneous



Sources: Da Xu *et al.* (2014), Li *et al.* (2015)

Figure 10.
Enabling technologies for IoT

nature of wireless objects, concurrent operations, variable data types and convergence of data from devices aggravate the problem. To make IPv6 addressing suitable for low-power wireless networks, 6LoWPAN is provided, which is a compression mechanism over IPv6 headers (Al-Fuqaha *et al.*, 2015; Gubbi *et al.*, 2013).

Sensing on IoT means collecting data from related objects within the network and sending them back to the database, data warehouse or cloud. The collected data are then analyzed to take specific actions based on the required services. The communication technologies in IoT connect heterogeneous things/objects to each other to deliver specific smart services. IoT objects should operate on low power even in the presence of lossy and noisy communication links. WiFi, Bluetooth, IEEE 802.15.4, Z-wave and long-term evolution (LTE) advanced are examples of communication protocols used in IoT. RFID, NFC and ultra-wide bandwidth (UWB) are some specific communication technologies that are also in use in IoT communication (Al-Fuqaha *et al.*, 2015).

6.1.1 Mapping RFID tags into IPv6 networks. Nowadays, an IPv6 address may be assigned to each object to be identified globally, whereas RFID tags are not accessible on the internet. Thus, solutions are required to enable the mapping of RFID tags into IPv6 networks. As standardized by Electronic Product Code global (EPCglobal), RFID tags use 64–96-bit identifiers. Integrating RFID tags into IPv6 networks has been investigated recently, and the methodologies for this integration have been proposed (Atzori *et al.*, 2010). For example, Lee *et al.* (2007) proposed a method and equipment for producing IPv6 that mapped EPC code. This mapping method between EPC code and network address will help to obtain real-time information. The EPC is used in RFID system to identify the products. However, the EPC code is not accessible using internet addresses because the EPC code is just an object identifier like the barcode. The authors have suggested the mechanism for producing mapped EPC IPv6 addresses to enable objects to communicate over the internet. In other words, they introduced a method for preparing the EPC code from the electronic tag and for combining the network prefix address with EPC code. The authors have suggested using the 64 bits (interface identifier) of the IPv6 address as the RFID tag identifier, whereas the other 64 bits (network prefix) are used to address the gateway between the RFID system and the internet. However, this approach cannot be used if the RFID tag identifier is 96-bit long.

To solve the aforementioned problem, Yoon *et al.* (2008) suggested a mechanism for communication using address management agents; thus, networks will be using IP in RFID. This mechanism receives an RFID tag ID of different lengths and generates the IP addresses. Moreover, the information of IP address and RFID tag ID is mapped and stored in the address management agent. By using the stored information in the address management agent, the RFID would be able to make networking using IPv6 between RFID tags. The proposed methodology maps the RFID identifier, regardless of its length, into a 64-bit field, which will be used as the interface ID of the IPv6 address. It is clear that the agent must continue to update a mapping between the generated IPv6 addresses and the RFID tag identifier.

A completely different approach is illustrated in IPv6 (2018), as shown in Figure 11. In this approach, the RFID message and the header are included in the payload of the IPv6 packet. The address formats fit without any loss of functionality. In EPC, the RFID space for a company is 60 bits, with a 28-bit object class and a 32-bit serial number. A single IPv6 subnet can map this space completely. With this integration, the RFID object class and serial number become the IPv6 interface ID. Therefore, each RFID tag can be addressable in an IPv6 network. With RFID and IPv6, everything can be tracked.

RFID mobility is not supported in all of the previously discussed cases. In fact, the common assumption is that each RFID tag can be accessed through a specifically given gateway between the RFID system and the network (Atzori *et al.*, 2010).

6.2 Context awareness computing

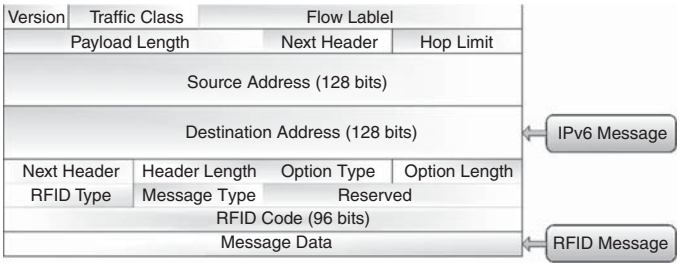
The system is context aware if it uses context to provide pertinent information and/or services to the user according to the user's task. The concept of context awareness emerged in the early 1990s, and it has been employed since then. Moreover, with the introduction of the term "ubiquitous computing" by Mark Weiser in 1991 in his ground-breaking paper "The Computer for the 21st Century", context awareness became more popular. The term "context-aware" was first used by Schilit and Theimer in 1994. Since then, context awareness has been established as a well-known research area in computer science and has become a substantial quality of ubiquitous and pervasive computing systems. Over the last decade, the focus on context-aware computing progressed from desktop applications, web applications, MobiComp, pervasive/UbiComp to the IoT applications (Perera *et al.*, 2014).

6.3 Semantics

Semantic represents the brain of IoT. Semantic refers to the ability to extract knowledge intelligently, which involves the discovery and use of resources and modeling information to provide the required services. Knowledge extraction includes data identification and analysis, which help in making the correct decision to deliver the desired service accurately by sending the demands to the right resource. Web Ontology Language (OWL) and the Resource Description Framework are examples of Semantic Web technologies that provide extensible high-level data representations and advanced query capabilities. In 2011, the World Wide Web Consortium (W3C) adopted the Efficient XML Interchange (EXI) format as a recommendation. EXI is essential with regard to the IoT because it is intended to optimize XML applications for resource-constrained environments. Moreover, EXI minimizes the required storage space and reduces the needed bandwidth by converting XML messages to binary without influencing related resources such as battery life, size of the code, consumed energy for processing and memory size (Al-Fuqaha *et al.*, 2015; Singh *et al.*, 2014).

6.4 Cyber-physical system (CPS)

CPS is a new paradigm of embedded smart systems. CPS integrates the cyber world with the physical world through sensors and actuators to create a global network for businesses and provide real-time data access and data processing services. It is a type of time-sensitive, spatially distributed and large-scale networked systems consisting of physical and computational elements. CPS integrates various devices that have sensing, identification, communication, processing and networking capabilities. CPS is a complex system that requires specific applications, including elastic sensor and actuator interfaces, highly precise timing, small memory footprints and robust safety characteristics. CPS provides a key platform for the creation of advanced industrial applications by integrating innovative



Source: IPv6(2018)

Figure 11. Encapsulation of RFID message into an IPv6 packet

functionalities through the IoT and Web of Things to enable physical processes to get connected to computing and communications infrastructures. There is a trendy interest to apply CPS on Industry 4.0 where CPS is predicted to provide unique solutions to transform and upgrade the operation and role of existing key industrial processes such as CPS-related key techniques, big data management, cybersecurity procedures, industrial infrastructure design and smart manufacturing. There are a number of CPS projects that have been implemented in some industries such as healthcare, aircraft, environmental monitoring, manufacturing, automatic robotics, security surveillance and others. Research on CPS, which can be grouped into theoretical foundations and applications, has grown very rapidly in recent years, and many workshops and conferences on CPS have been organized (Chen, 2017; Lu, 2017).

CPS are developed through collaboration between many engineering disciplines. Each individual discipline is utilizing a powerful software tool; however, connecting these software tools into tool chains to increase CPS efficiency is still a big challenge. Integrated tools are increasingly having considerable importance in CPS areas, as the creation of modern CPS requires seamless integration of development, business and manufacturing tools. Currently, the overall integration effort is limited to technology and platforms/standards such as the IEEE standard for CASE tool interconnections, the Eclipse platform, and the Open Services for Lifecycle Collaboration standard. Large EU projects such as CORDIS, CRYSTAL, SPRINT, EMC2 and iFEST have spent considerable effort on technology such as the Semantic Web and Linked Data. There is no determined way to guide tool chain developers, analysts, decision makers or other stakeholders to understand the current situation for tool chain integration; however, it is an important issue to determine the priorities, dependencies and correct decisions to make the development process more integrated (Gürdür and Asplund, 2018).

6.5 5G–IoT

The 5G–IoT represents the integration of future IoT and the emerging 5G technology. 5G technology can dramatically expand IoT technologies beyond what is possible with existing technologies. 3G and 4G technologies have been widely used in IoT. Since 2012, the LTE to 4G has become more consistent and fastest in comparison to competing techniques like ZigBee, BLE, SigFox, WiMaxb and LoRa. But both 3G and 4G were not optimal for IoT applications. 5G technology is expected to solve the challenges facing 4G networks, such as device computational capabilities, more complex communications, intelligence, etc., to be more suitable for the smart environments, Industry 4.0 and other applications. 5G technology is expected to extend the current IoT by boosting the cellular operations and driving the future internet into the edge. Moreover, by providing the fastest connectivity and capacity, the 5G wireless network can dramatically expand the scope and size of IoT coverage. The 5G will be able to provide huge internet connectivity, where billions of smart devices will be connected to the internet. The 4G LTE will be the foundation of 5G development; however, it is reported that there will be some early 5G networks (beyond LTE Advanced) by 2020. The full 5G will be ready after 2025 when the standardization process is completed. 5G networks will provide a flexible and faster network, which can be implemented via the wireless software define networking paradigm. Moreover, there are some key enabling technologies for 5G–IoT such as wireless network function virtualization, HetNet, direct device-to-device (D2D) communication, advanced spectrum sharing and interference management, just to name a few (Li *et al.*, 2018).

7. IoT standardizations

One of the main challenges of deploying the IoT is standardization. Standards play a crucial role in IoT formation and success because they are important to allow all the actors to access

and use services equally. The lack of standards may decrease the competitiveness of IoT products; therefore, in the last decade, different organizations have developed a number of technical standards such as middleware and interface standards. Research on IoT standards has attracted attention on different levels. At the international level, the ITU, IETF, EPCglobal, IEEE, International Organization for Standardization (ISO) and International Electro-Technical Commission (IEC) have introduced a set of standards for identifying, capturing and sharing data using RFID technologies. At the regional level, the European Telecommunications Standards Institute (ETSI) and the European Committee for Electro-Technical Standardization (CEN/CENELEC) have issued a set of standards related to essential IoT technologies such as RFID and WSN. The Internet of Things European Research Cluster sponsored a number of IoT fundamental research projects such as IoT-A, which was launched to design a reference model and architecture for IoT, and the RERUM project, which focuses on IoT security. The ANSI in the USA is working on standards for managing IoT. ITIF, also in the USA, focuses on new information and communication technologies for IoT. In Japan, they developed an infrastructure, called uID, to connect fundamental research with applied research and development. The Japan government also proposed u-Japan and i-Japan strategies to promote a sustainable ICT society. South Korea conducted RFID/USN and "New IT Strategy" program to advance the IoT infrastructure development. The China government officially launched the "Sensing China" program in 2010. Moreover, the China Electronics Standardization Institute and the China Communications Standards Association are working on developing standards of ultra-high frequency band RFID and semi-passive RFID. In China, there are also 973 projects that have been developed on the essential techniques of IoT and standardization. The development and harmonization of the introduced proposals would encourage the effective development of infrastructure of IoT devices, applications and services. The standards, protocols in the standards and information models, which are developed by cooperated multi-parties, shall be publicly and freely available and be open to all participants. In the world of today's networks, global standards are more relevant than any local agreements (Chen *et al.*, 2014; Karagiannis *et al.*, 2015; Li *et al.*, 2015). Table VII shows a summary of standards in IoT.

The integration of embedded devices and IoT introduced many challenges since many of the current internet standards and protocols were not designed for these type of devices. The embedded devices have very limited power, memory and processing capabilities, and they are often disabled for long periods (sleep periods) to save energy. These constrained networks also have significant packet loss, different traffic patterns, frequent topology changes, low throughput and small useful payload sizes. Moreover, there are several communication media that could be used such as BluetoothLE, IEEE 802.11, IEEE 802.15.4, RFID and NFC. One attempt to address such differences and to standardize IPv6 at the network layer protocol is the adaptation layers proposed by the IETF IPv6 over networks of resource-constrained nodes (6Lo) workgroup. RPL will be used as the routing protocol for those networks. As an effort to standardize and deploy IP version 6 and the IoT, IETF created the IPv6 over low-power WPAN (6LoWPAN) workgroup to address the use of IPv6 over IEEE 802.15.4 networks. The 6LoWPAN workgroup was active from 2005 until 2013, and it published RFCs concerning overview and hypothesis, transmission of IPv6 Packets over IEEE 802.15.4 Networks, neighbor discovery, IPv6 compression header, routing requirements and use cases. Another IETF workgroup, active since 2008, is the Routing Over Low-power and Lossy networks (ROLL). Among the RFCs related to the routing requirements in different environments, metrics and security issues, RFC 6,550 –RPL (IPv6 Routing Protocol for Low-power and Lossy Networks) was introduced. RPL-supported traffic flows include point to point, point to multipoint (i.e. from a sink to nodes) and multipoint to point (i.e. from nodes toward a sink). Therefore, the RPL can support different types of applications running on a particular WSN

Technology	Standard
RFID	RFID payment system and contactless smart card: ISO 14443/15693
	ISO 18000-1 – Generic Parameters for the Air Interface for Globally Accepted Frequencies
	RFID tag ISO 11784
	ISO 18000-7 – for 433 MHz
	RFID air interface Protocol: ISO 11785
	ISO 18000-3 – for 13.56 MHz
	ISO 18000-6 – for 860 to 960 MHz
	Mobile RFID: ISO/IEC 18092 ISO/IEC 29143
	ISO 18000-4 – for 2.45 GHz
	ISO 18000-2 – for frequencies below 135 kHz
Data content and encoding	EPC Global Object Naming Service (ONS)
	EPC Global Electronic Product Code, or EPCTM
	EPC Global Physical Mark Up Language
Electronic Product Code (Auto-ID)	Global Location Number (GLN)
	Serial Shipping Container Code (SSCC)
	Global Trade Identification Number (GTIN)
Sensor	Sensor Interfaces: IEEE 1451.x, IEC SC 17B, EPC global, ISO TC 211, ISO TC 205 JTC1 WG7
	ISO/IEC JTC1 SC31 and ISO/IEC
Communication	UWB
	IEEE 802.15.6 (Wireless Body Area Networks)
	IEEE 802.15.1(Bluetooth, Low energy Bluetooth)
	IEEE 802.11 (WLAN)
	Low Power WiFi
	IEEE 802.15.4(ZigBee)
	IPv6
	IEEE 1888
	3G/4G
Network management	IETF SNMP WG, IEEE 1588, ITU-T SG 16, ZigBee Alliance, ITU-T SG 2
Middle	ITU-T SG 16, ISO TC 205
QoS	IETF, ITU-T

Table VII. A summary of standards in IoT

Source: Li *et al.* (2015)

network (or low-power lossy network – LLN). The new standards enable the recognition of the IoT, where end-to-end IP-based network connectivity with tiny objects such as sensors and actuators becomes possible. As IETF 6LoWPAN and ROLL groups focus on the IPv6 addressability and routing, the IETF Constrained RESTful Environments group focuses on realizing an embedded counterpart for RESTful web services. By combining these protocols, an embedded protocol stack can be created that has characteristics similar to traditional internet protocol stacks. Actually, IETF protocols were designed to enable easy translation between internet protocols and sensor protocols (de Oliveira *et al.*, 2015; Ishaq *et al.*, 2013). The software-defined network (SDN) can be used in the context of IoT. de Oliveira *et al.* (2015) examined RPL and TinySDN protocols concerning routing features, interoperability possibilities and support for legacy networks. Then they discussed how SDN could improve WSN and IoT deployments by enabling better management and resource sharing.

IoT has been an important topic of standardization in the International Telecommunications Union (ITU) in different study groups (SGs). The ITU has been standardizing IoT for several years in the Telecommunication Standardization Sector (ITU-T). Industry and academic experts take part in several ITU-T SGs to standardize different aspects of IoT, such as architectural framework, capabilities, requirements, use cases and applications. Moreover, to further consolidate IoT standardization activities, a

new ITU-T SG 20 was established recently. This SG develops international standards for improving the coordinated progress of global IoT technologies, with an initial focus on IoT applications in smart sustainable cities and communities (Kafle *et al.*, 2016).

8. IoT protocols

Although the term IoT has been introduced to the public for more than 15 years, great efforts still have to be done by innovative corporations and research groups. Till now, there is no clear definition of a unified IoT architecture and there is no common agreement in defining standards and protocols for all IoT parts. The current IoT platforms use a framework that contains technologies that partially meet certain IoT requirements. Whereas developers use existing technologies to build the IoT, research groups are working on adapting protocols to the IoT to improve communication. Many IoT standards are introduced to simplify the jobs of application programmers and service providers. Nevertheless, there are few existing guidelines to help the programmer choose the right IoT protocols to accomplish the desired goals of performance, reliability, energy efficiency and expressiveness. There are many factors that affect the choice of protocols, the most important of which are the computational and communication power of the devices, the battery consumption and the final application in mind (Karagiannis *et al.*, 2015; Mun *et al.*, 2016).

Various groups have provided protocols to support IoT ecosystem such as the efforts led by the IETF, W3C, IEEE, EPCglobal and the ETSI. Table VIII provides a summary of the most prominent protocols, which are categorized into four groups: application protocols, service discovery protocols, infrastructure protocols and other influential protocols, based on the suggested classification by Al-Fuqaha *et al.* (2015).

The application layer protocols are thoroughly explained in Al-Fuqaha *et al.*, 2015; Karagiannis *et al.*, 2015; Mun *et al.*, 2016, with details of architecture, security, transport layer protocol (TCP or UDP), QoS, characteristics, drawback, etc., whereas service discovery and influential protocols are explained in Al-Fuqaha *et al.* (2015). Infrastructure protocols, which include communication technologies, data link layer protocols, network layer encapsulation protocols and network layer routing protocols, are fully explained in Al-Fuqaha *et al.*, 2015; Al-Sarawi *et al.*, 2017; Sethi and Sarangi, 2017; Triantafyllou *et al.*, 2018.

The advent of standardized protocols is not an endpoint. In reality, it is a starting point for exploring additional open issues that must be resolved to achieve a universal IoT. There are still many open challenges such as dealing with sleeping nodes, energy efficiency, security, scalability, resource representation, linking with cloud services, easy creation of applications, interoperability with other wireless standards, integration with existing web service technologies and tools, use of semantics and maintainability (Ishaq *et al.*, 2013).

9. IoT cybersecurity

Cybersecurity is an unavoidable issue that must be solved while developing the IoT. If cybersecurity is not well managed, the hackers will take advantage of the weaknesses and defects of IoT objects and then will distort data or disrupt systems through the global IoT network. IoT failures and attacks may outweigh any of its benefits and it would be unlikely that the involved stakeholders will adopt IoT solutions widely. Hence, new techniques and methodologies have to be developed to meet IoT security, privacy and reliability requirements. There are various techniques and devices used in IoT, such as smartphones, bar codes, cloud computing and social networks, that to some extent affect IoT cybersecurity. At the earlier stage of IoT deployments, for example, based on RFID only, security solutions were mostly placed in an ad hoc way. From an open ecosystem perspective, where different stakeholders can participate in a particular application scenario, a number of security challenges do arise, for example, one of the stakeholders

Application Protocols		DDS	CoAP	AMQP	MQTT	MQTT-SN	XMPP	RESTFUL	WebSocket
Service discovery		mDNS				DNS-SD			
Infrastructure Protocols	Network Layer Routing Protocols	RPL		P2P-RPL		CORPL	CARP	AODV, LOADng and AODv2	
	Network Layer Encapsulation Protocols	6LoWPAN		IPv4/IPv6		6TiSCH	ZigBee IP	IPv6 over G.9959	
		IPv6 Over Bluetooth Low Energy		IPv6 over NFC		IPv6 over MS/ TP−(6LoBAC)	IPv6 over DECT/ ULE	IPv6 over 802.11ah	
	Data Link Layer Protocols	IEEE 802.15.4e (TSCH)	IEEE 802.11 ah – wifiHallow		WirelessHART	Z-Wave	INGENU RPMA (IEEE 802.15.4k)	Bluetooth Low Energy	Zigbee Smart Energy
		DASH7	HomePlug		G.9959 (~Z-Wave)	LTE-A	LoRaWAN	Weightless	DECT/ULE
	Communication technologies	Bluetooth – BLE	ZigBee		Z-Wave	6LoWPAN	Wifi-ah (HaLow)	LTE-A or eMTC (3GPP)	
		EPCglobal	2G(GSM), 3G,4G, 5G (3GPP)			Weightless-N/-W/-P		Tread	
		RFID	NFC		LoRaWAN	SigFox	Neul	Dash7	
		WirelessHART	EnOcean		DigiMesh	Ingenu	ANT & ANT+	NB-IoT (3GPP)	
Influential Protocols		IEEE 1888.3, IPSec				IEEE 1905.1			

Table VIII. Standardization efforts in support of the IoT

Sources: Al-Fuqaha *et al*. (2015), Triantafyllou *et al*. (2018)

owns the physical sensors/actuators, the other stakeholder handles and processes the data and various stakeholders provide different services to the end users based on such data. Cybersecurity ensures that IoT will be a safe network for people, things, processes, hardware and software. Cybersecurity aims to provide the world with a higher level of confidentiality, integrity, availability, scalability, accessibility and interoperability in IoT global network. In the upcoming years, cybersecurity issues will be one of the main research challenges of IoT (Lu and Da Xu, 2018; Miorandi *et al.*, 2012).

There are many reasons that make the IoT more vulnerable to attacks. The most important reasons are as follows: IoT's components spend most of their time unchecked, which makes them more vulnerable to physical attacks; since most of the communications in IoT are wireless, eavesdropping would be very simple; traditional security models cannot be adapted to new security challenges and the era of massive information produced by IoT; traditional security mechanisms and protocols are not appropriate for IoT devices because they were designed for the existing devices, which have low levels of interoperability, scalability and integrity; and IoT components mostly have low power and computing capabilities; hence, they cannot support complex security schemes (Atzori *et al.*, 2010; Lu and Da Xu, 2018).

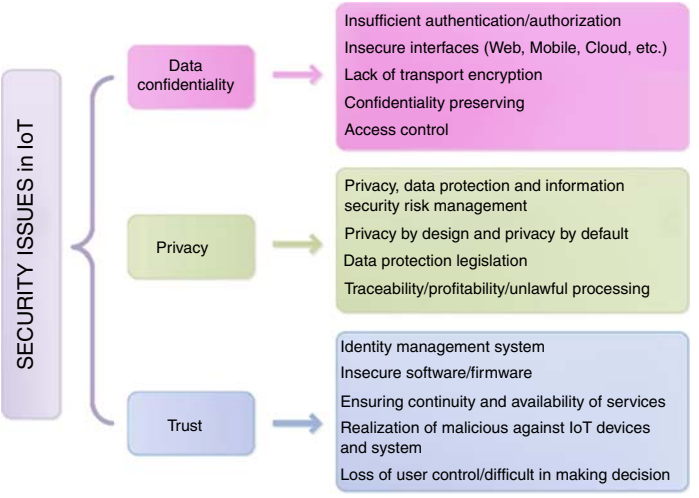
For the IoT to be successful and to provide interoperability, reliability, effectiveness and compatibility of the operations in a secured global scale, there is a need for security standardization at various levels. Each connected device in the IoT could be used as a doorway into the personal data or the IoT infrastructure. Moreover, the IoT introduces a new level of potential risks since mashups, interoperability and autonomous decision making launch potential vulnerability and security loopholes. The risk of privacy will arise in the IoT because the complexity may create more service-related security vulnerabilities. Furthermore, the IoT deals with a lot of information associated with our personal life like

date of birth, location, budgets, etc., which is considered as one of the big data challenges. Security at both service applications and physical devices is important to the operation of IoT. Open issues remain in some areas, for example, network protocols, identity management, standardization, trusted architecture and security and privacy protection. Security must be addressed throughout the IoT lifecycle from the initial design to running services (Li *et al.*, 2016). Figure 12 shows the main security challenges in IoT, which include data confidentiality, privacy and trust.

Data confidentiality is a key issue in IoT scenarios. Data confidentiality shows that data can be accessed and modified only by the authorized entities. The data of IoT applications will be linked to the physical realm; thus, ensuring data confidentiality will be essential to many use cases. Moreover, in IoT applications, the data may not only be accessed by users, but also by authorized objects. Therefore, defining the access control mechanism and the process of object authentication must be addressed (Miorandi *et al.*, 2012).

Privacy determines the rules by which data that refer to the individuals can be accessed. Privacy is a real issue that may limit the evolution of IoT. The lack of appropriate mechanisms to ensure the privacy of personal and/or sensitive information will reduce the adoption of IoT technologies. The main reason, which makes privacy a prerequisite for IoT, lies in the fact that IoT is expected to be used in critical applications like healthcare. Moreover, the risk of violation is increased because of using wireless channels, which make the system vulnerable to attack and eavesdropping due to remote access capabilities (Miorandi *et al.*, 2012). Whereas privacy issues in traditional internet arise mostly from internet users (individuals who play an active role), privacy issues in IoT arise even from people who do not use IoT services. Accordingly, individuals must be able to determine which of their personal data can be collected, who collects such data and when these data can be collected. Moreover, the collected data should be used only when they are urgently needed to support services authorized by the accredited service providers (Atzori *et al.*, 2010).

Trust is a complex idea used in a wide variety of contexts. Depending on the adopted perspective, different definitions of trust are possible. The main problem in many approaches to defining trust is that these definitions do not create metrics and evaluation standards of



Sources: Li *et al*. (2016), Miorandi *et al*. (2012)

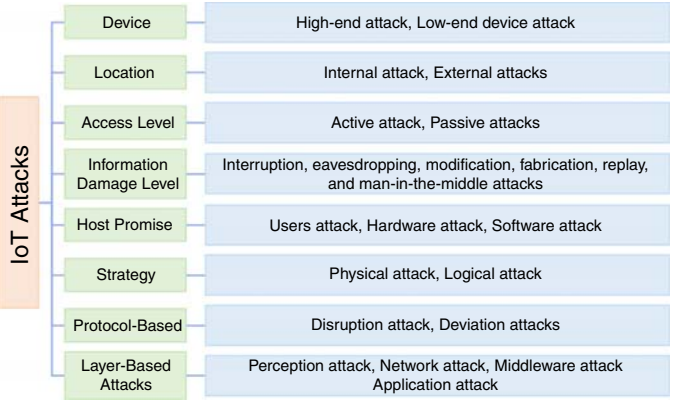
Figure 12.
Security challenges in IoT

trust. Trust is considered a very important concept of IoT, even if the dynamic and fully distributed nature of IoT makes it very difficult to deal with the trust challenges. In IoT environment, where smart objects take the decisions themselves, the first trust relationship must be set up between humans and the objects around them (Miorandi *et al.*, 2012).

Lu and Da Xu (2018) introduced a taxonomy of IoT attacks. The researchers classified IoT attacks into eight categories, as shown in Figure 13. They also thoroughly surveyed the important aspects of IoT cybersecurity, especially the discussion of the possible four-layered IoT cybersecurity infrastructure, the major countermeasures against IoT attacks, the threats regarding application industries and the state-of-the-art current situation and possible future directions.

As an open ecosystem that integrates diverse research areas, IoT is very vulnerable to attacks against security, privacy, availability and service integrity. In Li *et al.*, 2016; Lu and Da Xu, 2018, the authors modeled the IoT as a four-layer architecture: the sensing layer, the network layer, the service layer and the application interface layer. At the lower layer, the sensing layer, the devices cannot provide well security protection because they have very limited energy supply and computation capacity. At the middle layers (the service layer and network layer), the network is prone to an interception, eavesdropping and DoS attacks. Furthermore, a self-organized topology, without centralized control in the network layer, can cause attacks against authentication, for example, node suppression, node replication and node impersonation. At the upper layer, the application layer, the vulnerability problems of all layers can be mitigated using data aggregation and encryption. To build a trustworthy IoT system, a self-adaptive security policy framework and system-level security analytics are needed. Each layer is able to provide corresponding security controls such as device authentication, confidentiality in transmission, access control, data integrity and availability and the ability to prevent attacks. Furthermore, the security requirements on each layer might be different due to its features. Attacks on IoT four-layer architecture are shown in Table IX.

IoT concept was coined depending on the RFID identification and tracking techniques. Authentication is necessary to prevent data attacks between two things on the internet. As the RFID technique joined to create a large sensor network, policy questions arise, which make the security not just a technical issue. RFID cybersecurity measures include data encryption, access control, cryptography technology schemes, IPSec-based security channels and physical cybersecurity schemes (Li *et al.*, 2016; Lu and Da Xu, 2018).



Source: Lu and Da Xu (2018)

Figure 13.
Taxonomy of cybersecurity attacks on IoT

Layers	Job	Attack focus	Security requirements	Security threats and vulnerabilities
Sensing layer	Sensing objects and data	Confidentiality	Confidentiality, data source authentication, device authentication, integrity, availability, and timeless, etc.	Unauthorized access, spoofing attack, selfish threat, malicious code, denial of services (DoS), transmission threats, routing attack, replay attacks, node capture attacks, malicious data attacks, side channel attack (SCA)
Network layer	Networking and data transmission	Confidentiality Privacy Compatibility	Overall security requirements such as confidentiality, integrity, privacy protection, authentication, group authentication, keys protection, communication security, etc.	Data breach, denial of services (DoS), public key and private key compromise, malicious code, transmission threats, routing attack, privacy leakage, fake network message, man-in-the-middle attack, spoofing, altering or replayed routing information, Sybil, wormholes
Service layer	The service layer relies on middleware technology, which is an important enabler of services and applications	Authenticity Integrity Confidentiality	Authorization, service authentication, group authentication, privacy protection, integrity, security of keys, non-repudiation, anti-replay, availability	Privacy threats, Services abuse, identity masquerade, service information manipulation, repudiation, denial of services (DoS), replay attack, routing attack, service information sniffer and manipulation, malicious insider, underlying attack, third-party relationships attack, virtualization threat
Application interface layer	Provision of requested service	Data privacy Identity Authentication	Remote safe configuration, software downloading and updating, security patches, administrator authentication, unified security platform, etc.	Misconfiguration, phishing attack, virus, worms, Trojan Horse and Spyware, malicious scripts, unauthorized access

Sources: Li *et al.* (2016), Lu and Da Xu (2018)

Table IX.
Attacks on IoT four-layer architecture

Another enabling technology is the WSN, in which attackers can actively and aggressively attack WSN-related data or things. Hence, many protection measures have to be taken to deal with different attacks in WSN. These measures include the following: key management, in which the security keys will be generated and updated and the proper algorithm can be built; secret key algorithms, which include symmetric key and asymmetric key algorithms; security routing protocol that includes the SPINS security framework protocol, which is widely used in secure routing technologies and encompasses the Secure Network Encryption Protocol, and Micro Timed Efficient Streaming Loss-tolerant Authentication Protocol; authentication and access control; and physical security design, which covers the design of the nodes and the antennas (Lu and Da Xu, 2018).

The integration of WSNs and RFID enables the implementation of IoT in industrial services and the further deployment of services in more extended applications. Nevertheless, the security issues in this integration encompass the following challenges: privacy of RFID devices and WSNs devices; identification and authentication, which have to be well protected from tracking by an unauthorized user; communication security, the

communication between IoT devices and RFID devices introduces security threats, which need to be addressed proactively by implementing the appropriate measures well; trust and ownership, the authenticity and integrity of the communicating things such as sensor nodes and RFID tags; integration; and user authentication (Li *et al.*, 2016).

Jeon *et al.* (2018) used an open-source Mobius platform to propose a new IoT server platform. The researchers proposed a data storage method by creating a blockchain as a database instead of a common/traditional server construction method such as MySQL server. This research introduced a public fee system service configuration of the IoT platform with enhanced security, exploiting smart contract and encryption and authentication methods of Ethereum blockchain. However, MySQL's Mobius configuration has many vulnerabilities and security threats, many of which have not yet been addressed in this research.

In previous work, Dorri *et al.* (2016) proposed a method that addressed security and privacy challenges by taking advantage of the Bitcoin (BC), which is an unchangeable logbook of blocks. For representing and discussing the idea, the researchers used a smart home as a case study. The proposed approach has three main tiers: overlay, smart home and cloud storage. In Dorri *et al.* (2017), the researchers dug deeper. They outlined the different core components of the smart home tier and discussed the various associated transactions and procedures. Each smart home is equipped with a high resource, always online device, known as "miner", which is responsible for handling all communications inside and outside the home. The miner maintains a private and secure BC, which is used to control communications and auditing. Simulation results showed that overhead expenses of their method are low and manageable in IoT resource-constrained devices. They argued that the overheads, in terms of traffic, processing time and energy consumption, can be tolerated because of the offered security and privacy features. This research was the first work aimed at improving BC in smart homes. However, there is a need to investigate their framework applicability in other IoT domains.

Singh *et al.* (2018) provided guidance on the use of blockchain technology through cases to make a more secure and trustable IoT model. The researchers concluded that IoT is not going to be a full member of a blockchain network because blockchains involve a significant overhead computing, which is improper for most IoT resource-constrained devices. However, the IoT will certainly benefit from blockchain technological functionalities through the APIs provided by the network nodes or any specialized intermediaries. Using these functionalities, the IoT could be made highly secure. The researchers also have discussed the new and prominent blockchain technology cybersecurity points. It is well known that blockchain mostly focuses on funding research of Bitcoin as a blockchain-based cryptocurrency. The researchers in this paper tried to introduce blockchain technology for IoT to transfer data between connected devices and the internet securely.

Inspired by immunology, a novel approach to IoT security is proposed in Liu *et al.* (2013). The proposed approach used traditional security models for reference and adopted circular links to construct a dynamic defense system for IoT security. The whole process for IoT security was dynamic. In addition, the principles of immunology were simulated and applied to the key links. The authors made the proposed model suitable for the IoT environment by changing the defense ways for IoT security. The results of the simulation showed that the proposed approach has provided a novel effective method to ensure IoT security.

A security-on-demand technique, to adjust or change the security functionality of a device easily, has been proposed in Chung *et al.* (2016). The proposed technique suggested using a security profile and configuration map to generate and reconfigure a device image for just-in-time security configuration. By configuring appropriate security modules, the researchers used the security profile to gather information and dependency analysis. Then, the determined modules were included in the map. Since the map contains information

about the security modules available in the device, the security modules can be easily appended or replaced according to the map without renewing the device image.

Yoon and Kim (2017) proposed the functional architecture of remote security management services to improve the security and safety of IoT devices. Various security functions were provided and managed by the remote security management server in an integrated and systematic manner. Accordingly, various incidents of infringement, which may occur in the IoT environment, can be prevented in advance and the damage can be minimized by enabling rapid and effective countermeasures even in the case of a severe attack.

IoT devices are expected to increase rapidly. At the same time, there will be a growing interest in IoT platform that provides the common functions of IoT devices. Various IoT platforms such as oneM2M, FIWARE and IoTivity have been developed. The vulnerability of the IoT platform will affect IoT devices directly and will have a decisive impact on all connected IoT platforms during their interoperability process. In the case of oneM2M, security architecture has been discussed; nevertheless, there is no formal security component currently provided. Therefore, Oh and Kim (2017) developed a oneM2M security component according to the oneM2M standard. This security component can be used in Mobius that was created by Korea Electronics Technology Institute. The introduced OAuth 2.0-based oneM2M security component provided authentication and authorization, which are essential goals of IoT security to secure interoperability between IoT platforms.

Pérez *et al.* (2016) presented ARMOUR, a research project funded by the EU. ARMOUR aimed to develop a methodology for testing and verifying technological experiments in large-scale environments that require the interaction of a wide variety of heterogeneous IoT devices. In ARMOUR, a methodology was determined to experiment, validate and adopt different technological solutions in large-scale conditions. In addition, a set of bootstrapping, group sharing and software programming experiments were proposed, on which different tests have been taken to verify their security and trust in IoT scenarios. Nevertheless, the researchers needed to conduct different tests by changing the technologies used to carry out large-scale experiments. Therefore, the obtained results would be disseminated on security communities like OneM2M to contribute to accredited activities.

10. IoT quality of service

QoS is another important issue with the networking aspects related to the characteristics of the traffic. It is known that the characteristics of traffic in WSNs are highly dependent on the application scenarios. It was not a big problem in WSN because the attention was focused on the flow of traffic within the WSN itself. Nonetheless, when the sensor nodes become part of the internet, according to the IoT paradigm, complications do arise. In IoT scenarios, the internet will be crossed through a large volume of data generated by distributed sensor networks for heterogeneous purposes with very different traffic characteristics. Moreover, since the deployment of large-scale and distributed RFID systems is still at its inception, the characteristics of the relevant traffic flows have not been studied yet; thus, traffic over IoT is not fully known. Even if most of the work has been done on QoS support on WSN, the problem is still not fully explored in RFID systems. Consequently, a major research effort is required on the area of IoT QoS support. QoS management schemes introduced to M2M scenarios can be applied to the IoT QoS scenarios. Certainly, this should be just a starting point because specific solutions for IoT must be presented in the future (Atzori *et al.*, 2010). The essential factors such as bandwidth and throughput are used to measure QoS of IoT applications (Porkodi and Bhuvaneswari, 2014).

Vithya and Vinayagasundaram (2014) introduced a system that uses a new layered network architecture approach for IoT. The data of the captured images or video from remote locations must be forwarded without delay, which is a rigorous task in a large-scale

IoT ecosystem. By using agents on different wireless networks, the packets are routed depending on the priority. The goal of the researchers' study is to avoid delay and interference during data transfer from various wireless networks. The study dealt with data transfer without congestion under many cramped networks. It introduced low overhead for processing and conducted realistic failure scenarios to provide traffic priority in a flexible way, which would be robust to both traffic variations and topology failures. The paper introduced a QoS routing method by prioritizing networks to differentiate the selective sensing neighborhoods between the sender and the receiver in the smart space.

The best effort service model cannot be used in IoT as a mechanism to handle sensitive traffic delays. Because of the heterogeneous and limited buffer capacity, smart devices need an efficient buffer management scheme and outstanding service priorities to provide preferential treatment to sensitive traffic delays. The researchers in Awan and Younas (2013) proposed a cost-effective analytical model of a low-capacity queue system with a proactive self-service priority and a push-out management system. The smart devices' performance, under various traffic conditions that meet the QoS requirements of IoT devices, can be predicted using the analytical model.

Li *et al.* (2014) proposed a QoS scheduling model of the three layers for service-oriented IoT. This framework includes the sensing layer, the network layer and the application layer. At the application layer, QoS scheduler explores the configuration of the optimal composition of QoS-aware services by using knowledge of each component service. At the network layer, the model is designed to handle the HetNet environment. At the sensing layer, it handles the acquisition of information and scheduling of resource allocation for various services. The proposed QoS-aware scheduling model for service-oriented IoT architecture is able to improve the scheduling performance of IoT network and reduce the cost of resources.

In IoT fog computing QoS, some research works have been proposed. For example, Brogi and Forti (2017) proposed a simple, general and scalable model to support QoS-aware deployments of IoT applications to fog infrastructures. The proposed model described, at an appropriate abstract level of interest, characteristics and operational systemic qualities of fog facilities and IoT applications to be deployed. These operational systemic qualities include the quality of the available infrastructure (bandwidth and latency), interactions between things and software components and business policies. Algorithms have been provided to determine eligible deployments for an application to a fog infrastructure.

FogTorch, a Java tool, based on the proposed model, has been prototyped in Aazam and Huh (2015) and Aazam *et al.* (2016). In the previous work of Aazam and Huh (2015), a basic mathematical model has been proposed for resource estimation in the fog. In Aazam *et al.* (2016), the authors provided a methodology for resource estimation based on historical records to minimize the resource underutilization as well as enhance the QoS for multimedia in IoT. They developed a methodology, named MEdia FOg Resource Estimation, to provide resource estimation on the basis of service give-up ratio, called relinquish rate and improve QoS on the base of previous net promoter score and quality of experience records. CloudSim is used to implement the algorithms. These algorithms also have been applied to real IoT effects based on Amazon EC2 resource pricing.

11. IoT applications

The possible applications offered by the IoT are numerous and diverse in all areas of everyday life. IoT new applications are likely to improve the quality of our daily lives in many areas and environments: at home, workplace, hospital, gym, while traveling, just to name a few. With IoT, these environments would be equipped with intelligent objects that are able to communicate with each other and exchange the captured information; therefore, a very wide range of applications can be deployed. IoT application areas are limited only by imagination (Atzori *et al.*, 2010; Whitmore *et al.*, 2015). In this survey, IoT applications are

categorized according to their domains into five groups: community, transportation, environmental, industrial and entertainment domains. Table X shows the applications in each category, which will be explained in the next subsections.

11.1 Community domain

11.1.1 Smart home/smart building. IoT smart home services enhance personal lifestyle by facilitating remote monitoring of home appliances and systems such as energy consumption meters, air conditioners, heating systems, etc., and performing operations remotely and automatically, for example, closing the windows automatically and lowering the blinds of upstairs windows based on the weather forecast (Miorandi *et al.*, 2012). IoT smart home can help in reducing and optimizing resource consumption like electricity and water, detecting emergencies, providing home safety and security and finding things at home easily. This would help in reducing operational expenditure and carbon footprint, which would have the impact of the global greenhouse gas emission (Al-Fuqaha *et al.*, 2015; Asghar *et al.*, 2015; Khan *et al.*, 2012; Miorandi *et al.*, 2012).

Smart buildings connect building automation systems (BAS) to the internet. BAS can help in improving building maintenance, such as cooling/heating system or blinking dishwasher, by providing signals when there is a problem that needs to be checked and solved. Maintenance requests can be sent automatically to the contracted company without any human intervention. Smart buildings also would enable homeowners to protect their houses against disasters (Al-Fuqaha *et al.*, 2015; Asghar *et al.*, 2015; Miorandi *et al.*, 2012).

11.1.2 Smart grids and smart metering. Smart grids and smart metering can use IoT to collect data about energy consumption for the purpose of improving and enhancing the energy consumption in houses and buildings. To achieve better load balancing, this model can be extended over a city level, not just a house or a building (Singh *et al.*, 2014). IoT technologies can be utilized to make cities more efficient. IoT-collected data are typically integrated into reports that illustrate the usage patterns and include recommendations to reduce power consumption and cost. Employing IoT in the smart grid would reduce potential failures, increase efficiency and improve the QoS. It would also help power

Community domain	Smart home/smart building Smart grids and smart metering Smart cities Security and surveillance Healthcare Social IoT (SIOT)
Environment domain	Smart agriculture and smart water Animals and breeding Environmental monitoring Recycling
Transportation domain	Intelligent transportation systems (ITS) or transportation cyber-physical systems (T-CPS) Robot taxi
Industry and manufacturing domain	Industrial automation Oil and gas industry Supply chains/logistics Pharmaceutical industry Insurance industry
Entertainment industry	Enhanced game room Media Music cognition

Table X.
IoT applications

suppliers to control and manage resources by providing power in proportion to population growth and improving their services to meet the needs of their consumers. Smart grids employee IoT to connect millions or billions of buildings to the network of power suppliers to be used in collecting, analyzing, controlling, monitoring and managing energy consumption (Al-Fuqaha *et al.*, 2015; Whitmore *et al.*, 2015).

11.1.3 Smart cities. A smart city is designed to improve the quality of life for city residents by making it easier and more comfortable to find information of interest. Smart city applications, in which IoT plays a vital role, may include the following: smart roads, monitoring of vehicles and pedestrian levels, monitoring of parking spaces availability, intelligent highways that send warning messages depending on climate conditions or unexpected incidents like traffic jams or accidents, adaptive lighting on streets, monitoring of sound in sensitive areas of cities, weather monitoring and trash collection and detection of waste container level. Sensors can be used in forensics by detecting violations and transferring relevant data to law enforcement agencies in order to identify the violator or store the information for future use. This information would help in the event of an accident to analyze the scene of the subsequent incidents (Al-Fuqaha *et al.*, 2015; Miorandi *et al.*, 2012; Porkodi and Bhuvaneswari, 2014).

11.1.4 Security and surveillance. Security and surveillance become necessary for shopping malls, factories, parking lots, enterprise buildings and many other public places. IoT technologies are increasing dramatically in the field of security and emergency such as perimeter access control, detecting the presence of liquids, measuring the level of radiation and detecting the hazardous explosive of gases. The perimeter access control can be used in detecting unauthorized access and controlling people entrance to restricted areas. Liquid detection is used for detecting the presence of liquid in the data centers, warehouses, and sensitive places to prevent breakdowns and rustiness. The measurement of radiation levels in the surroundings of nuclear power stations is used to generate alerts in the case of leakage. IoT is also used to detect the leakage of gases in industrial environments, chemical plants surroundings and inside mines (Porkodi and Bhuvaneswari, 2014). Furthermore, sensors that monitor people's behavior can be used in detecting the presence of people who are acting suspiciously. Thus, early effective and efficient warning systems can be built. At the same time, standardization policymakers are engaging in a series of initiatives to guide the development of IoT in order to maximize their socio-economic value while reducing threats to data confidentiality and privacy (Miorandi *et al.*, 2012).

11.1.5 Healthcare. Monitoring human body health conditions is a multidisciplinary mission that includes sensing and computing network, AI and anatomy. The IoT has a number of applications in the healthcare sector, which will enhance the existing living assistance solutions and significantly increase the survival rate of fatal diseases. Medical sensors can be implanted in, wearable on or placed around the human body to monitor body vital signs such as body temperature, blood pressure, heart rate, cholesterol level, blood glucose and breathing activity for better patient care. These sensors will be used in collecting data while monitoring patient activities in their living environments. After collecting the data locally, these data would be transferred to remote medical centers to conduct advanced remote monitoring and take fast response action when needed (Miorandi *et al.*, 2012; Porkodi and Bhuvaneswari, 2014).

Without a doubt, the IoT will be the basis for future smart rehabilitation system aimed to mitigate the problem of aging populations and the lack of healthcare professionals. Smart IoT-based technology is expected to become an indispensable tool in modern healthcare systems. Unlike traditional rehabilitation services in local hospitals, all relevant resources in communities of IoT-based system are shared through smart rehabilitation to provide convenient and flexible treatment for patients. In this way, the use of rehabilitation

resources can be maximized (Yan *et al.*, 2015; Yang and Xu, 2018; Yuehong *et al.*, 2016), for example, a scenario of aged persons, infants or pregnant ladies who live in a village and have RFID tags on their bodies to track their vital health signs and medical history. Any unusual activity on their bodies will trigger a warning or an alert to the nearby local medical assistance. The medical assistance, then, will use this information to provide the proper response action for the person in need. In the case of higher incidence, nearby hospitals can be alerted and the costs of hospital treatment can be reduced through early intervention and treatment (Singh *et al.*, 2014).

Yan *et al.* (2015) developed a new model and algorithms for detecting the abnormality in health conditions according to the data captured by the wearable wireless sensor network. The new proposed algorithm identifies abnormalities by preprocessing and transforming the collected raw data into a set of directed graphs to represent the correlation between data flows from different sensors. Then, it analyzes these directed graphs to identify variations and frequency patterns and finally, it quantifies health conditions by the coefficient number, which depends on identified variations and patterns.

Xu *et al.* (2018) introduced a system for analyzing healthcare data for the regional medical union, which was designed to support the physicians from different hospitals to assess patient's health conditions in a comprehensive data view. The designed healthcare data analysis system focuses on how heterogeneous healthcare data can be integrated and shared on the basis of MobiComp technology and semantic processing. Behavior patterns are extracted from the physiological index values. Tags are created from social networking data by discovering hot words of interest to people living in a common area. Those hot words are divided into three different topics, namely, health, sports and diet. An experiment was conducted, in which data of 18 students about the day-to-day diet, blood pressure and heartbeat were collected for one month. After finishing the experiment, the data of 14 students were complete and easy to use. The complete data were presented to demonstrate the system's feasibility in supporting the analysis of healthcare data. The authors suggested the design of a convenient fog–cloud architecture for regional medical unions to properly balance data processing workloads between hospital information systems and smart medical devices.

Xu *et al.* (2017) introduced cloud–m-health monitoring system (MHMS) framework to support remote health monitoring. Cloud–MHMS introduces data integration and interoperability between several medical agencies such as Class-A comprehensive hospitals and community hospitals. To protect the security and privacy of health data, access mechanisms are designed in multi-tenant data access. Moreover, to increase data interoperability and reduce data heterogeneity, the meaning of the data has been explained semantically by connecting open linked data sources to personal health data. Moreover, to support treatment selection, the authors implemented calculation methods, for instance similarity and process mining. The researchers conducted a use case of antimicrobial drug usage monitoring, which showed that the cloud–MHMS framework provides a considerable elasticity to meet the requirements of out-hospital on-demand healthcare monitoring.

Yang *et al.* (2018) proposed LPAV–IoT, a rule-based adaptive LPA validation model (LPAV), to eliminate irregular uncertainties and to assess the reliability of the IoT-based personal healthcare data. LPAV–IoT enables the validation of IoT environment data to be dynamic standardized empirical analysis workflow. It provides an adapted efficient solution for the validation of IoT-based LPA data with four layers and three modules. The four layers include factors, knowledge, methodologies and actions. The modules or factors, which affect the validity of physical activity, are grouped into device, personal and geographic groups. Each factor determines the strategy of a longitudinal data analysis-based investigation. A set of reliability indicators and uncertainty parameters,

which can be initiated with historical and updated data according to the need of the IoT-based personal health system, are used to represent the validation rules. A case study was conducted on myhealthavatar (MHA), an IoT-enabled healthcare platform with state-of-the-art mobile applications and wearable devices. The results of the case study showed that LPAV–IoT action criteria and validation rules improved the validity of LPA data in the MHA system.

11.1.6 Social Internet of Things (SIoT). The integration between IoT and social networking services such as Facebook, Twitter and microblog has introduced a new paradigm called SIoT. As IoT devices are likely to be connected to any things, even to people themselves, employing IoT devices in collecting information about an individual's activities and location can save the user time and improve social networking services. SIoT applications can automatically collect and integrate personal information and inform the persons when they are in the vicinity of their friends, social events or other activities that may interest them. Furthermore, mobile phones, which support IoT technology, may connect to other mobile phones directly and send contact information when they found compatibility on the pre-defined appointment or friendship profiles. Protection and privacy technologies, used in social networking, can be used in IoT (Li *et al.*, 2015; Whitmore *et al.*, 2015).

11.2 Environment domain

11.2.1 Smart agriculture and smart water. IoT devices can be used in monitoring soil moisture and stem diameter in vineyards to improve and strengthen agriculture work. IoT can maximize production and improve the quality of fruits and vegetables by controlling and maintaining the amount of vitamin in agricultural products and controlling micro-climatic conditions. IoT also can be used in studying weather conditions in the fields to forecast ice formation, drought, snow or wind changes, and controlling humidity and temperature levels to prevent fungus or other microbial contaminants. Smart monitoring of soil can help in making the decisions about agriculture to increase the production of food grains and prevent loss of crops. As drought threatens the environment, water conservation is a major concern, which can take benefits of smart technology. IoT can be used in studying water adequacy in rivers and sea for agricultural and drinkable use, detecting liquid presence outside tanks and pressure variations along pipes and monitoring changes in water levels in rivers, reservoirs and dams. IoT applications already reported in this type are SiSviA, GBROOS and SEMAT (Porkodi and Bhuvaneswari, 2014; Singh *et al.*, 2014).

11.2.2 Animals and breeding. IoT technology can be used in tracking and monitoring farm animals to allow real-time surveillance in a crucial situation such as the outbreaks of infectious diseases. Moreover, in some situations, the countries may give subsidies to farms with cattle, goats and sheep according to the number of animals in a herd. Since the identification of the herd number is difficult, there is always possibility of fraud. To reduce the fraud, a good identification system has to be used. With the help of IoT identification system, the number of herds can be determined and the animal diseases can be surveyed, controlled and prevented. Moreover, vaccination and testing of livestock under official disease control are also possible (Bandyopadhyay and Sen, 2011).

11.2.3 Environmental monitoring. IoT technology can be used in environmental monitoring applications. The ability of a large number of IoT devices to communicate, along with real-time information processing, provides a solid platform for detecting and controlling anomalies that would endanger the human and animal lives. The sensing ability of IoT devices could help in fire detection for environmental safety. A fast response would save human lives, mitigate the damage to the vegetation or property and generally reduce the level of disaster. Sensors also can detect the level of air pollution and smog, for example,

the level of carbon dioxide to deliver that information to health agencies. A wide deployment of miniaturized sensors may allow access to critical areas where the presence of humans might not be possible, for example, in the oceanic abyss, volcanic areas or remote areas. The collected information from sensors can be transferred to a decision point for the purpose of detecting anomalies (Miorandi *et al.*, 2012). In natural disasters, the combination of sensors, self-coordination and simulation can help in predicting the occurrence of fire, flood, earthquake, lightning, landslide or other natural disasters to take appropriate action in advance (Khan *et al.*, 2012).

11.2.4 Recycling. IoT can be used to enhance the efficiency of many important urban and national environmental programs such as vehicle emissions monitoring, recyclable materials collection, packaging resources and electronic parts reusing, e-waste elimination, trash collection and waste containers-level detection (Bandyopadhyay and Sen, 2011; Porkodi and Bhuvaneswari, 2014).

11.3 Transportation domain

11.3.1 Intelligent transportation systems (ITS) or transportation cyber-physical systems (T-CPS). IoT can also be used in traffic monitoring as an important portion of smart city infrastructure. To make the transportation system reliable and smart, normal traffic and highway traffic require sufficient information about the available support and logistics. Any kind of road congestion will eventually lead to fuel and economic loss; therefore, traffic foresight will help in improving the entire system (Singh *et al.*, 2014). ITS utilizes four main components: vehicle subsystem (consists of GPS, RFID reader, OBU and communication), station subsystem (road-side equipment), ITS monitoring center and security subsystem (Al-Fuqaha *et al.*, 2015). The deployment of ITS will make the transportation of people and goods more efficient. For example, by using ITS, the containers can self-weigh and scan themselves; thus, packing containers would become more efficient. The use of IoT technologies to manage passenger luggage at airports will allow automatic tracking and sorting, increase reading rates for each bag and increase security. Vehicle-to-vehicle and vehicle-to-infrastructure communications will greatly contribute to ITS applications in traffic management and vehicle safety services, and they will be fully integrated into the IoT infrastructure (Bandyopadhyay and Sen, 2011). Nevertheless, ITS must be safe enough to prevent any terrorist attack in urban cities. Smart Santander EU project is an example of a prototype implementation (Singh *et al.*, 2014).

11.3.2 Robot taxi. The futuristic robot taxis will organize city traffic in real time, and they will be standardized to minimize congestion at bottlenecks in the city. With or without a human driver, robot taxis will be able to weave in and out of traffic at an optimum speed. The robot taxis would avoid accidents by using vicinity sensors, which would magnetically repel them from other objects on the road. To hail the robot taxis from the side of the street, you can point a mobile phone at them or use hand gestures. By using GPS, the user's location would be automatically tracked and the user would be able to request a taxi to a certain location at a specific time by just pointing at a map. When the robot taxis will not be in use, they will go to "pit stops" where they will automatically arrange themselves on queues for simple maintenance tasks, for instance batteries recharging and cleaning the car, which can be performed by sensors and actuators. The pit stops will communicate with each other to ensure that no overutilization or underutilization takes place (Atzori *et al.*, 2010). There were some research studies on the self-driving car such as Audi, Volvo and Google. Audi, in Nevada, is the first automaker that has a license for a self-driving car. In December 2013, Volvo announced its self-driving car to drive nearly 30 miles on busy roads in Gothenburg, Sweden. Google is also another pioneer in this field (Al-Fuqaha *et al.*, 2015).

11.4 Industry and manufacturing domain

11.4.1 Industrial automation. Industrial automation uses computerized robotic devices to perform manufacturing missions with minimum human intervention. With IoT, the industrial automation will be able to control and monitor operations, functionalities and productivity rate of production machines through the internet. The products are produced rapidly and more accurately by a group of machines based on four elements: transportation, processing, sensing and communication. IoT also raises productivity by analyzing production data and timing and production problem causes. For example, if any machine faces a sudden problem, IoT system will immediately send a maintenance request to the maintenance department to handle and fix the issue (Al-Fuqaha *et al.*, 2015).

For the assembly modeling to be powerful and effective, it has to be decentralized, modularized and automated. Wang, Bi and Da Xu (2014) suggested an automated system for assembly modeling of complex products. As the authors stated, the current computer-aided tools face a bottleneck in dealing with dynamics, complexity and uncertainties of the modern enterprise applications. Moreover, there is a difficulty and complexity of making decisions because of the decentralization of manufacturing enterprises. It has been suggested that the adoption of IoT and cloud computing would overcome those challenges. The researchers proposed an object-oriented model template for assembly planning, a new data mining algorithm for cloud computing and an extended matrix, based on dynamic interference analysis, for automatic identifying of interference-free paths. To demonstrate the effectiveness of the proposed method, the researchers used the assembly modeling for aircraft engines as a case study.

Another topic related to the industrial sector is service-dominant (S-D) logic. The S-D logic is in its early stages of development and few experimental studies exist to test this logic in real business applications. Unlike traditional goods-dominant paradigm, S-D logic has emerged to understand economic exchange and value creation and to provide the correct perspective, assumptions and vocabulary, on which a service-centric logic can be built. The IoT facilitates the conversion of traditional manufacturing companies into a more service-centric business perspective. There is much research on the technical specifications of IoT, but they are few on real business applications (Lai *et al.*, 2017). Lai *et al.* (2017) conducted an in-depth case study of a new service solution of an elevator company, which coveted transition to the S-D by adopting IoT, to examine whether the core technology can combine different ways of thinking when deploying a new industrial service offering. This case study has two sections: the initial stage and a two-year longitudinal case study. The authors focused on the first stage of demonstrating concept development.

11.4.2 Oil and gas industry. In many oil and gas industrial factories, scalable architectures, integrated with sensing/actuating and IoT infrastructure, are used. Theses scalable architectures take into account the possibilities of new identification methods. Petroleum personnel also use IoT in the wireless monitoring of sensitive land and marine areas, tracking of container and drill string components pipes and monitoring and managing of fixed equipment. IoT can help in reducing the number of accidents in the oil and gas factories and supplying dangerous chemical containers. Thus, incidents such as chemical segregation and poor management of processes and storage would be minimized (Bandyopadhyay and Sen, 2011).

11.4.3 Supply chains/logistics. Using RFID and sensor networks in supply chains is not a new thing. They already have long established roles. RFID is often used in tracking products on a supply chain dominated by a particular enterprise, whereas the sensors have been utilized in manufacturing facilities in assembly lines. However, pervasiveness and ubiquity of IoT will enable the use of RFID and WSN beyond the organizational and geographic boundaries, not just at the boundary of a specific enterprise. IoT will provide

more detailed and up-to-date information, which can further improve the efficiency of logistics and supply chain, mitigate the effect of the bullwhip, reduce forgery and improve the tracking of products (Whitmore *et al.*, 2015). The implementation of IoT in retail/supply chain management provides many advantages, for example, monitoring the storage conditions in the supply chain and tracking the products for traceability purposes and payment processing in public transportations, gyms and theme parks, where the payment is based either on duration or location of activity. IoT also will offer many applications in the shop itself such as in-store guidance according to a pre-defined shopping list, fast payment solutions using biometrics on automatic check-out, detection of allergy to a particular product and controlling product recycling in shelves and warehouses to make re-storage operations automatic. SAP future retail center is an example of using IoT in retail (Porkodi and Bhuvaneswari, 2014).

11.4.4 Pharmaceutical Industry. By attaching smart labels to drugs, IoT can be used to track drugs through the supply chain in the pharmaceutical industry. The sensors can also be used to monitor drug status. For example, items that require specific storage conditions can be monitored continuously using IoT to check if storage conditions were violated during transportation. IoT can also help the patients to directly check the medication authenticity assurance, inform the consumers of doses and expiration dates, remind the patients to take their medicine at the appropriate time and monitor the patients' compliance (Bandyopadhyay and Sen, 2011).

11.4.5 Insurance industry. Using IoT, the cars would have electronic recorders that record car speed and acceleration information to deliver to the insurance company. On one hand, the insurance company can save money by participating at a very early stage of the in-the-wind accident. On the other hand, the customers would save their money through discounts on insurance premiums. Other assets such as buildings, machinery, etc., would gain benefits by using IoT technology. IoT will often help in preventing large-scale maintenance operations and minimizing cost by predicting the need for maintenance before the accident (Bandyopadhyay and Sen, 2011).

11.5 Entertainment industry

11.5.1 Enhanced game room. Game rooms and players will be equipped with various devices and sensors for tracking the information of movement, acceleration, humidity, temperature, noise, sound, visual information, location sensitivity, blood pressure and heart rate. By using this information, the game room can measure the excitement and energy level of players to control the game activity depending on the players' situations (Atzori *et al.*, 2010).

11.5.2 Media. IoT technology could be queried for news gathering. The information would be collected from multimedia devices available in a specific location, and the multimedia footage about a certain event would be sent to the news station. Another example is attaching NFC tags to the posters to provide more information by connecting the reader to a URI address that contains more detailed information about the poster (Bandyopadhyay and Sen, 2011).

11.5.3 Music cognition. With the proliferation of mobile devices and the IoT, music recognition as a meaningful mission to promote music has attracted a lot of attention around the world. Creating music score automatically is an important part of music cognition to get rid of a huge amount of music data on the IoT networks or the internet. However, due to the computer deficiency on domain knowledge and cognitive ability, it is difficult for computers, while listening to music, to recognize music melodies or write scores (Lu *et al.*, 2018). Lu *et al.* (2018) proposed a system for music cognition and automatic score writing using machine learning methods and fog computing. The authors provided a case study, and then they compared their platform with others like Pandora and Last.fm. The result showed that the

proposed platform is effective and provides a promising way for the research and application of music cognition. However, as the researchers mentioned, they have to improve the system because the focus of their work was on a single musical instrument. The data sets also can be expanded to make the analysis more representative.

12. IoT challenges and open research issues

There are many challenges that need to be addressed for better achievement of IoT vision (Al-Fuqaha *et al.*, 2015). In order to ensure the adoption and dissemination of IoT, these challenges must be overcome appropriately. This survey introduced some of IoT challenges in the next subsections.

12.1 Architecture

TCP/IP protocol stack has been used in today's network hosts communication for a long time. Nevertheless, the TCP/IP protocol stack is not suitable for IoT. IoT will connect billions of objects, in which everything and everyone would exchange information; thus, much more traffic will exponentially be generated and more data storage will be needed. Therefore, the new proposed architecture for IoT has to address many issues such as scalability, interoperability, reliability, QoS, etc (Khan *et al.*, 2012). Since IoT includes various smart devices and sensors with a wide range of technologies, single reference architecture cannot be used as a blueprint for all applications (Chen *et al.*, 2014). Moreover, the choice of IoT architecture itself is a big challenge that paves the way for the development of a new architecture and the modification of the current architecture (Porkodi and Bhuvaneswari, 2014; Singh *et al.*, 2014).

12.2 TCP

The data transmission on IoT will use the TCP protocol because UDP is not a reliable protocol, which makes it unsuitable for IoT applications. However, TCP has many challenges in IoT, for example, data buffering, in which the data in TCP is required to be stored in a memory buffer at the source and the destination for the purpose of retransmitting lost packets at the source or reordering received packets at the destination. Managing the buffers would be very expensive for resource-constraint devices in terms of storage and energy. As a result, the TCP protocol cannot be used effectively for the end-to-end transmission control in IoT. So far, IoT solutions for TCP have not been proposed; hence, research contributions are required (Atzori *et al.*, 2010).

12.3 Standardization

Many manufacturers provide devices that use their proprietary technologies and services and to which others may not have access. Therefore, the standardization on IoT is very vital to provide better interoperability for all sensor devices and objects (Khan *et al.*, 2012). The aim of IoT standardization is to reduce the access barriers to new service providers and new users. Furthermore, standardization allows products or services to compete better at a higher level. Nevertheless, the fast growth of IoT makes the standardization difficult. Open standards for IoT such as identification standards, communication standards, security standards, etc., are some key factors in the deployment of IoT technologies (Li *et al.*, 2015).

12.4 Middleware

Middleware solutions are needed to support IoT (Perera *et al.*, 2014). The functionalities of IoT middleware solutions are mentioned previously in this survey. Challenges in developing IoT middleware solutions such as interoperability, scalability, abstraction provision, spontaneous interaction, etc., were discussed in detail in Chaqfeh and Mohamed (2012).

12.5 OS challenges

The optimal OS for IoT has not been born yet (Sabri *et al.*, 2017). Many research challenges, most specifically the severe shortage of resources and the requirements of various IoT applications, need to be considered for developing a practical and efficient IoT OS (Musaddiq *et al.*, 2018). Challenges in developing IoT OS and general IoT OS research directions are introduced in Gaur and Tahiliani (2015), Hahm *et al.* (2016), Hicham *et al.* (2017), Musaddiq *et al.* (2018) and Sabri *et al.* (2017).

12.6 Data management challenges

IoT sensors and devices will generate a heterogeneous and huge amount of data. However, the current data center architecture is not prepared to deal with this huge volume of data appropriately. Few organizations will be able to invest in large data storage, enough for storing their IoT collected data. Approximately, more than 2.5 trillion bytes of new data will be recorded every day; therefore, data analysis will play a major role. McKinsey Global Institute estimated that the USA needs between 140,000 and 190,000 additional workers with analytical skills and 1.5m managers and analysts to make business decisions based on big data analysis. Furthermore, there will be a need for a model to integrate data semantic in order to create meaning and knowledge from this big data. AI algorithms and machine learning methods, based on genetic algorithms, evolutionary algorithms, neural networks and other AI techniques, should be applied to extract the knowledge from the redundant data and perform automated decision making (Lee and Lee, 2015; Porkodi and Bhuvaneswari, 2014; Singh *et al.*, 2014).

12.7 Cloud computing

IoT and cloud computing integration would create a smart environment that combines multi-stakeholder services and large-scale support for a massive number of users in a decentralized and reliable way. Nonetheless, this integration needs to cope with constraints like data sources or access devices with unreliable connectivity and limited power. The cloud application platforms need to be enhanced to support the rapid creation of applications. By providing seamless execution of applications, domain-specific programming tools and utilizing the capabilities of multiple dynamic and heterogeneous resources, the cloud platforms would be able to meet the requirements of QoS of the different users (Gubbi *et al.*, 2013).

12.8 Context awareness

Understanding sensor data is one of the main challenges that would face the IoT ecosystem. The research in this area has been strongly invested and funded by the governments, interested groups, companies and research institutes like CERP-IoT. CERP-IoT, funded by the EU, had set a time frame for the development and research on IoT context awareness during 2015–2020 (Perera *et al.*, 2014).

12.9 Quality of service

QoS is easy to be provided in WSNs because of the ability to manage shared wireless media and the limitation of the allocated resource. Research on WSN QoS may be applied to IoT as a short-term solution. Moreover, the research on M2M communication QoS cannot be applied properly to IoT because M2M communication paradigm is different from IoT. IoT objects are various, each of which has its own behavior and characteristics. Another major research area related to QoS is cloud computing QoS, which will lead to the development of a better approach for resource allocation and management and an optimal and controlled way for serving different network traffics. Furthermore, the QoS in IoT cybersecurity is still an unexplored field of research (Lu and Da Xu, 2018; Porkodi and Bhuvaneswari, 2014).

12.10 Security and privacy

IoT improves the productivity of companies and enhances the quality of people's lives. However, since IoT devices are typically wirelessly connected and may be located in public places, IoT will generate a potential surface of attacks for hackers and other cybercriminals. In addition, IoT devices have serious vulnerabilities due to insecure web interfaces, lack of transport encryption, insufficient software protection and inadequate authorization. It has been found that each IoT device has on average 25 holes or risks endangering the whole network. IoT devices are not powerful enough to support robust encryption. Hence, to enable encryption on IoT, efficient key distribution schemes are needed, and the encryption algorithms have to be more efficient and consume less energy. IoT applications will also generate a huge amount of personal data about health, financial status, etc., from which companies will be able to take benefits. Nevertheless, this information, when exposed, will threaten the personal life. The lack of security and privacy solutions will cause resistance to IoT adoption by companies and individuals. Security challenges can be solved by training developers to integrate security solutions, such as intrusion prevention and firewall systems, into their products and encouraging users to use the embedded IoT security features in their devices (Lee and Lee, 2015; Li *et al.*, 2015; Sarkar *et al.*, 2014; Whitmore *et al.*, 2015).

12.11 Managing heterogeneity

IoT devices have different operating conditions, functionalities and resolutions. They are published by different people, authorities or entities. Therefore, seamless integration of this heterogeneity is a major challenge. Heterogeneity should be supported at both the architectural and protocol levels (Bandyopadhyay and Sen, 2011; Miorandi *et al.*, 2012; Sarkar *et al.*, 2014).

12.12 Greening of IoT

IoT ecosystem will cause a significant increase in the network energy consumption due to the increase in data rates, the rapid growth of internet-connected edge devices and the rise in the number of internet-enabled services. Thus, adopting green technologies is important to make IoT devices energy efficient (Khan *et al.*, 2012).

12.13 Interoperability

The actors in IoT applications are various, which can be human or non-human objects. The single actor can play multiple roles depending on the current environment and situations. For the vision of IoT to be visualized, smooth interaction between the various actors is vital (Sarkar *et al.*, 2014). IoT applications will globally serve different applications and industries in the open model. The information interoperability will occur between diverse things, enterprises, industries and regions or countries. The new and open applications or services in IoT should be provided without adding excessive burdens on the market or other operation barriers. Interoperability is important for cross-layer interconnection because the layers are traditionally built with different languages and protocols. A comprehensive approach is required in addressing and resolving the interoperability issues in IoT devices and services in several layers (Chen *et al.*, 2014; Li *et al.*, 2015).

12.14 Scalability

The deployment of a sheer number of miniature devices like sensors and actuators and the huge volume of data produced by them pose a scalability challenge in IoT. Scalability issues arise at different levels, including naming and addressing, data communication and networking, information and knowledge management, and service provisioning and management (Miorandi *et al.*, 2012; Sarkar *et al.*, 2014).

12.15 Challenging issues of searching in IoT

Due to the challenging issues related to the characteristics of IoT, traditional search engines fall short of IoT. IoT search engines issues are identified in Zhang *et al.* (2011). The researchers have discussed research challenges posed by IoT ecosystem such as architecture design of search engines, search locality, scalability and real time. These issues prevent the sharing and processing of information. The authors also explained the trends that should be considered while building IoT search systems.

12.16 Legal/Accountability

IoT will not be dedicated to a single group. In fact, different stakeholders will be involved. The management of IoT global paradigm will pose challenges and accountability. There is a need for a shared administration structure of IoT, which includes all the relevant stakeholders and establishes global accountability and enforcement by issuing punishments (Whitmore *et al.*, 2015).

12.17 Cyber-physical system (CPS)

The research on CPS has made a great progress. Several concepts and technologies are proposed to improve the functionality, adaptability, usability, efficiency, autonomy, reliability and safety of CPS paradigm. Nevertheless, the research is still in its infancy stage due to the complexity of CPS, which comes from the intrinsically distributed nature of CPS systems, the heterogeneity of physical elements (i.e. sensors and actuators), the deficiency of reliability in communications and the variability and the large-scale of the environments. CPS requires systematic innovation and more advanced techniques to solve complex problems in its environments. Moreover, CPS are developed by the cooperation of many engineering disciplines, each of which is utilizing a powerful software tool. To increase CPS efficiency, different technologies and scientific fields have to be integrated and various software tools have to be connected into tool chains, which present a big challenge to the theoretical bases of CPS systems. The future of CPS should use various emerging technologies like SOA, M2M, MAS, big data, cloud computing and augmented reality. New methodologies and technologies have to be designed to meet the higher requirements on performance, real time, flexibility, security and the new applications of CPS in healthcare, manufacturing and business process modeling domains (Chen, 2017; Gürdür and Asplund, 2018; Lu, 2017).

12.18 5G–IoT

5G offers advantages that can meet future IoT requirements. However, it opens up a new set of research challenges on some issues, for example, introducing reliable communication between devices that have various communication technologies. The interoperability between IoT and 5G technology has to consider cybersecurity issues such as data privacy, information transmission management and security protocols and mechanisms (Li *et al.*, 2018; Lu and Da Xu, 2018). In addition, the challenges in 5G–IoT architecture, NFV, D2D communications, etc., have to be addressed. The researchers in Li *et al.* (2018) mentioned these challenges and many more.

12.19 Integrating IoT and cloud computing in Industrial Information Integration Engineering (IIIE)

IIIE has appeared as a new scientific discipline. At first, it was proposed as a new scientific subdiscipline at the meeting of the International Federation for Information Processing Technical Committee for Information systems (TC8) that was held at Guimarães, Portugal, in June 2005. IIIE is a set of fundamental techniques and concepts that facilitate

industrial information integration. IIIE comprises the methods for solving complex issues when developing the infrastructure of information technology on industrial sectors, especially in information integration. As an interdisciplinary subject, IIIE works with various scientific disciplines including computer science, mathematics and almost every engineering discipline. Many techniques have been utilized to examine IIIE such as service-oriented architecture (SOA), enterprise application integration, business process management, workflow management, grid computing and some new technologies like semantics, ontology, modeling and sensors. Although the number of publications in some disciplines is small, it can be seen that research in IIIE has extended from engineering to broader fields, especially business, management and healthcare. There is a close link between IIIE and the IoT and cloud computing. IoT is likely to increase data generation by integrating billions of sensors and multi-core and parallel computers faster than before. The cloud will provide an ideal platform for IIIE. Future research should explore effective approaches for integrating IoT and cloud computing into IIIE (Chen, 2016).

12.20 The Internet of Nano Things (IoNT)

The research on the IoT has grabbed much attention recently. The objects of IoT are interconnected and have embedded computing capabilities that are used to extend the internet to many application domains. As the research and development of IoT devices continue, there are many areas of application that need very tiny non-intrusive hidden objects. The technology is described by Akyildiz and Josep Jornet using graphene-based nano-antennae operating at terahertz frequencies. The IoNT is accomplished by integrating nanoscale sensors into various objects by using nanoscale networks. Each functional task like sensing and actuating is performed in IoNT by a nanomachine whose dimensions may range from 1 to 100 nm. IoNT will allow access to data from places that were impossible to sense or access because of the sensor size. Moreover, it will enable the collection of new medical and environmental data, which will improve the existing knowledge and provide new discoveries and better medical diagnosis. However, the artificial nature of IoNT devices can be harmful as the deployment of Nano Things may have undesirable effects on health or pollution. For the IoNT to be pervasive, more studies have to be conducted on many issues such as the interfacing of IoNT with existing micro-devices, especially in the biomedical and industrial domains. Moreover, there is a need to address key challenges in the fields of electromagnetic channel modeling and network protocols (Akyildiz *et al.*, 2015; Miraz *et al.*, 2015). IoBNT, a novel paradigm introduced by (Akyildiz *et al.*, 2015), represents a paradigm shift in the concept of network engineering and communication. Nonetheless, it faces the challenges of enabling an interface to the electrical domain of the internet and developing efficient and secure technologies for exchanging information, interaction and networking within the biochemical field.

13. Conclusion

The widespread use of miniature devices, which have the ability to communicate and actuate, is the reason that brings IoT vision. New capabilities are made possible as the sensors and actuators function and run smoothly in the background and access rich sources of massive information. The creativity of users in designing new applications will control the evolution of the next-generation mobile phone system. IoT paradigm is ideal for influencing that field by providing new data and computational resources required to create revolutionary applications. IoT devices will enable ambient intelligence and they will be pervasive, ubiquitous and context aware. IoT will improve human being's lives through automation and augmentation. IoT capabilities can save time and money of organizations and people by helping them in making the proper decisions fast. The ground on which the

IoT ecosystem was built is not new. In reality, IoT utilizes existing technologies such as WSNs and RFID along with standards and protocols for M2M communication. IoT has the potential ability to reshape our world by combining existing technologies in a newer way. The research on IoT is conducted in different disciplines such as architecture, OS, network stack protocols, software, application, etc. The question remains whether IoT will be a permanent technology, fail to be achieved, or be the starting point of another model. Time alone will eventually answer these questions.

References

Aazam, M. and Huh, E.-N. (2015), "Dynamic resource provisioning through Fog micro datacenter", *2015 IEEE International Conference on Pervasive Computing and Communication Workshops*, pp. 105-110.

Aazam, M., Khan, I., Alsaffar, A.A. and Huh, E.-N. (2014), "Cloud of Things: integrating Internet of Things and cloud computing and the issues involved", *2014 11th International Bhurban Conference on Applied Sciences and Technology, IEEE*, pp. 414-419.

Aazam, M., St-Hilaire, M., Lung, C.-H. and Lambadaris, I. (2016), "MeFoRE: QoE based resource estimation at Fog to enhance QoS in IoT", *2016 23rd International Conference on Telecommunications (ICT), IEEE*, pp. 1-5.

Akyildiz, I.F., Pierobon, M., Balasubramaniam, S. and Koucheryavy, Y. (2015), "The Internet of Bio-Nano Things", *IEEE Communications Magazine*, Vol. 53 No. 3, pp. 32-40, available at: https://doi.org/10.1109/MCOM.2015.7060516

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M. and Ayyash, M. (2015), "Internet of Things: a survey on enabling technologies, protocols, and applications", *IEEE Communications Surveys & Tutorials*, Vol. 17 No. 4, pp. 2347-2376.

Al-Sarawi, S., Anbar, M., Alieyan, K. and Alzubaidi, M. (2017), "Internet of Things (IoT) communication protocols", *2017 8th International Conference on Information Technology, IEEE*, pp. 685-690.

Amjad, M., Sharif, M., Afzal, M.K. and Kim, S.W. (2016), "TinyOS-new trends, comparative views, and supported sensing applications: a review", *IEEE Sensors Journal*, Vol. 16 No. 9, pp. 2865-2889.

Asghar, M.H., Negi, A. and Mohammadzadeh, N. (2015), "Principle application and vision in Internet of Things (IoT)", *2015 International Conference on Computing, Communication & Automation (ICCCA), IEEE*, pp. 427-431.

Atzori, L., Iera, A. and Morabito, G. (2010), "The Internet of Things: a survey", *Computer Networks*, Vol. 54 No. 15, pp. 2787-2805.

Awan, I. and Younas, M. (2013), "Towards QoS in Internet of Things for delay sensitive information", in Matera, M. and Rossi, G. (Eds), *International Conference on Mobile Web and Information Systems, Trends in Mobile Web Information Systems, MobiWIS 2013. Communications in Computer and Information Science*, Vol. 183, Springer, Cham, pp. 86-94.

Azzarà, A., Bocchino, S., Pagano, P., Pellerano, G. and Petracca, M. (2013), "Middleware solutions in WSN: the IoT oriented approach in the ICSI project", *2013 21st International Conference on Software, Telecommunications and Computer Networks*, pp. 1-6, available at: https://doi.org/10.1109/SoftCOM.2013.6671886

Babu, S.M., Lakshmi, A.J. and Rao, B.T. (2015), "A study on cloud based Internet of Things: CloudIoT", *2015 Global Conference on Communication Technologies (GCCT), IEEE*, pp. 60-65.

Baccelli, E., Gündoğan, C., Hahm, O., Kietzmann, P., Lenders, M.S., Petersen, H., Schleiser, K., Schmidt, T.C. and Wählisch, M. (2018), "RIOT: an open source operating system for low-end embedded devices in the IoT", *IEEE Internet of Things Journal*, Vol. 5 No. 6, pp. 4428-4440.

Bandyopadhyay, D. and Sen, J. (2011), "Internet of Things: applications and challenges in technology and standardization", *Wireless Personal Communications*, Vol. 58 No. 1, pp. 49-69.

Bin, S., Yuan, L. and Xiaoyi, W. (2010), "Research on data mining models for the Internet of Things", *2010 International Conference on Image Analysis and Signal Processing, IEEE*, pp. 127-132.

Botta, A., De Donato, W., Persico, V. and Pescapé, A. (2014), "On the integration of cloud computing and Internet of Things", *2014 International Conference on Future Internet of Things and Cloud, IEEE*, pp. 23-30.

Brogi, A. and Forti, S. (2017), "QoS-aware deployment of IoT applications through the fog", *IEEE Internet of Things Journal*, Vol. 4 No. 5, pp. 1185-1192.

Cantoni, V., Lombardi, L. and Lombardi, P. (2006), "Challenges for data mining in distributed sensor networks", *18th International Conference on Pattern Recognition, IEEE*, pp. 1000-1007.

Casado, L. and Tsigas, P. (2009), "ContikiSec: a secure network layer for wireless sensor networks under the Contiki Operating System", in Jøsang, A., Maseng, T. and Knapskog, S.J. (Eds), *Nordic Conference on Secure IT Systems, Identity and Privacy in the Internet Age, NordSec 2009, Lecture Notes in Computer Science*, Vol. 5838, Springer, Berlin and Heidelberg, pp. 133-147.

Chaqfeh, M.A. and Mohamed, N. (2012), "Challenges in middleware solutions for the Internet of Things", *2012 International Conference on Collaboration Technologies and Systems, IEEE*, pp. 21-26.

Chen, H. (2017), "Theoretical foundations for cyber-physical systems: a literature review", *Journal of Industrial Integration and Management*, Vol. 2 No. 3, p. 1750013.

Chen, S., Xu, H., Liu, D., Hu, B. and Wang, H. (2014), "A vision of IoT: applications, challenges, and opportunities with china perspective", *IEEE Internet of Things Journal*, Vol. 1 No. 4, pp. 349-359.

Chen, Y. (2016), "Industrial information integration – a literature review 2006–2015", *Journal of Industrial Information Integration*, Vol. 2, pp. 30-64.

Chiang, M. and Zhang, T. (2016), "Fog and IoT: an overview of research opportunities", *IEEE Internet of Things Journal*, Vol. 3 No. 6, pp. 854-864.

Chung, B., Kim, J. and Jeon, Y. (2016), "On-demand security configuration for IoT devices", *2016 International Conference on Information and Communication Technology Convergence*, pp. 1082-1084, available at: https://doi.org/10.1109/ICTC.2016.7763373

Cooprider, N., Archer, W., Eide, E., Gay, D. and Regehr, J. (2007), "Efficient memory safety for TinyOS", *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems, ACM*, pp. 205-218.

Dai, H., Neufeld, M. and Han, R. (2004), "ELF: an efficient log-structured flash file system for micro sensor nodes", *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, ACM*, pp. 176-187.

Da Xu, L., He, W. and Li, S. (2014), "Internet of Things in industries: a survey", *IEEE Transactions on Industrial Informatics*, Vol. 10 No. 4, pp. 2233-2243.

de Oliveira, B.T., Alves, R.C.A. and Margi, C.B. (2015), "Software-defined wireless sensor networks and Internet of Things standardization synergism", *2015 IEEE Conference on Standards for Communications and Networking*, pp. 60-65.

Dolui, K. and Datta, S.K. (2017), "Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing", *2017 Global Internet of Things Summit (GIoTS)*, pp. 1-6, available at: https://doi.org/10.1109/GIOTS.2017.8016213

Dorri, A., Kanhere, S.S. and Jurdak, R. (2016), "Blockchain in Internet of Things: challenges and solutions", pp. 1-13.

Dorri, A., Kanhere, S.S., Jurdak, R. and Gauravaram, P. (2017), "Blockchain for IoT security and privacy: the case study of a smart home", *2017 IEEE International Conference on Pervasive Computing and Communications Workshops*, pp. 618-623.

Dunkels, A., Gronvall, B. and Voigt, T. (2004), "Contiki – a lightweight and flexible operating system for tiny networked sensors", *29th Annual IEEE International Conference on Local Computer Networks*, pp. 455-462.

Fosstrak.github (2018), "Fosstrak – Welcome", available at: http://fosstrak.github.io/ (accessed August 20, 2018).

Gaur, P. and Tahiliani, M.P. (2015), "Operating systems for IoT devices: a critical survey", *2015 IEEE Region 10 Symposium*, pp. 33-36.

Giang, N.K., Blackstock, M., Lea, R. and Leung, V.C. (2015), "Developing IoT applications in the fog: a distributed dataflow approach", *2015 5th International Conference on the Internet of Things (IOT), IEEE*, pp. 155-162.

Gubbi, J., Buyya, R., Marusic, S. and Palaniswami, M. (2013), "Internet of Things (IoT): a vision, architectural elements, and future directions", *Future Generation Computer Systems*, Vol. 29 No. 7, pp. 1645-1660.

Gürdür, D. and Asplund, F. (2018), "A systematic review to merge discourses: Interoperability, integration and cyber-physical systems", *Journal of Industrial Information Integration*, Vol. 9 No. 7, pp. 14-23.

Hahm, O., Baccelli, E., Petersen, H. and Tsiftes, N. (2016), "Operating systems for low-end devices in the Internet of Things: a survey", *IEEE Internet of Things Journal*, Vol. 3 No. 5, pp. 720-734.

Hicham, A., Sabri, A., Jeghal, A. and Tairi, H. (2017), "A comparative study between operating systems (Os) for the Internet of Things (IoT)", *Transactions on Machine Learning and Artificial Intelligence*, Vol. 5 No. 4, doi: 10.14738/tmlai.54.3192.

IPv6 (2018), "Using RFID and IPv6 | IPv6.com", available at: www.ipv6.com/applications/using-rfid-ipv6/ (accessed August 21, 2018).

Ishaq, I., Carels, D., Teklemariam, G., Hoebeke, J., Abeele, F., Poorter, E., Moerman, I. and Demeester, P. (2013), "IETF standardization in the field of the Internet of Things (IoT): a survey", *Journal of Sensor and Actuator Networks*, Vol. 2 No. 2, pp. 235-287.

Jeon, J.H., Kim, K. and Kim, J. (2018), "Block chain based data security enhanced IoT server platform", *2018 International Conference on Information Networking (ICOIN)*, pp. 941-944, available at: https://doi.org/10.1109/ICOIN.2018.8343262

Jeong, Y., Joo, H., Hong, G., Shin, D. and Lee, S. (2015), "AVIoT: web-based interactive authoring and visualization of indoor Internet of Things", *IEEE Transactions on Consumer Electronics*, Vol. 61 No. 3, pp. 295-301, available at: https://doi.org/10.1109/TCE.2015.7298088

Kafle, V.P., Fukushima, Y. and Harai, H. (2016), "Internet of Things standardization in ITU and prospective networking technologies", *IEEE Communications Magazine*, Vol. 54 No. 9, pp. 43-49.

Karagiannis, V., Chatzimisios, P., Vazquez-Gallego, F. and Alonso-Zarate, J. (2015), "A survey on application layer protocols for the Internet of Things", *Transaction on IoT and Cloud Computing*, Vol. 3, pp. 11-17.

Karlof, C., Sastry, N. and Wagner, D. (2004), "TinySec: a link layer security architecture for wireless sensor networks", *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, ACM*, pp. 162-175.

Keller, T. (2011), "Mining the Internet of Things – detection of false-positive RFID tag reads using low-level reader data", PhD thesis, University of St. Gallen, School of Management.

Khan, R., Khan, S.U., Zaheer, R. and Khan, S. (2012), "Future internet: the Internet of Things Architecture, possible applications and key challenges", *2012 10th International Conference on Frontiers of Information Technology (FIT), IEEE*, pp. 257-260.

Lai, C.T., Jackson, P.R. and Jiang, W. (2017), "Shifting paradigm to service-dominant logic via Internet-of-Things with applications in the elevators industry", *Journal of Management Analytics*, Vol. 4 No. 1, pp. 35-54.

Lee, I. and Lee, K. (2015), "The Internet of Things (IoT): applications, investments, and challenges for enterprises", *Business Horizons*, Vol. 58 No. 4, pp. 431-440.

Lee, S.-D., Shin, M.-K. and Kim, H.-J. (2007), "EPC vs IPv6 mapping mechanism", *The 9th International Conference on Advanced Communication Technology, IEEE*, pp. 1243-1245.

Li, L., Li, S. and Zhao, S. (2014), "QoS-aware scheduling of services-oriented Internet of Things", *IEEE Transactions on Industrial Informatics*, Vol. 10 No. 2, pp. 1497-1505.

Li, S., Da Xu, L. and Wang, X. (2013), "Compressed sensing signal and data acquisition in wireless sensor networks and Internet of Things", *IEEE Transactions on Industrial Informatics*, Vol. 9 No. 4, pp. 2177-2186.

Li, S., Da Xu, L. and Zhao, S. (2015), "The Internet of Things: a survey", *Information Systems Frontiers*, Vol. 17 No. 2, pp. 243-259.

Li, S., Tryfonas, T. and Li, H. (2016), "The Internet of Things: a security point of view", *Internet Research*, Vol. 26 No. 2, pp. 337-359.

Li, S., Xu, L.D. and Zhao, S. (2018), "5G Internet of Things: a survey", *Journal of Industrial Information Integration*, Vol. 10, pp. 1-9, available at: https://doi.org/10.1016/j.jii.2018.01.005

Lindgren, P., Mäkitaavola, H., Eriksson, J., Eliasson and J. (2012), "Leveraging TinyOS for integration in process automation and control systems", *IECON 2012 – 38th Annual Conference on IEEE Industrial Electronics Society*, pp. 5779-5785.

Liu, C., Zhang, Y. and Zhang, H. (2013), "A novel approach to IoT security based on immunology", *2013 Ninth International Conference on Computational Intelligence and Security, IEEE*, pp. 771-775, available at: https://doi.org/10.1109/CIS.2013.168

Liu, F., Tan, C.-W., Lim, E.T. and Choi, B. (2017), "Traversing knowledge networks: an algorithmic historiography of extant literature on the Internet of Things (IoT)", *Journal of Management Analytics*, Vol. 4 No. 1, pp. 3-34.

Lu, L., Xu, L., Xu, B., Li, G. and Cai, H. (2018), "Fog computing approach for music cognition system based on machine learning algorithm", *IEEE Transactions on Computational Social Systems*, Vol. 5 No. 4, pp. 1142-1151.

Lu, Y. (2017), "Cyber physical system (CPS)-based Industry 4.0: a survey", *Journal of Industrial Integration and Management*, Vol. 2 No. 3, p. 1750014.

Lu, Y. and Da Xu, L. (2018), "Internet of Things (IoT) cybersecurity research: a review of current research topics", *IEEE Internet of Things Journal*, Vol. 6 No. 2, pp. 2103-2115.

Marjani, M., Nasaruddin, F., Gani, A., Karim, A., Hashem, I.A.T., Siddiqa, A. and Yaqoob, I. (2017), "Big IoT data analytics: architecture, opportunities, and open research challenges", *IEEE Access*, Vol. 5, pp. 5247-5261.

Masciari and E. (2007), "A Framework for outlier mining in RFID data", *11th International Database Engineering and Applications Symposium, IEEE*, pp. 263-267.

Miorandi, D., Sicari, S., De Pellegrini, F. and Chlamtac, I. (2012), "Internet of Things: vision, applications and research challenges", *Ad Hoc Networks*, Vol. 10 No. 7, pp. 1497-1516.

Miraz, M.H., Ali, M., Excell, P.S. and Picking, R. (2015), "A review on Internet of Things (IoT), Internet of Everything (IoE) and Internet of Nano Things (IoNT)", *2015 Internet Technologies and Applications*, pp. 219-224, available at: https://doi.org/10.1109/ITechA.2015.7317398

Mun, D.-H., Le Dinh, M. and Kwon, Y.-W. (2016), "An assessment of Internet of Things protocols for resource-constrained applications", *2016 IEEE 40th Annual Computer Software and Applications Conference*, pp. 555-560.

Munir, A., Kansakar, P. and Khan, S.U. (2017), "IFCIoT: Integrated Fog Cloud IoT: a novel architectural paradigm for the future Internet of Things", *IEEE Consumer Electronics Magazine*, Vol. 6 No. 3, pp. 74-82.

Musaddiq, A., Zikria, Y.B., Hahm, O., Yu, H., Bashir, A.K. and Kim, S.W. (2018), "A survey on resource management in IoT operating systems", *IEEE Access*, Vol. 6, pp. 8459-8482.

Necula, G.C., Condit, J., Harren, M., McPeak, S. and Weimer, W. (2005), "CCured: type-safe retrofitting of legacy software", *ACM Transactions on Programming Languages and Systems*, Vol. 27 No. 3, pp. 477-526.

Ngu, A.H., Gutierrez, M., Metsis, V., Nepal, S. and Sheng, Q.Z. (2017), "IoT middleware: a survey on issues and enabling technologies", *IEEE Internet of Things Journal*, Vol. 4 No. 1, pp. 1-20.

Oh, S. and Kim, Y. (2017), "Development of IoT security component for interoperability", *2017 13th International Computer Engineering Conference*, pp. 41-44, available at: https://doi.org/10.1109/ICENCO.2017.8289760

Perera, C., Zaslavsky, A., Christen, P. and Georgakopoulos, D. (2014), "Context aware computing for the Internet of Things: a survey", *IEEE Communications Surveys & Tutorials*, Vol. 16 No. 1, pp. 414-454.

Pérez, S., Martínez, J.A., Skarmeta, A.F., Mateus, M., Almeida, B. and Maló, P. (2016), "ARMOUR: Large-scale experiments for IoT security amp; trust", *2016 IEEE 3rd World Forum on Internet of Things*, pp. 553-558, available at: https://doi.org/10.1109/WF-IoT.2016.7845504

Porkodi, R. and Bhuvaneswari, V. (2014), "The Internet of Things (IOT) applications and communication enabling technology standards: an overview", *2014 International Conference on Intelligent Computing Applications, IEEE*, pp. 324-329.

Ray, P. (2016), "A survey of IoT cloud platforms", *Future Computing and Informatics Journal*, Vol. 1 Nos 1-2, pp. 35-46, available at: https://doi.org/doi.org/10.1016/j.fcij.2017.02.001

Razzaque, M.A., Milojevic-Jevric, M., Palade, A. and Clarke, S. (2016), "Middleware for Internet of Things: a survey", *IEEE Internet of Things Journal*, Vol. 3 No. 1, pp. 70-95.

Regehr, J., Cooprider, N., Archer, W. and Eide, E. (2006), "Memory safety and untrusted extensions for TinyOS", CiteSeer.

Riot-os (2018), "File systems: RIOT documentation", available at: https://riot-os.org/api/group__sys__fs.html (accessed October 25, 2018).

Sabri, C., Kriaa, L. and Azzouz, S.L. (2017), "Comparison of IoT constrained devices operating systems: a survey", *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications*, pp. 369-375.

Sarkar, C., Nambi, S.A.U., Prasad, R.V. and Rahim, A. (2014), "A scalable distributed architecture towards unifying IoT applications", *2014 IEEE World Forum on Internet of Things*, pp. 508-513.

Sarna, S.K. and Zaveri, M. (2010), "EATT: Energy aware target tracking for wireless sensor networks using TinyOS", *2010 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, pp. 187-191.

Sethi, P. and Sarangi, S.R. (2017), "Internet of Things: architectures, protocols, and applications", *Journal of Electrical and Computer Engineering*, Vol. 2017, 25pp., available at: https://doi.org/10.1155/2017/9324035

Singh, D., Tripathi, G. and Jara, A.J. (2014), "A survey of Internet-of-Things: future vision, architecture, challenges and services", *2014 IEEE World Forum on Internet of Things*, pp. 287-292.

Singh, M., Singh, A. and Kim, S. (2018), "Blockchain: a game changer for securing IoT data", *2018 IEEE 4th World Forum on Internet of Things*, pp. 51-55, available at: https://doi.org/10.1109/WF-IoT.2018.8355182

Triantafyllou, A., Sarigiannidis, P. and Lagkas, T.D. (2018), "Network protocols, schemes, and mechanisms for Internet of Things (IoT): features, open challenges, and trends", *Wireless Communications and Mobile Computing*, Vol. 2018, 24pp., available at: https://doi.org/10.1155/2018/5349894

Tsai, C.-W., Lai, C.-F., Chiang, M.-C. and Yang, L.T. (2014), "Data mining for Internet of Things: a survey", *IEEE Communications Surveys and Tutorials*, Vol. 16 No. 1, pp. 77-97.

Tsiftes, N., Dunkels, A., He, Z. and Voigt, T. (2009), "Enabling large-scale storage in sensor networks with the coffee file system", *International Conference on Information Processing in Sensor Networks, IEEE*, pp. 349-360.

Vithya, G. and Vinayagasundaram, B. (2014), "Qos by priority routing in Internet of Things", *Research Journal of Applied Sciences, Engineering and Technology*, Vol. 8 No. 21, pp. 2154-2160.

Wang, C., Bi, Z. and Da Xu, L. (2014), "IoT and cloud computing in automation of assembly modeling systems", *IEEE Transactions on Industrial Informatics*, Vol. 10 No. 2, pp. 1426-1434.

Wang, L., Da Xu, , L., Bi, Z. and Xu, Y. (2014), "Data cleaning for RFID and WSN integration", *IEEE Transactions on Industrial Informatics*, Vol. 10 No. 1, pp. 408-418.

Whitmore, A., Agarwal, A. and Da Xu, L. (2015), "The Internet of Things – a survey of topics and trends", *Information Systems Frontiers*, Vol. 17 No. 2, pp. 261-274.

Xu, B., Li, L., Hu, D., Wu, B., Ye, C. and Cai, H. (2018), "Healthcare data analysis system for regional medical union in smart city", *Journal of Management Analytics*, Vol. 5 No. 4, pp. 334-349.

Xu, B., Xu, L., Cai, H., Jiang, L., Luo, Y. and Gu, Y. (2017), "The design of an m-health monitoring system based on a cloud computing platform", *Enterprise Information Systems*, Vol. 11 No. 1, pp. 17-36.

Xu, X., Huang, S., Chen, Y., Browny, K., Halilovicy, I. and Lu, W. (2014), "TSAaaS: time series analytics as a service on IoT", *2014 IEEE International Conference on Web Services*, pp. 249-256.

Yan, H., Xu, L.D., Bi, Z., Pang, Z., Zhang, J. and Chen, Y. (2015), "An emerging technology–wearable wireless sensor networks with applications in human health condition monitoring", *Journal of Management Analytics*, Vol. 2 No. 2, pp. 121-137.

Yang, P. and Xu, L. (2018), "The Internet of Things (IoT): informatics methods for IoT-enabled health care", *Journal of Biomedical Informatics*, Vol. 87, pp. 154-156.

Yang, P., Stankevicius, D., Marozas, V., Deng, Z., Liu, E., Lukosevicius, A., Dong, F., Xu, L. and Min, G. (2018), "Lifelogging data validation model for Internet of Things enabled personalized healthcare", *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol. 48, pp. 50-64.

Yang, Z., Yue, Y., Yang, Y., Peng, Y., Wang, X. and Liu, W. (2011), "Study and application on the architecture and key technologies for IOT", *2011 International Conference on Multimedia Technology (ICMT)*, pp. 747-751.

Yoon, D.G., Lee, D.H., Seo, C.H. and Choi, S.G. (2008), "RFID networking mechanism using address management agent", *Fourth International Conference on Networked Computing and Advanced Information Management, IEEE*, pp. 617-622, available at: https://doi.org/10.1109/NCM.2008.156

Yoon, S. and Kim, J. (2017), "Remote security management server for IoT devices", *2017 International Conference on Information and Communication Technology Convergence*, pp. 1162-1164, available at: https://doi.org/10.1109/ICTC.2017.8190885

Yuehong, Y.I.N., Zeng, Y., Chen, X. and Fan, Y. (2016), "The Internet of Things in healthcare: an overview", *Journal of Industrial Information Integration*, Vol. 1, pp. 3-13.

Zhang, D., Yang, L.T. and Huang, H. (2011), "Searching in Internet of Things: vision and challenges", *2011 IEEE Ninth International Symposium on Parallel and Distributed Processing with Applications*, pp. 201-206, available at: https://doi.org/10.1109/ISPA.2011.53

Zhang, M., Sun, F. and Cheng, X. (2012), "Architecture of Internet of Things and its key technology integration based-on RFID", *2012 Fifth International Symposium on Computational Intelligence and Design, IEEE*, pp. 294-297.

Zheng, X., Martin, P., Brohman, K. and Da Xu, L. (2014a), "CLOUDQUAL: a quality model for cloud services", *IEEE Transactions on Industrial Informatics*, Vol. 10 No. 2, pp. 1527-1536.

Zheng, X., Martin, P., Brohman, K. and Da Xu, L. (2014b), "Cloud service negotiation in Internet of Things environment: a mixed approach", *IEEE Transactions on Industrial Informatics*, Vol. 10 No. 2, pp. 1506-1515.

Corresponding author
Elham Ali Shammar can be contacted at: eshammar@gmail.com