Hindawi
Wireless Communications and Mobile Computing
Volume 2019, Article ID 9538016, 1 page
https://doi.org/10.1155/2019/9538016

Retraction

Retracted: A Survey on Management Frameworks and Open Challenges in IoT

Wireless Communications and Mobile Computing

Received 31 January 2019; Accepted 31 January 2019; Published 7 March 2019

Wireless Communications and Mobile Computing has retracted the article titled "A Survey on Management Frameworks and Open Challenges in IoT" [1]. The article was found to not belong to the author, Dr. Farzad Kiani. Other researchers, Sufian Hameed and Faraz Idris, say the research is theirs and was published without their knowledge or permission.

Dr. Kiani said he had received the article in good faith from a now-deceased colleague, he did not intend to violate anyone's rights, he sincerely apologizes to the researchers, and he agreed to retraction.

References

[1] F. Kiani, "A survey on management frameworks and open challenges in IoT," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 9857026, 33 pages, 2018.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 9857026, 33 pages
https://doi.org/10.1155/2018/9857026

Review Article

A Survey on Management Frameworks and Open Challenges in IoT

Farzad Kiani

Computer Engineering Department, Engineering and Natural Sciences Faculty, Istanbul Sabahattin Zaim University, 34303 Istanbul, Turkey

Correspondence should be addressed to Farzad Kiani; farzad.kiani@izu.edu.tr

Received 14 April 2018; Revised 31 May 2018; Accepted 5 July 2018; Published 1 August 2018

Academic Editor: Bernard Cousin

The new era of technology is a rollback in which massive information comes from interconnected devices. These types of things that have Internet connectivity are called the Internet of Things (IoT). IoT and its applications have enabled novel service provisioning, whereas at the same time it has raised complex challenges of managing such network. Resource limitation such as computational power and energy limitation defines the characteristics of IoT that require a novel design of management framework. Designing a management framework that meets the IoT requirements is a daunting task. Recently, Software-Defined Networking (SDN) paradigm has eased the management of traditional Internet by provisioning centralized view of the network. This makes SDN a potential candidate for managing IoT. However, traditional design of the SDN paradigm needs to be redefined to adapt it for the IoT. Hence, in this paper, we have discussed trends and research of management architecture for IoT based on Software-Defined Networking principles. In addition, we discuss non-SDN-based approaches. In this paper, we compare all of the efforts in the last five years in addressing issues of IoT management that involves security service provisioning, fault tolerance, energy management, and load balancing. We also discuss future research directions that evolved from our detailed survey and taxonomy of the Software-Defined IoT (SDIoT) management frameworks.

1. Introduction

The emerging trends in embedded technologies and Internet have enabled objects surrounding us to be interconnected with each other. The increase in connected devices and device to device communication has been referred to in any way such as the Internet of Thing (IoT), Internet of Everything (IoE), Internet of Anything (IoA), Machine-to-Machine (M2M), and Industrial Internet of Things (IIoT). The common aspect between all these terms is the connection of new kinds of objects to the Internet in order to build a connected world. We envision a future where Internet-based devices will be invisibly implanted around us and they will be generating enormous amounts of data. This data will be gathered, processed, and presented in an understandable form. The collected data will generate big data and the special software systems will give information to their users by analyzing this data in servers' side. IoT model involves numerous actors, which includes mobile operators, software developers, and access technology providers. Additionally, IoT has extended domain in heterogeneous applications such as medicine, manufacturing, agriculture, and utility management. IoT can be seen as the next-generation interconnection paradigm that will enable connectivity among people's devices and machines to enable actions without human intervention. The success of the IoT world requires a merger of heterogeneous communication infrastructure. This has led to the design of smart gateways to connect IoT devices with the traditional Internet. Most recently, efforts are directed to interconnect IoT infrastructure and cloud computing, which supplements the potentials of IoT. The IoT network is different from the traditional networks and is characterized by limited resources such as power and energy. This poses challenges for designing solutions for IoT, as solutions pertaining to the traditional Internet are inapplicable. IoT network requires novel solutions to manage the resources over the network. Typically, the architecture of IoT is divided into three basic layers application layer, network layer, and perception layer.

Software-Defined Networking (SDN) is an umbrella term encompassing several kinds of network technology aimed at making the network as agile and flexible as the virtualized server and storage infrastructure of the modern data center. The goal of SDN is to allow network engineers and administrators to respond quickly to changing business requirements. In a software-defined network, a network administrator can shape traffic from a centralized control console without having to touch individual switches and can deliver services to wherever they are needed in the network, without regard to what specific devices a server or other hardware components are connected to. The key technologies for SDN implementation are functional separation, network virtualization, and automation through programmability. It can be usable in various concepts, as one of them is IoT. In recent years, researchers have proposed resource management frameworks to tackle the management problem in IoT networks. Recently, the orientation is moving towards exploiting the potentials of SDN paradigm to manage IoT resources. The SDN is an emerging networking framework, which can unify several network operations through virtualization. In [1], the authors have presented an SDN-based framework for managing traffic and network resources in an IoT environment. Along the same lines, there are a handful of other efforts that suggest taking an SDN-based approach to solving management issues in IoT [2–6].

In this paper, we focus on Software-Defined Networking frameworks in the IoT that is called the Software-Defined Internet of Things (SDIoT). We also examine non-SDN-based methods. Here, our goal is to categorize existing SDIoT frameworks and to provide comprehensive survey of the existing efforts in this area. In addition, we evaluated current work-studies in non-SDN-based frameworks. The implementation of IoT-based frameworks and their management are important topics in this trend. It is not possible to use traditional management mechanisms in IoT networks because IoT has different characteristics. Thanks to the SDN concept and features that have emerged in recent years, researchers have begun using this technique for IoT framework management. The specific features of SDNs are compatible with the IoT-based frameworks and the researchers are beginning to get good and efficient results in these frameworks management. However, SDN-based solutions are not required for framework management in IoT. Thus, in some studies in the literature, researchers have suggested that IoT systems can be managed by developing a non-SDN-based custom API. Of course, some of these studies are also inspired by SDN-based methods. In this paper, the studied methods on IoT framework management are examined in two different categories so one of them is SDN-based methods and the other is non-SDN custom API programmable methods.

For this purpose, we have done comparison of the existing proposals from the year 2011 to 2016 in terms of crucial management issues of IoT, which are security service provisioning, fault tolerance, energy management, and load balancing. Such a study of SDIoT management frameworks and the presented survey is important to understand the critical issues related to this emerging IoT networking paradigm. Our contribution is discussion on the role of SDN systems in the IoT management and current challenges in the IoT. We prepare a general taxonomy of the existing researches in these study area. So, we divided our working area to two fields. One of them is SDN-based IoT management framework and the other category is non-SDN API programmable management frameworks. It is accepted that the literature is categorized in this way in relation to IoT and SDN. We are evaluating various studies in these fields by structurally dividing SDN-based methods into three different categories. Non-SDN-based methods also follow the evaluation method in four parts.

In this paper, their plus and minus are mentioned by explaining all the studies that are discussed. This study wishes might be useful for researchers in this area. To the best of our knowledge, this is the very first extensive survey to review all the current efforts for SDIoT. Therefore, it can be a helpful work to researchers that they want investigate on the IoT and SDN areas. Our method is to present the reader in a certain accepted form by examining valuable works in this field and to explain it in terms of advantages and disadvantages of relevant studies and to compare all these work-studies based on what we have explained according to our target parameters, so, they are fault tolerance, energy management, load balancing, and security management. In summary, we categorize this paper as follows:

(i) Overview of SDN

(ii) Overview of IoT and its management challenges

(iii) Current SDN-based IoT management frameworks

(iv) Current non-SDN-based IoT management frameworks

(v) Comparison of literature methods and evaluation

(vi) Insights for the future IoT network

The following sections of the paper are organized as below. In Section 2, SDN and its working principles are introduced. Section 3 discusses the management challenges of IoT. The SDN-based IoT management framework category is discussed in Section 4. In Section 4 of the paper, many studies in the literature have been carefully analyzed and categorized. The other category is non-SDN API programmable management frameworks that are discussed in Section 5 of the paper. This category is divided into four main headings (software-defined sensor network, software-defined IoT for smart urban sensing, software-defined unified device management for smart environments, and provisioning software-defined IoT cloud systems). Sections 6 and 7 included comparisons, evaluations, conclusions, and future works, respectively.

2. Software-Defined Networking (SDN)

Innovations in network research have led to a new paradigm of Software-Defined Networking (SDN) [7]. SDN eases management of network services by abstracting high-level functionalities. This is achieved by decoupling the control plane (brain that decides where the traffic should be sent) from the data plane (underlying dumb forwarding element

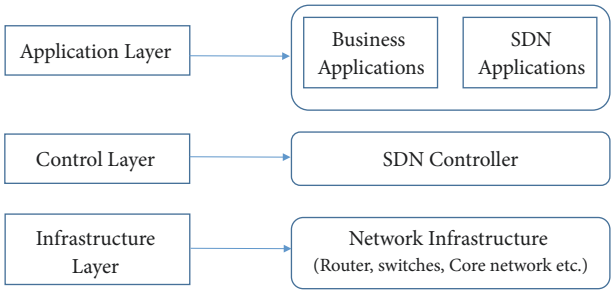


Figure 1: SDN architecture.

that forwards traffic to the selected destination). SDN enabled network devices to make forwarding decisions by communicating with SDN controller, which makes appropriate forwarding decisions. Communication between the controller and network devices takes place by using a protocol. OpenFlow is the most widely used protocol for this purpose. Devices that support OpenFlow are often called OpenFlow switches.

The OpenFlow protocol sends control messages that enable SDN controller to securely connect with the network devices. By connecting with the network devices forwarding instructions can be installed and current state can be read. Flow table can be found in the data plane of the flow switch where each entry in the table has a set of packet fields that are matched and consequently action is performed. Upon reception of a packet by OpenFlow switch, the packet is sent to the controller without any matching in the flow entries. The controller on how the packet is dealt with makes decision. The decision can be made to discard the packet or a flow entry that instructs the switch on how to advance the similar packets in future. Flow entries are populated in the flow tables by defining flow rules in OpenFlow. The rules are changed vigorously to transform heterogeneous network changes. Hence, SDN paradigm has potential to solve network and resource management in the heterogeneous network. Recent efforts have led to the standardization of SDN paradigm by the research community. This makes it possible to adopt SDN paradigm for various kinds of networks such as wired, wireless, and lower power networks.

SDN architecture, as shown in Figure 1, is segregated into three layers. The lowest layer often called infrastructure layer consists of network devices. Typically, these devices are dumb and their control function is concentrated in the control layer, which generally can have one or more than one controller to manage the devices in the infrastructure layer. Infrastructure layer interacts with the control layer through an interface of control-data plane usually called southbound interface. The controllers in providing services of the control layer are for the use of the above layer, which is an application layer through an API called northbound interface. Using the application layer, the application developers and network operators utilize the network by using programmable interfaces fulfilling business requirements that include traffic engineering, access control, QoS, energy management, and route management. Such paradigm annihilates conventional networks prone to errors due to manual configuration.

Communication over OpenFlow happens in a secure manner using a secure channel. The OpenFlow architecture is shown in Figure 2. The controller adds and removes flow entries from the related flow table by using a secure channel. OpenFlow switch anatomy contains secure channel, one or more flow tables, and group tables. Groups are organized into multiple flow entries that route to a single identifier by defining a node in the network. Such abstraction allows common output actions applied to flow entries that can be changed efficiently. Incoming packets to OpenFlow switch matches against multiple flow tables until a match is found and set of actions applicable for that particular flow entry are performed.

3. Management Challenges of IoT

Recent efforts of integrating Wireless Sensor Network (WSN) and Internet have led to incorporate WSN in IoT [8]. Hence, it is no doubt that WSN forms an essential component of IoT [9]. Most of the management challenges of WSN are inherited from [10]. The four most important parameters in these systems, which have specific characteristics, are generally regarded as critical parameters [11–14]. They are fault tolerance, energy management, load balancing, and security management. Even if the device on the network is faulty due to the malfunctions, the service should be continued. Due to resource-constrained nature of the IoT devices, especially energy and memory constraint in sensor devices, conserving energy in the network is an important concern. Hence, energy management in IoT deals with conserving energy within IoT network. Imbalance traffic in the network cause to wastage of system resources especially these are inherently resource constraints. Load balancing the traffic within IoT network results in effective resource utilization within the IoT network.

Security is crucial in IoT. Because the applications tend to collect data from the environment and send to central servers for analysis, it is important to make sure that the data is secure [15]. Therefore, the privacy is a sensitive issue. Besides privacy, there are several other security concerns [16, 17] such as routing attacks within the network that disrupts the IoT services. Hence, security management in IoT is a good research topic. Unique challenges of IoT make conventional network management schemes inapplicable. Devices in IoT network are constrained in resources. IoT network is characterized by high fault rates due to shortages in energy and connectivity interruptions. Therefore, the main issue in IoT for effective network configuration is monitoring and managing nodes communication. Management solutions for IoT should be designed in a manner that provides various management functions like integrating configuration, security operations, and administration of devices and services of IoT. Following set of functions should be provided by management solution for IoT. All the work-studies to be described in the other sections will be compared with each other according to the following four parameters.

3.1. Fault Tolerance. The abundance of data is inherent in IoT networks. The problems in data transmission in the

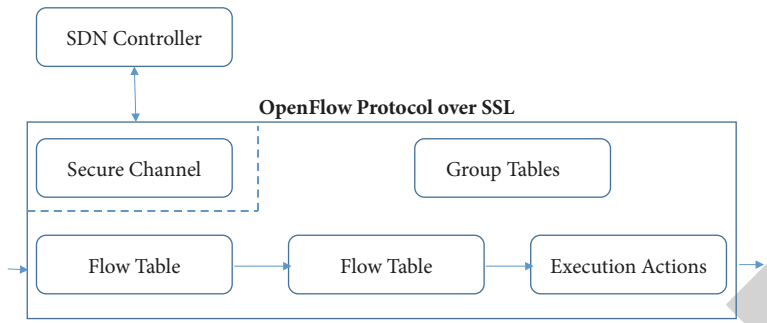


Figure 2: OpenFlow architecture.

IoT nodes with limited resources are normally subject. The relevant nodes may be failed for a variety of reasons; for example, the energy may be distorted, disrupted by natural disasters, and so on. Additionally, the devices may propagate inaccurate readings due to some unexpected influence on the sensing components of the devices. In addition, inherently ad hoc wireless network links have tendency of failures that cause partitions in the network and vigorous changes in the network topologies. Delivered packets are corrupted because of fallacious nature of communication. Not only failures of link cause packet loss, it may also result due to congestion in the network. All of the fault scenarios discussed are exacerbated due to multihop communications in nature of IoT. Summary of efforts in this direction can be found in [18].

3.2. Energy Management. Energy is a scarce resource in IoT especially as the nodes are deployed in a distant region that results in depleting of available energy and it becomes impossible to substitute for energy. In order to avoid energy scarcity in the network, balanced energy management between supply and load is required. Regulating data traffic from the devices by techniques such as duty cycling, which schedules the sleep and wake-up modes of a device, has been investigated [19]. In order to desire smooth operation of IoT network, an introduced management solution must focus on the energy issue absolutely.

3.3. Load Balancing. A suitable load balancing method, which is used for lessening energy consumption in the network, can extend the lifetime of IoT. Clustering is one of the widespread methods to achieve the objective of load balancing. By organizing the IoT network into clusters, where each of them has at least one cluster head node, numerous advantages such as reduction in routing table size, conservation of network bandwidth, lengthening network lifetime, reduction in redundant data packets, and decreasing energy consumption can be reached. For smooth operation of IoT network, load balancing is an essential component of a management solution for IoT. Efforts in this direction can be found in [20].

3.4. Security Management. IoT security is a critical requirement in peer-to-peer distributed systems. Many IoT applications focus on the privacy of data. Moreover, trust management is also required by various applications of IoT. Challenges of provisioning security are worsened by the fact that IoT devices are a constraint in resources. Moreover, it is difficult to implement traditional security methods in these networks; the adaptation of them is also not possible. This poses unique challenges for management framework designer of IoT. The summary of researches on the security management can be found in [21].

4. SDN-Based IoT Management Frameworks

Centralize control of the network with the emerging paradigm of SDN has initiated various efforts of managing the network. SDN paradigm proposes its own designed API to programmatically manage the network. Lately, management for IoT network has received attention, as they are different from the traditional network. Most recently, there are numerous efforts to use the potentials of SDN paradigm to manage IoT network. We classify the existing SDN-based IoT management framework in the literature as a taxonomy in three main groups based on their characteristic properties. The taxonomy of current efforts in this category is shown in Figure 3 that the general comparison of these work-studies with their advantages and disadvantages in the literature are shown in Figures 28 and 29. All of the case studies in this section will be evaluated based on the four parameters, which were discussed in the Section 3. Towards the end of the paper, we will compare the existing framework in terms of provisioning expected functionalities of IoT management framework.

4.1. Network Function Virtualization Based Management Framework. Network Functions Virtualization (NFV) is highly complementary to SDN but is not dependent on it (or vice versa). NFV can be implemented without an SDN being required, although the two concepts and solutions can be combined and potentially greater value accrued. NFV goals can be achieved using non-SDN mechanisms, relying on the techniques currently in use in many data centers. However, current approaches in literature are focused on the SDN-based NFV architectures. The proposed approaches by SDN can enhance performance, simplify compatibility with existing deployments, and facilitate operation and maintenance procedures. NFV can support SDN-based methods by providing the infrastructure that SDN software can run

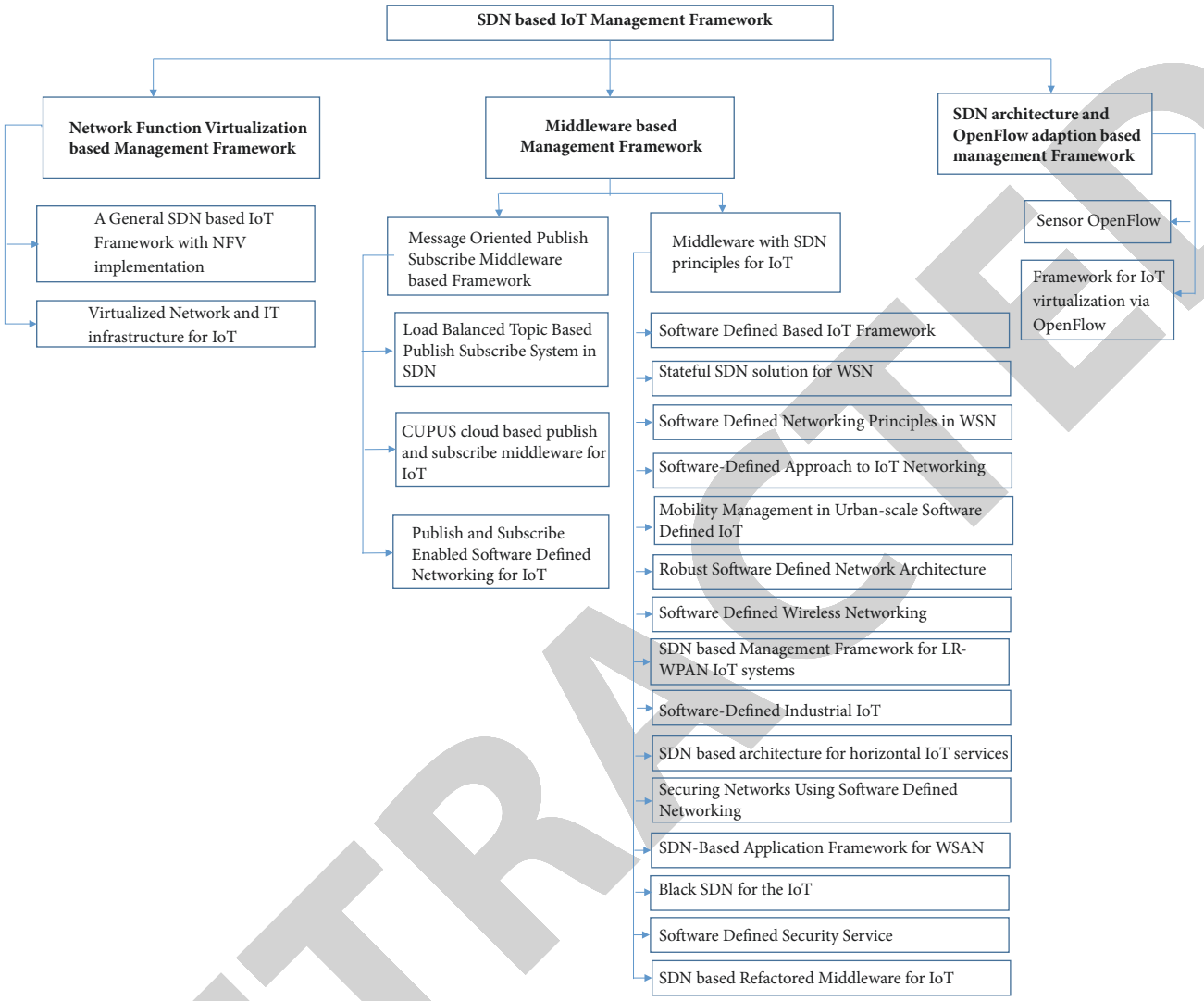


Figure 3: Taxonomy of software-defined networking based IoT management framework.

on. Furthermore, it aligns closely with the SDN objectives to use commodity servers and switches. Frameworks that have virtualized network functions and used network function virtualization for managing IoT are classified under NFV-based management. In this section, we discuss two methods of NFV-based architecture that are based on SDN-based category.

4.1.1. Virtualized Network and IT Infrastructure for IoT [3]

Background. Latest efforts by ETSI have led to standardization called NFV that focuses on virtualizing next-generation network infrastructure. SDN aims at programming network functions that included control layer functions. Hence, NFV enables deploying infrastructure services and SDN provides a tool for dynamic resource control.

Motivation. The network administrators or operators develop catalogs of services related to infrastructure like QoS parameters and bandwidth using principles from graph theory. This ability is possible by combining potentials of NFV and SDN.

Proposed Framework and Critical Analysis. The authors in [3] highlighted some challenges that the IoT pose on the network and IT infrastructure. It is anticipated because the incorporation of the IoT objects in the network infrastructure excessive traffic is generated progressively. In such case, security and privacy are critical for certain IoT application. Moreover, ensuring QoS and congestion control will be a challenge for IoT. It is required that the architecture for IT infrastructure with IoT, usually constrained in resources, to be vigorous, extensible, adjustable, and predictable in order to enable provisioning of infrastructure services.

As shown in Figure 4, the proposed architecture was affected by SDN and virtualization policies and dynamically distributed IOT infrastructure services. Virtualization will decouple hardware from network functions lowering cost. An architectural framework is being standardized enabling

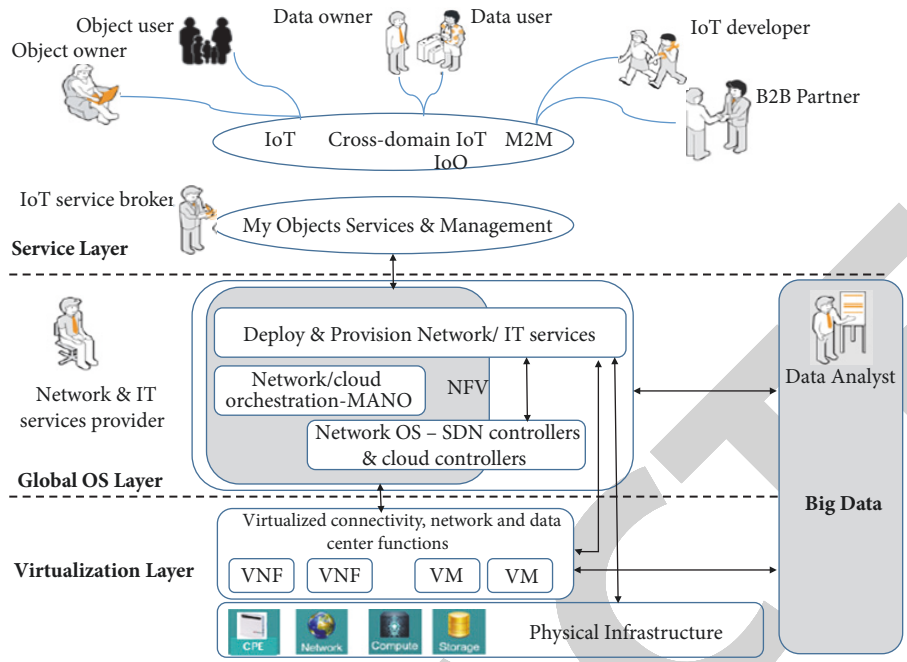


Figure 4: Proposed framework in [3].

a virtualized infrastructure that is called NFV. The NFV and SDN allow network services to be programmed. Virtual Network Functions (VNF) component in NFV moves network functions out of dedicated hardware to software. SDN allows dynamic establishment of connectives between VNFs in NFV. The NFV in the proposed architecture is used for virtualization of resources in the physical infrastructure.

The proposed architecture is organized into layers that are service, global OS, NFVO, SDN controller, and virtualization layer. Services layer implants all service level functions. Global OS layer implants infrastructure's services inventory and description. NFVO manage resources to fulfill infrastructure services. It is responsible for generating new gateways. SDN controller handles end-to-end network control and IT resources. Virtualization layer assembles hardware resources in virtual machines that are made available to the above layers.

Pros and Cons. The proposed architecture is very generic and lacks any detail working of the components. The architectural layers do not have extensive algorithmic work and it is difficult to understand what each layer is responsible for. Description of service level functions and infrastructure services is not given. In addition, no scenarios are discussed which results in new gateway creation as NFVO is responsible for creating new gateways. Furthermore, the authors present no implementation and evaluation of the architecture. As a future direction of the work, detail functionality of virtualized functions requires in-depth study and investigation. In addition, no details are presented on how SDN and NFV are interworking cooperatively to manage IoT.

4.1.2. A General SDN-Based IoT Framework with NFV Implementation [22]

Background. The era of cloud computing and big data have enabled computing and communication resources to be shared and provisioned to the users. As expected, enormous data will be produced by heterogeneous IoT devices. Hence, diversity and big data explosion have brought a huge challenge in designing IoT architecture in this era.

Motivation. Due to challenges raised by emerging paradigms of computing such as cloud computing and big data, it is required that the future generation networks should be agile, intelligent, efficient, secure, and scalable. In order to enable such characteristics within the network innovative adaptive management solutions are required that should be able to deal with a diversity of IoT network. With the emergence of SDN and NFV, it is possible to leverage the potentials offered by this two paradigm to design IoT architecture meeting the requirements.

Proposed Framework and Critical Analysis. The authors in [22] proposed SDN-based IoT framework with NFV implementation. A typical IoT architecture is layered into service, network, and sensing layers. Considering the user and system requirements, the service layer provision information services such as data mining and data analytics are necessary. They may include powerful data centers and heterogeneous data servers. Network layer includes a gateway that routes the transmitted data from gateways to heterogeneous application users. Sensing layers have sensing devices, so, collected data from the sensing devices are transmitted to the gateway using MQTT or HTTP protocol. As the amount of sensed

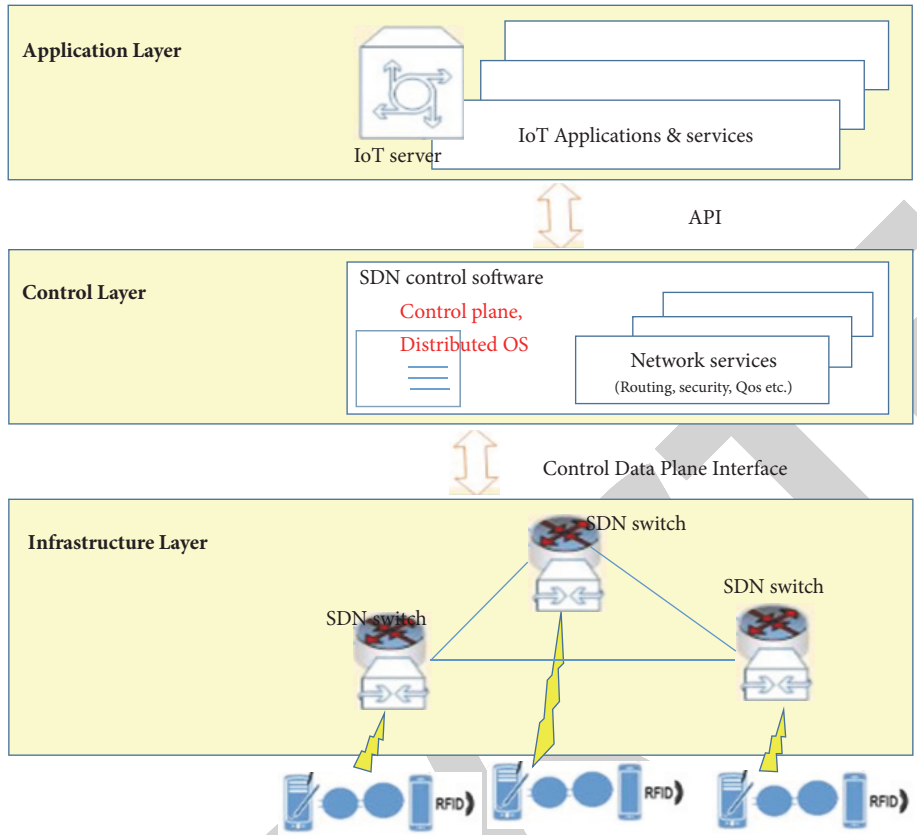


Figure 5: Proposed framework in [22].

and collected data is large, data compression and advance aggregation methods are used for efficient data transmission. NFV virtualizes entire network functions that are connected to create communication services. Network functions are virtualized with one or more virtual machines, which run heterogeneous processes instead of having custom hardware devices for network function.

The authors suggested an OpenFlow-based SDN and NFV implementation, so, it will be possible to implement IoT networking function and a secure tunnel between IoT gateway and prioritization of traffic by a centralized programmable controller. The authors proposed a framework that consists of an application, control, and infrastructure layer, as shown in Figure 5. Application layers consist of IoT servers for different IoT applications and services through API. Control layer consists of distributed OS. The OS provides a centralized control and view of IoT in a physically distributed environment for network data forwarding. Infrastructure layer has IoT gateways combined with SDN switches to provide access to heterogeneous IoT devices that are RFIDs and sensors with a control-data plane interface. Efficient distributed OS for SDN-based IoT supplements the NFV-based SDN framework for IoT. With such an approach, it is possible to have a centralized control and visibility of heterogeneous IoT services.

Heterogeneity of users and infrastructures makes efficient distributed OS design and implementation that is considered a big challenging issue. Lightweight portability layer for SDN has been proposed that ensures portability and performance across heterogeneous switches. SDN-based distributed OS has been released by ONOS (Open Network Operating System) project. Still, there are significant issues of performance, scalability, and availability at the control plane that are ongoing efforts for an efficient OS at this layer.

Pros and Cons. The proposed architecture is generic and the authors do not present detailed working. It also lacks implementation and evaluation. As a future work, detail working of the components is required. In addition, by highlighting the significance of improving the performance of managing IoT with distributed OS, in-depth investigation of its impact on the management framework for IoT is required. Further, the overhead of virtualizing the functions in the architecture requires in-depth study.

4.2. Middleware-Based Software-Defined Management Framework. OpenFlow is a communications protocol that gives access to the forwarding plane of a network switch or router over the network. If it sounds boring and geeky (to a layperson), it is because, in the big picture, it is kind of it. It is specific and does not, on it is own, solve the bigger headaches of network management. Over time, layer 2-3 routers and switches will probably support OpenFlow (or perhaps some other protocol that wins the standardization war), and we will all take for granted that OpenFlow and SDN have enabled network virtualization. Designing middleware

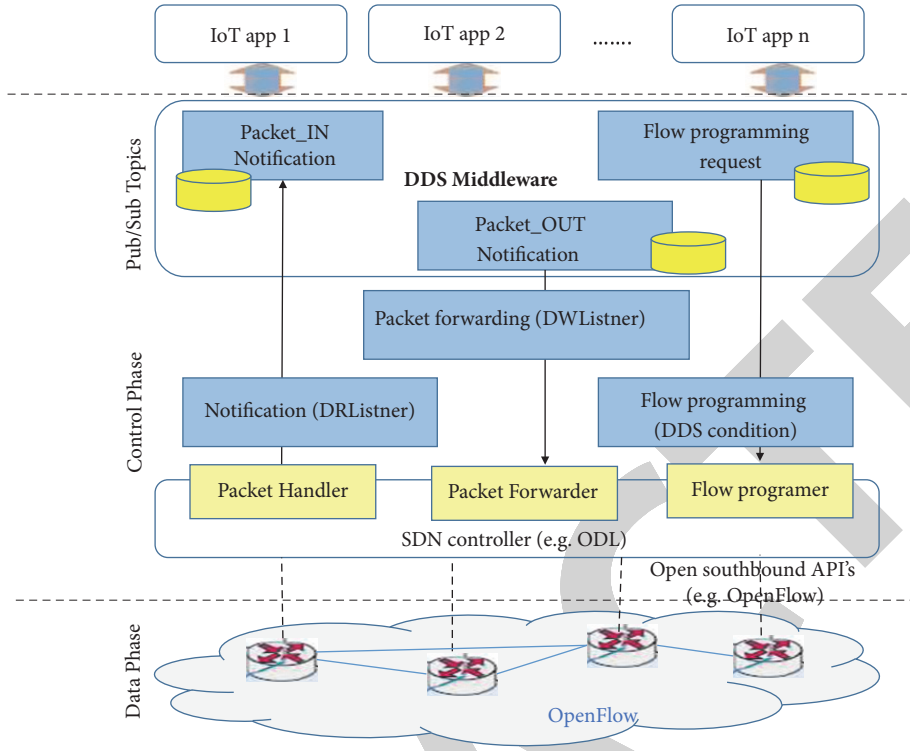


Figure 6: Proposed framework in [23].

for WSN is a heavily researched topic that resulted in several proposals. Efforts of middleware for managing IoT network are classified as a middleware-based management framework. In this section, we will discuss all the efforts that have adopted this approach to managing IoT.

4.2.1. Message-Oriented Publish-Subscribe Middleware Model-Based Framework. There are efforts where the research community has tried to design a message-oriented publish-subscribe middleware with the principles of Software-Defined Networking. We discuss these efforts as follows.

(1) Publish and Subscribe Enabled Software-Defined Networking for IoT [23]

Background. Due to resource-constrained nature of IoT network, the resources in the IoT should be utilized effectively and efficiently for avoiding quick power depletion. A typical IoT network is a mesh of wireless and mobile devices with adaptive states such as switching sleep and wake-up states, maintaining connected or disconnected states. This heterogeneous and dynamic aspect of IoT is a major challenge for the network beneath to support heterogeneity. In addition, existing network protocols have severe limitations and cannot cope with heterogeneous nature of IoT.

Motivation. Varying and dynamic aspects of IoT lead to huge tests for the concealed network that have a requirement to support heterogeneity. Existing provisioning approaches do not deal with the dynamic nature of IoT application and there is no concern of resource utilization. There is a desire need for a programmable management framework for IoT. The emergence of SDN paradigm has oriented the focus of researchers to unveil the power of paradigm to manage IoT network. Although it solves the resource management needs of IoT, fails to address the heterogeneity of IoT network. To address the heterogeneity issue, Web of Things (WoT) is introduced to solve the problem where the devices use common language that is based on open web technologies of HTTP and REST for information sharing. Even though WoT solves the heterogeneity requirement of IoT, it fails to address the need of distributed p2p publish-subscribe semantics as required by IoT applications. Object Management Group's Data Distribution Service (DDS) providing real-time publish-subscribe capabilities is widely used to develop scalable, efficient, and predictable M2M (Machine-to-Machine) applications based on IoT. In order to address the heterogeneity issues of IoT and p2p publish-subscribe requirement of IoT SDN and DDS can be combined in a framework.

Proposed Framework and Critical Analysis. Authors in [23] proposed a Software-Defined Data Distribution Service (DDS) based on a publish-subscribe model, as shown in Figure 6. Proposed architecture consists of three domains such as application, network, and device domain. Device connectivity with the access network is provided by the device domain. Heterogeneous access network enabling connectivity through diverse technologies is provided by network domain. IoT application and cloud infrastructure are provided by application domain. The controller integrates DDS messaging layer that acts as a moderator between IoT and

network. Mediation-layer DDS data reader-listener, packet forwarder service, and finally the flow programming service implement three services. Packet listener to handle events uses DDS data reader so the controller that is forwarded by the SDN data plane gathers them. Forwarding of the received packets (PACKET IN) created by IoT application is handled by packet forwarder service. These received packets are encapsulated in DDS topics sent to packet forwarder service with the help of publisher and data writer listener.

Flow programming service provides rules for flow programming on OpenFlow switches. The authors raise five key networking challenges. Firstly, it is argued that all standardization attempts for IoT are isolated. Existing standardization efforts at different layers of the protocol stack are as follows. At the network layer, standard called 6LowPAN protocol residing between MAC and IPv6 makes IPv6 well matched with resource-limited devices. In addition, ROLL (routing for low power and loss networks) addresses routing issues for low powered devices. At the application layer, CoAP (Constrained Application Protocol) is a specially designed application protocol for resource-limited devices, which cooperate with 6LowPAN protocol to provide application services. M2M effort emerged to encourage the development of end-to-end IoT architecture. There is a need to integrate and interoperate these different standards to enable end-to-end architecture for IoT. Secondly, IoT devices are highly mobile; managing efficient mobility received high interest from IoT community.

SDN can assist by handling mobility as it maintains the complete view of the network but how to provision is a challenge. Thirdly, middleware is required in IoT environment and special interest is on publish-subscribe DDS middleware, which is used by the SDN for IoT architecture to pub/sub-data-centric middleware as it helps in the propagation of events to interested destinations in asynchronous ways. Fourthly, TCP is not suitable for IoT scenarios and there is a requirement to study reliable transport protocol for IoT scenarios. Currently, UDP has been used for CoAP. Lastly, there is no infrastructure to provide security in IoT, as traditional security schemes are complex and heavy. DDS northbound interface to the SDN controller is added to enable IoT to support SDN.

Proposed architecture addresses the networking challenges as follows. Firstly, proposed architecture is based on a standardized paradigm of DDS and SDN. DDS allows the existing standardized protocol to interoperate, as they are able to fit in as an expansion to middleware apart from supplementing complication to network beneath. Secondly, SDN and DDS allow reconfigurable wireless data plane that allows constant dynamic channel configuration, swift client alliance, and mobility management. Thirdly, a single middleware is proposed in the architecture that supports multiple communication pattern a profitable way to establish and preserve large distributed IoT systems. Fourthly, DDS layer implements DDS discovery that assists heterogeneous smart devices to discover each other. IoT gateway maintains a list of local objects in string formatted version building a view of all the neighboring devices. Lastly, SDN and DDS handle security issues. DDS security model provides simple and interoperable security policies that are brought forth by DDS standard.

Pros and Cons. There is no implementation of the proposed architecture. DDS middleware along with SDN paradigm is an added expensive overhead for provisioning IoT. Services of DWListener, DRListener, and DDSCondition add up computation cost as they are running and listening to the packet events all the time. In addition, the cost is further alleviated by the presence of database at the DDS middleware storing, processing requests and packets coming and going from IoT application and control plane.

(2) Load Balanced Topic-Based Publish-Subscribe System in SDN [24]

Background. A massive stream of data is expected to be collected across IoT network that raises challenges in delivering sophisticated multisource sensor data in real-time efficiently. Traditional request-reply is insufficient to address this challenge and hence can be addressed by publish-subscribe paradigm facilitating data dissemination. In traditional network architecture because of the absence of global traffic information publish-subscribe systems usually experience inadequate utilization of physical network infrastructure, which may lead to traffic imbalance on heterogeneous links.

Motivation. A centralized management framework enabling programmability within the network can fulfill the need for universal traffic information. A type of publish-subscribe paradigm called topic-based publish-subscribe system can be used for IoT where events are published with specific identifiers called topics. It is the duty of publish-subscribe paradigm to guarantee dissemination of every new event to the subscribers expressing their interest for a topic similar to event. SDN enables a global view of the network that can fulfill the requirement of universal traffic information. Data Dissemination in IoT can be done with topic-based publish-subscribe paradigm combined with SDN enabling centralized control and management.

Proposed Framework and Critical Analysis. The authors [24] proposed an SDN-based publish-subscribe system known as SDNPS. They propose a minimal cost topic-connected overlay (MCTCO) algorithm that works by constructing an optimized routing schema with the assistance of traffic engineering. Global view of the topology is obtained by accumulating link state. Traffic distribution is predicted for an interval of time in terms of a likeness of a pile of data from IoT. Usually, in a topic-based publish-subscribe system; specific identifiers distinguish the events that are published and are called topics. The publish/subscribe paradigm then guarantees disseminating every new event to those subscribers who have expressed their interest in the topic similar to the event. Upon completing the prediction of traffic distribution, the lowest level overlay networks in a topic are calculated.

The overlay network is constructed using intercluster routing technique. Such technique usually has various varying features of event dissemination mechanism. The system

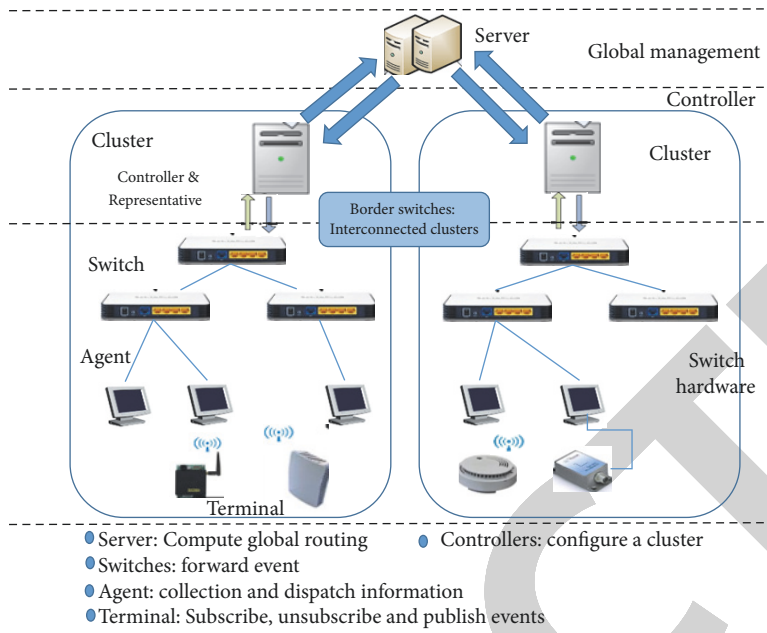


Figure 7: Proposed framework in [24].

consists of three layers. They are switch hardware, cluster controller, and global management layer. Switch hardware layer performs filtering and event forwarding. Controller layer executes OpenFlow protocol. In addition, the controller has control applications running in them. Global management layer has server and standby server having universal information of physical topology and subscription topology. A topic-connected overlay network is used for traffic engineering.

Proposed SDNPS, shown in Figure 7, maintain two types of topologies, subscription and physical topology. In case of subscription topology, every cluster has universal subscription information for every topic constituting subscription topology. For the case of physical topology, the controller detects physical link state of its own cluster. They form intracluster topology and maintain attainability information to neighbor clusters selected periodically by transmitting heartbeat information. Topics in the proposed SDNPS are maintained as a topic tree. To enhance security and privacy, some clusters might prevent some event with specific topics that are handled by strategy management. In order, to achieve optimal performance traffic engineering is used by the proposed system. Servers and controllers have all the functions of traffic engineering deployed in them. The server seeks optimal routing path among clusters and flows are divided among numerous paths to equipoise traffic. Routing in proposed architecture is segregated into two levels. These levels are intercluster routing and intracluster routing. Servers handle intercluster routing and the controllers handle intracluster routing.

Pros and Cons. Proposed architecture has the overhead of computing topic tree and maintaining clusters in the network. There is no evaluation of assessing the performance of proposed technique with the growing tree size depth wise. As the depth of the topic tree will increase, the performance of the proposed algorithm is expected to degrade. This is because the time to compute minimal cost path in the tree is increased due to the large depth of the tree. In addition, maintaining clusters in the network is an extra overhead of managing cluster states that requires storage. At the same time, the proposed architecture addresses scalability issues and delays in the IoT environment. Evaluation is performed on simple SDN testbed that consists of commodity PC hardware and virtualization technologies. Three-hop topology is constructed with link rates of 10Mb/s and packet length of 200 bytes. Three IBM servers with eight core CPU and 16 GB memory virtualize network elements. Virtual machines used for virtualization validate the proposed schema. Results suggest that the proposed algorithm generates a load balanced routing schema that addresses traffic imbalance situation usually experienced in the conventional publish-subscribe system. In future, the authors look forward to addressing the real-time requirement for some events in IoT.

CUPUS Cloud-Based Publish and Subscribe Middleware for IoT [25]

Background. In order to collect and gather data over a large geographical area, an emerging paradigm of mobile crowd sensing (MCS) has lately attracted the attention of the research community. MCS is a novel mobile Internet of Things application that sensors and mobile devices jointly collect share data of interest. MCS applications have the characteristics of moving data sources that are generating sensor streams lying in the domain of big data. Challenges in MCS application domain are unique; some of them are as follows: dynamic environment comprising sensors, various mobile devices, and cloud makes it necessary to achieve energy efficient and context-aware sensing process. This

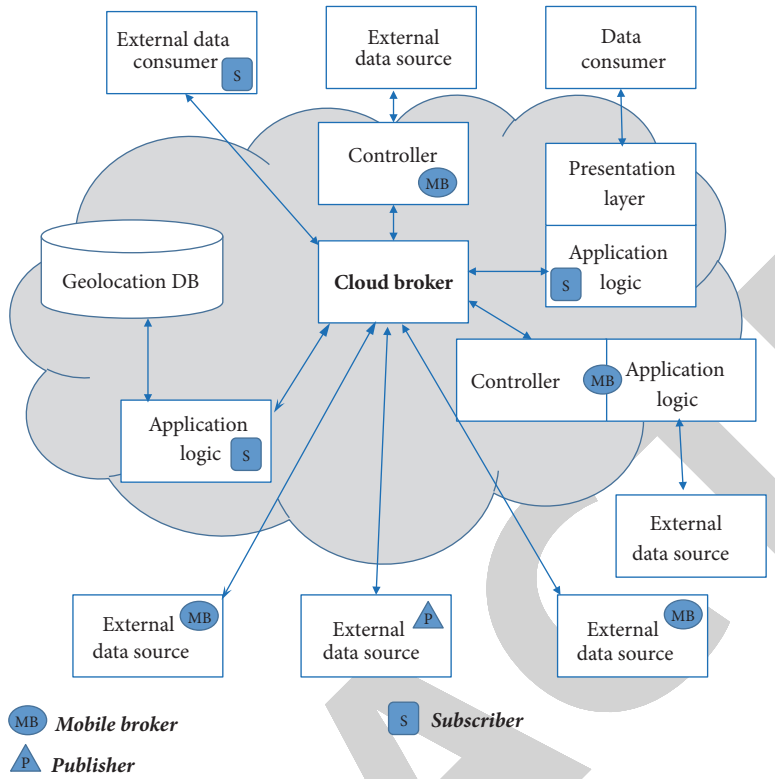


Figure 8: Proposed framework in [25].

means both sensing process and data transmission from mobile devices to the cloud need to be controlled.

Motivation. To deal with the challenges of MCS as discussed above publish-subscribe middleware is a potential candidate. They are dealt with by filtering sensor data on mobile devices afterward data is forwarded to the cloud. Efficient continuous processing of large data volumes within the cloud takes place. In the end real-time, delivery of notifications to mobile devices takes place. Motivated with the middleware the authors in [25] proposed content-based publish-subscribe model named CUPUS within the cloud and mobile devices.

Proposed Framework and Critical Analysis. The proposed framework, CUPUS (Cloud-based publish-subscribe middleware) [25], consists of two main software components that are a mobile broker and cloud broker as shown in Figure 8. The mobile broker has the responsibility of data filtering/aggregation on mobile devices. Main tasks of the mobile broker are to control connected sensors locally and corresponding data acquisition process to preprocess acquired data and transmit to the cloud and to receive publications from the cloud and display them to the user of interest. Cloud broker is used for continuous processing of big stream within IoT cloud. It acts as a central processing unit and performs the following task: manage publications and subscriptions received from data sources and destinations, perform matching of active subscriptions to publications, and deliver matching publications to subscribers and subscription delivery to mobile brokers. The authors compare the proposed framework CUPUS with two standardized message protocols for IoT. These are (MQTT) message queue telemetry transport and constrained application protocol (CoAP). CUPUS is evaluated in terms of end-to-end propagation delay from the data source to the data consumer.

Cloud broker and mobile broker are evaluated. In order to measure the performance of cloud broker SenseZgAir data set is used. The experimental setup consists of two different virtual machine instances that are running on Dell PowerEdge R720 rack server with the Citrix Xenserver virtualization software. Scalability and elasticity of cloud broker are by varying the number of subscriptions. It is varied from 1 to 20,000 for scalability and 5000 to 50,000 publications for elasticity. It is shown to be scalable when increasing number of subscriptions, while the broker also adapts well for increasing and decreasing workload. End-to-end propagation delay is shown to be in the range of 1-2s. Processing time on mobile devices is within the range of 10 ms for typical loads.

Pros and Cons. CUPUS is designed to cater the resource-constrained requirements with the objective of reducing overall energy consumption from data acquisition to transmission and delivery. The proposed framework also controls the data density in MCS with respect to both time and space. This is done by filtering and aggregating data closer to production place. At the same time, data transmission from mobile devices is compressed into the cloud. The authors do not define a number of acceptable cloud brokers and mobile brokers in CUPUS in [25]. It seems that there will be one

cloud broker in CUPUS that may result in increased load on a single cloud broker. The authors in their evaluation do not analyze this. Optimal load distribution over mobile and load brokers is not analyzed and evaluated by the authors.

4.2.2. Middleware with SDN Principles for IoT. In this section, we discuss all the efforts of designing middleware for IoT with SDN principles for resource management, meeting requirements of IoT services, and efficient data collection within IoT/WSN.

Background. IoT network requires a framework to manage resources in the network. The authors [26] stress that no former investigative study on developing an extensive software-defined solution for IoT exists. Due to the nonexistence of one fit for all management solution for IoT/WSN, there exist various vertical application-specific solutions, because there are severely fragmented context and market [27].

IoT services are remarkably different from the traditional Internet services. IoT services most of the time rely on energy and CPU constrained sensor technologies irrespective of if the service is for home automation, smart building, and e-health. Moreover, monitoring applications implemented with the help of IoT should forward the data in a secure manner without enabling manipulation of data [28]. In addition to data delivery, it is expected that IoT applications such as smart city will be heterogeneous because of diverse access networks as an effect of geographically wide scale IoT multinetworks being developed lately. In a typical IoT network, various wireless communication solutions will exist in coexistence [29]. The authors in [30] have stressed that, due to lack of single wireless standard with IoT devices, selecting the appropriate wireless connectivity is a daunting task. This is due to the fact that the conventional network is inadequate to encounter the challenge. In addition, with expected 50 billion devices, it is predicted that wireless communication interference is inevitable in IoT. This motivated the research community to investigate novel ways of increasing robustness due to the increase in the presence of wireless interference [31].

Statistical machine learning can be used to identify interference patterns over time. Considering all the above issues of resource management in IoT and the existence of diversified access technologies with severe interference, researchers are investigating programmable middleware designed by standardized technology such as SDN for controlling IoT/WSN devices. An effort such as in [32] can be found, where the authors have highlighted the problem of applying SDN in WSN that was considered to be impractical due to resource limitations of WSN devices.

Motivation. Conventional IP standards are helpless to cope with a huge number of things in IoT interconnected with the cyber world. In addition, in order to customize IoT/WSN according to specific application requirements, it is essential that the IoT should have reconfiguration capabilities. Motivated with designing a programmable IoT/WSN with a standard middleware, there are various efforts for applying SDN principles in a middleware to meet the issues discussed above, which can be found in [26, 27, 33]. The emerging paradigm of SDN centralizes the control plane enabling the whole network to be configured through the central controller. Apart from the reconfiguration of IoT/WSN SDN, facilitate data collection over IoT/WSN and IoT application services. One such effort can be found in [28].

At the same time, SDN paradigm has the potential to handle diverse traffic demands according to ever-changing traffic volumes. Hence, the authors in [29] are confident that SDN is an excellent approach to solve resource management and access control issues in IoT multinetworks. Existing implementations of SDN technologies do not address diverse and vigorous necessities of IoT applications, particularly for mobile environments. Managing multinetworks with persistent IoT access requires deep research as it has not been looked upon up until now.

Lastly, middleware with SDN principles for IoT needs to be resource efficient. An effort towards this direction can be found in [32]. Where the authors have stressed on developing a novel and customized OpenFlow approach for WSN/IoT. Other similar efforts, which applied SDN principles, can be found in [21, 34–39]. We discuss all of the above efforts of middleware solutions for IoT that have adopted SDN principles as follows.

(1) Software-Defined Based IoT Framework [26]

Proposed Framework and Critical Analysis. Based on the design principles of SDN, SDStore (Software-Defined Storage), and SDSec (Software-Defined Security) a software-defined IoT system architecture is put forward by the authors that assist in the management of IoT. Conventional data storage system stockpiles a large amount of data. Burden on underlying devices reduces performance due to occurrence of forwarding, processing, and management processes for the data at the same instance. SDStore simplify and facilitate such complexity by keeping apart control layer of the data from the storage layer.

Many commercial corporations have applied SDStore in their storage center [40]. Traditional security mechanism cannot be applied on emerging paradigm of SDN and SDStore. Emerging SDSec, an example of network function virtualization, provides a novel way of provisioning security by segregating security provisioning control plane from forwarding and processing plane. The scalable distributed security solution is possible with such separation and virtualizing security function that remains manageable as a single logical system. In addition, SDSec secures virtualized environment.

The architecture has three components physical layer, control (middleware) layer, and application layer as shown in Figure 9. Hardware devices are found in the physical layer. Physical devices collect data and send to the successive layer. The layer is laid out into clusters: sensor network, database pool, switch routers, and security appliance cluster. Sensor network cluster contains set of sensors organized in clusters. Database pool cluster is responsible to store data with different types. The middleware layer consists of software-defined controllers that are IoT, SDN, SDStore,

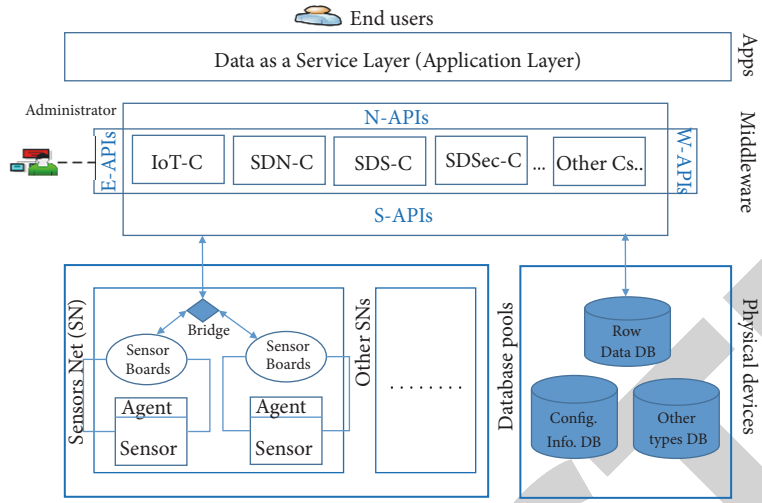


Figure 9: Proposed framework in [26].

and SDSec controller. The administrator reconfigures devices with East APIs. The sensor network clusterWest APIs enables communication among the controllers. Application layer has a combination of the user application that accesses the stored data using Northbound API. Group of collectors collects data and SDSec controller is used to authenticating the object used for collecting the data. After collection, messages are processed and passed to IoT controller that computes forwarding information being forwarded to SDN controller. Ultimate routing and forwarding information is then given to the devices in sensor network cluster. The authors discussed their work superficially by proposing a very generic architecture.

Pros and Cons. Integrating various SDN supported controllers for various functions in a single middleware is resource intensive and requires careful analysis of their interoperation. Any implementation and evaluation of the proposed architecture were not presented; hence, there is no way of knowing whether such a middleware is feasible to implement. As a future work, working details of communication among various components in the architecture by the APIs need to be investigated and presented in detail.

(2) Stateful SDN Solution for WSNs [27]

Proposed Framework and Critical Analysis. The authors in [27] proposed a stateful solution for wireless sensor networks (WSNs) called Software-Defined Networking for Wireless Sensor Networks (SDN-WISE) shown in Figure 10. In the beginning, the authors discussed the need for enlarging SDN paradigm to WSN. It is indicated that WSN usually is limited in resources in terms of energy, processing, and memory availability. Three data structures used to capture the behavior of SDN-WISE states array, accepted IDs array, and WISE flow table. The controllers from where the information is originated fill the structure. Accepted ID array enables sensor nodes accepting data payload given in the array. Controllers define network management policies that are implemented in WSN. In SDN-WISE networks, sink acts as an entry point between sensor nodes that have data plane and control plane. Sensor nodes are equipped with data plane protocol stack. Forwarding layer executes on top of IEEE 802.15.4 protocol stack that handles arriving packet using WISE flow table.

In-network processing (INPP) layer performs link data aggregation. Topology discovery layer admits all layers of protocol stack with the assistance of APIs. Control plane network management is performed by topology management (TM) by abstracting network resources and enables different networks with heterogeneous policies set by different controllers. Adaptation layer performs formatting for messages that are received from the sink topology discovery protocol responsible for generating local topology information that is delivered to WISE-Visor. Next hop to controller information is maintained by the protocol with the help of topology discovery packet over a broadcast channel. Packets contain the identity of the sink, power of the battery, and interspace from the sink. The packet header structure consists of a fixed header that consists of 10 bytes and has the fields of packet length, scope, source and destination addresses, type of packet, TTL, and next hop ID. The WISE flow table structure is similar to OpenFlow having three sections, which are distinguished as matching rules, action, and statistics. Three conditions are specified by the matching rules upon finding a match. At this point, an action is carried out and information that is communicated in statistics sections is refurbished.

The proposed architecture is tested using EMB-Z2530PA (wireless module developed by Embit for low rate wireless personal area networks applications) based sensor nodes. Six nodes have been deployed with five sensor nodes and one sink. The sink node was running adaptation layer with control plane functionality. Controller run Dijkstra algorithms and is implemented in Java. 5000 data packets have been sent every 15 seconds. The performance of the proposed SDN-WISE is shown in terms of the round-trip time, efficiency measured by an overall number of bytes sent and the number of payload bytes received at the intended destination. It is shown that the

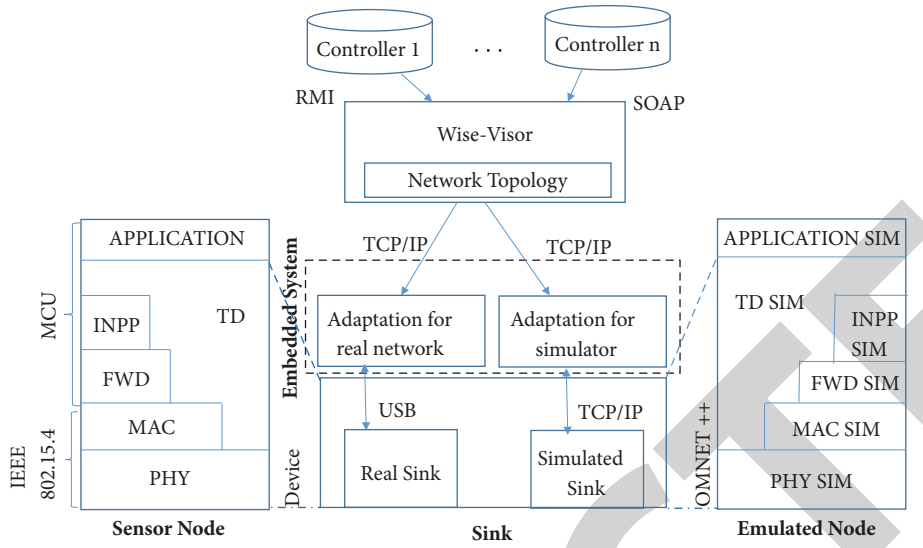


Figure 10: Proposed framework in [27].

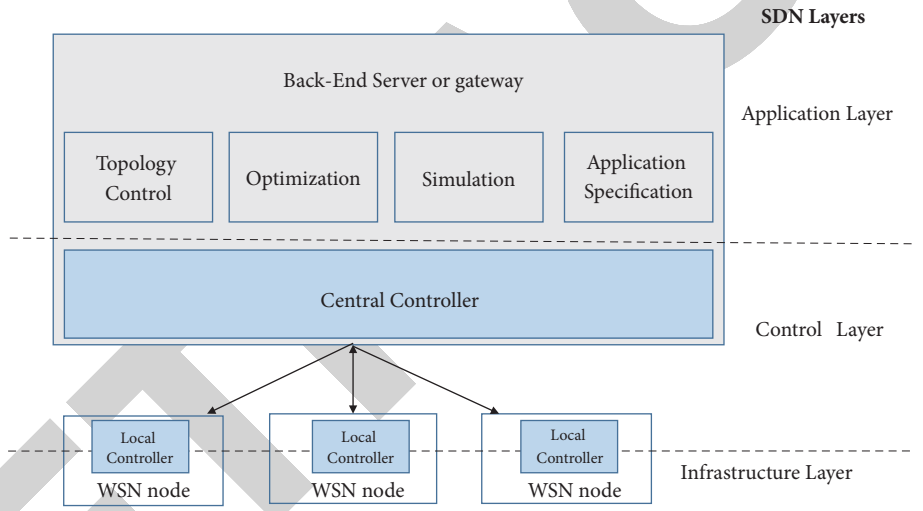


Figure 11: Proposed framework in [33].

proposed architecture is efficient and realizes programmable WSN.

Pros and Cons. It is not clear how the SDN-WISE protocol stack will fulfill management challenges of IoT. Proposed architectures have an overhead of topology discovery protocol to gather local topological information to assist in managing IoT. In addition, there is an extra overhead of maintaining states of WSN. Investigation of addressing management challenges of IoT is required and is a future direction of this work.

(3) Software-Defined Networking Principles in WSN

Proposed Framework and Critical Analysis. In [33], the authors propose an architecture, as shown in Figure 11, to manage WSN with SDN layers. A WSN node is equipped with a local controller that receives and executes the command from the central controller. One or more SDN applications will lie on top of the controller. Applications are related to networking of WSN topology control and routing. Local controllers lie in the sensor nodes. The controller can modify not only MAC but also routing. Researchers have focused on adaptive and reconfigurable adaptive MAC platform. Centralized architectures for WSN have been put forward where the main functionality is moved from simple WSN nodes to the sink. SDN for WSN and OpenFlow in WSN have been looked into and there are various proposals on the efforts. Radio hardware is replaced with software reprogrammable module where it is called software-defined radio.

Partial local controller dwells in the infrastructure layer. The local controller is responsible for reconfiguring and monitoring software. This is done in two ways; firstly, either

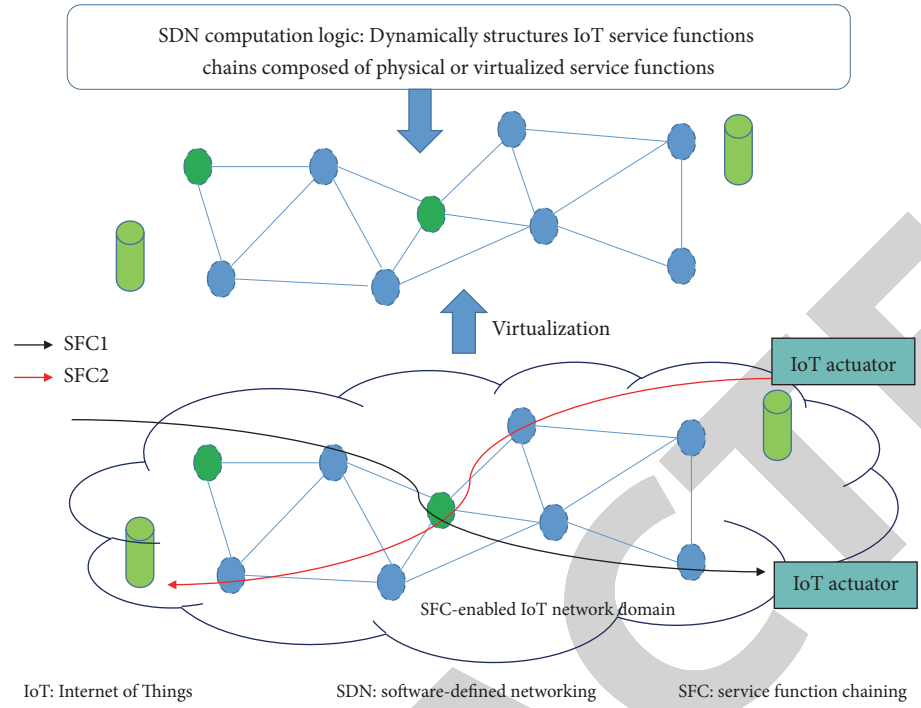


Figure 12: Proposed framework in [28].

parameter is changed in different function such as the central frequency of the radio, MAC layer retransmission limit, altering entries in the forwarding table. Secondly, the novel code is installed in varying functions that change behavior. Virtual machines (VM), native code, and dynamically linked libraries do this. SDN centralizes networking with MAC; forwarding and many routing decisions are carried out at individual nodes. Long-term decisions are taken by a centralize controller such as what protocol and parameters to use.

Central controller discovers actual topology and quality of the links. Packet trace information and Link Quality Estimation (LQE) are used to estimate the quality of the links. Topology and link information is used for various reasons. Four ways are used to apply this information with LQE and packet trace information is used for routing or tuning routing parameters. Optimal placement of a piece of code has value on the network lifetime. With knowledge about the network optimal code, placement strategy can be devised. In addition, knowledge of network can be used to predict WSN network behavior, network lifetime, and performance. The authors discussed their work superficially by proposing a very generic architecture.

Pros and Cons. Synchronization and coordination of the central controller and local controller are not addressed. It is not clear what management functions are distributed among central and local controller. In addition, there are vague details of what kind of management tasks for IoT are supported by the proposed architecture. A prototype implementation and evaluation of the proposed architecture are not presented and it is left as future work by the authors.

(4) Software-Defined Approach to IoT Networking

Proposed Framework and Critical Analysis. The idea, as shown in Figure 12, proposed by the authors in [28] can be used vigorously to expose and negotiate parameters of IoT service. Hence, anyone can have global systematic software-defined IoT networking approach. Furthermore, the authors introduce two IoT services e-health services, energy management, and distribution. E-health requires network infrastructure that is highly reliable in preserving data integrity. In contrast to IoT services, connected devices are responsible for sending data, while some e-health services require bidirectional traffic transmission that is for tweaking threshold settings. For some e-health scenarios, biometric data require quick reaction for health emergencies and often involve dynamic route computation for sending data.

The other use case that the authors considered is the dynamic management of energy distribution where large-scale IoT will be used. Collected data from the power meters are usually forwarded for billing and have a major contribution to managing energy distribution during peak seasons. IoT services require operators to be flexible and agile during service delivery. With the IoT adapted SDN it will be possible to enforce traffic forwarding policy in an IoT infrastructure dynamically that performed according to the virtualized functions. IoT service delivery with SDN is realized by negotiation of IoT service parameters that help in performing resource and IoT service allocation across IoT network and enforcement of policy in IoT network. In order, to observe the behavior of IoT infrastructure data analytics is performed on the data gathered over the IoT network as events.

SDN platform is used to manage IoT services. Differentiated traffic forwarding policies in an IoT infrastructure are used to prioritize traffic in IoT network. SDN approach usually has a bootstrapping method for dynamic discovery of IoT network topology. In the proposed architecture, lower layers with MAC are in commodity hardware. Upper layers from the IPv6 network layer to application layer CoAP (constrained application protocol) and HTTP are virtualized and controlled by SDN.

Pros and Cons. Details of the functional components of the proposed architecture are not given. It is claimed that most of the IoT gateway and IoT nodes function will be relocated and virtualized as VNF, which will be coordinated, by SDN/NFV controller or orchestrator. No details of where these VNF will be hosted in the IoT infrastructure are given, as this can be a major performance bottleneck in the network. Due to resource constraint, nature of the devices virtualization over IoT devices is not feasible. Hence, there is a need of few servers virtualizing a part of IoT infrastructure; the performance of the server will be dependent on the number of IoT nodes virtualized by a single server. Algorithms realizing the architecture are missing and the authors do not give implementation prototype of the idea and evaluation. This is left as future work by the authors that involves multimetric route computation, cross-platform IoT networking, and IoT specific service function chaining.

(5) Mobility Management in Urban-Scale Software-Defined IoT [29]

Proposed Framework and Critical Analysis. The authors in [29] have proposed Ubiflow, a software-defined IoT architecture, shown in Figure 13. Multiple controllers have been adopted to split up urban-scale SDN into geological partitions attaining distributed control of IoT flows called to share. Ubiflow controller distinguishes flow schedule that is set up according to device requirements. The controller also gives a view of the network status. Such view can be used for choosing better access points in multinetworks satisfying flow invitation. The main contribution of Ubiflow is unique overlay structure achieving mobility management and fault tolerance in software-defined IoT. It has an algorithm for allocating controller in meeting perfect unengaged access point to IoT gadgets. Core components of system architecture of Ubiflow are switches, access points, data server, controllers, and Internet devices. IoT gadgets within a single share connect different type of access points associated with local switches to desire varying types of data flow from the corresponding data server.

Data collection component gathers network/device information from IoT multinetwork neighborhood at the same time caching in the database. Gathered data is used by layered components in the controller. In addition, at the same time component responsible for task resource matching applies task request onto existing resources in multinetwork. When possible choice resources are chosen solution specification component sums more network attributes and limitations to filter resources. Flow scheduler component takes the demands and schedules flows to fulfill them.

To handle mobility, homogeneous schemes to handle controllers are needed for mobility management of IoT gadgets. There are two types of IDs that are utilized for mobility management that are mobile ID and controller ID. Efficient handover is maintained through the coordination between controllers. In order, to handle failures in distributed SDN different component failures are tackled in Ubiflow. With the presented flow requirements of an IoT device, Ubiflow controller finds the access point that satisfies flow request of IoT device and confirms optimized network performance.

In order to achieve network performance, the network is partitioned using network calculus by obtaining network condition within the share of a partition for flow scheduling. The handover procedure among shares is assisted by the partitioned view of the whole IoT network that helps in handing off to varying access network. One of the limitations of SDN is load imbalance where calibrating between a switch and controller is constantly configured that makes control plane adjusting to temporal and spatial traffic load displacement difficult.

Pros and Cons. Proposed architecture addresses most of the challenges of IoT management such as fault tolerance and load balancing. At the same time, the architecture has multiple controllers that inherently lead to consistency and synchronization problem (not discussed by the authors). Implementation is carried out in Omnet++; the parameters observed are end-to-end throughput, delay, and jitter. For real testbed experiments, ORBIT is used as wireless testbed to evaluate proposed architecture where throughput and delay are observed. ORBIT has floodlight based OpenFlow controller for WiFi and WiMAX interfaces and is composed of 400 radio nodes. There are number of experimental sandboxes, which include WiFi and WiMAX, accessed by management framework. Ubiflow scheduling is compared with conventional famous scheduling algorithms of Devoflow and Hedera. In addition, Ubiflow is compared with GENI an OpenFlow-based handover implementation where Ubiflow has shown effective performance in terms of throughput and packet delays and at the same time experiences less handover delay.

(6) Robust Software-Defined Network Architecture [31]

Proposed Framework and Critical Analysis. The proposed idea [31] is based on network architecture based on SDN principles that are energy efficient, reliable, and robust. A statistical model is implemented at the controller. Complex computation can take place without considering any resource limitation. Sensor nodes operate at data plane. SDN controller is implemented at controller plane. The controller gathers required information for the statistical model. Multivariate linear regression algorithm is used as the statistical model at SDN controller. Interference related training data is gathered for several weeks and foretell the existence of intervals of lofty interference. Based on the foretells, SDN controller is in charge of doing network modification. The authors claim

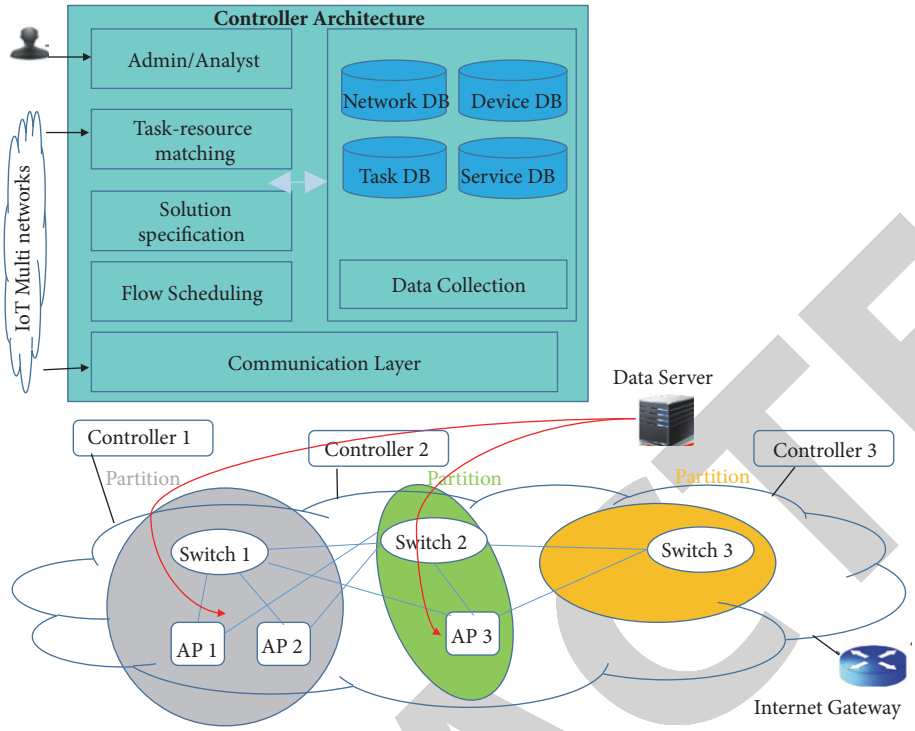


Figure 13: Proposed framework in [29].

that the proposed idea will be able to stop network defects in a well-organized way.

Pros and Cons. The work lacks details of working, implementation, and evaluation, as it is a position paper proposing a framework for managing IoT. It is discussed in a very abstract manner.

(7) Software-Defined Wireless Networking [30]

Proposed Framework and Critical Analysis. The authors [30] presented architecture of integrating SDN and IoT, as shown in Figure 14. New opportunities in SDN are possible with hybrid network architecture. The network control is simplified using a common protocol. With the integration of SDN and IoT, more issues are faced that are on security and scalability. In the proposed architecture, the IoT controller will coordinate IoT devices and in turn, and SDN controller will interact with IoT controller to implement routing policies over the IoT network. The authors discussed their work superficially.

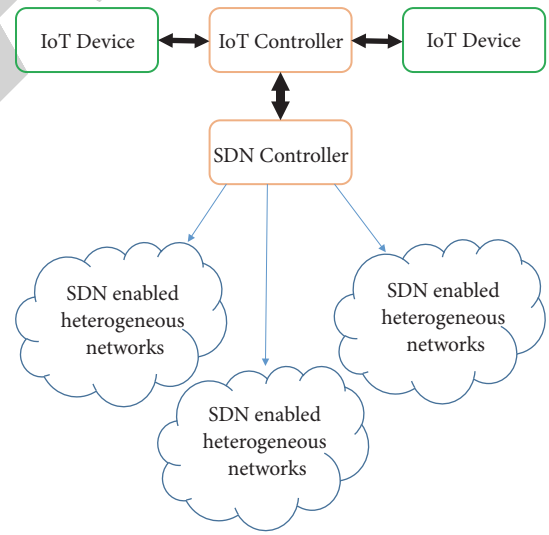


Figure 14: Proposed framework in [30].

Pros and Cons. The proposed architecture is presented in a very abstract manner. Major management challenges of IoT are not addressed. It is not clear how the components in the architecture will communicate. No implementation and evaluation are presented.

(8) SDN-Based Management Framework for LR-WPAN IoT Systems [32]

Proposed Framework and Critical Analysis. The authors in [32] proposed a SensorSDN framework, as shown in Figure 15, which is a novel SDN model fulfilling the specific requirements of diverse WSNs. A control plane is proposed that handles automatic topology discovery and the mobility of nodes and at the same time managing network policies. The controller in the framework provides these services. Sinks are discovered by topology discover service that performs packet aggregation and traffic engineering. The management service provisions network policy control and security over the network. Lastly, virtualization service is also part of the controller enabling sharing of the same WSN infrastructure.

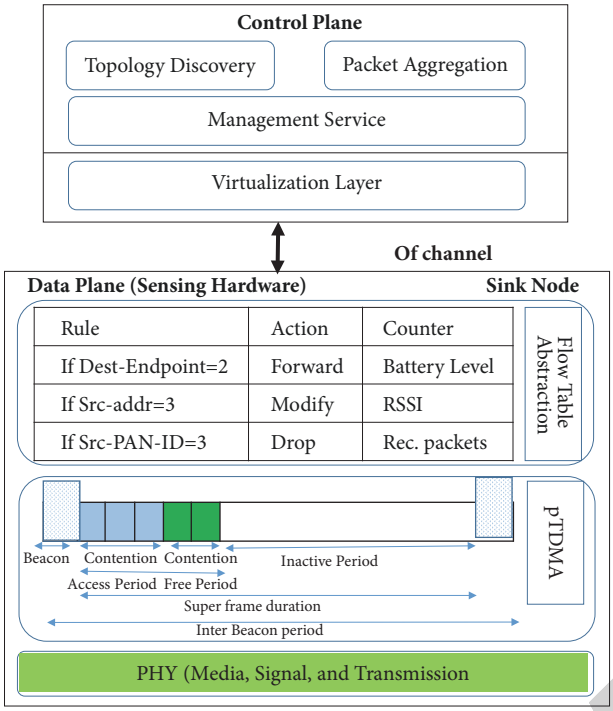


Figure 15: Proposed framework in [32].

The current implementation of SDN does not support WSN packet headers. Therefore, the authors define new header fields for packet matching. For data forwarding, the authors proposed a TDMA layer that interacts with network layer provisioning dynamic and flexible data forwarding. It is no doubt that great authors present work but no evaluations are performed. It is not possible to judge what will the overhead of provisioning services by the controller and proposed TDMA protocol. Also, it is not clear what will be a sequence of messages that will be exchanged by the control plane and the data plane for provisioning topology discover and virtualization service over the network. The authors discussed their work superficially.

Pros and Cons. The framework provisions novel services to WSN such as virtualization service and topology discovery. It is no doubt that these services will be useful for WSN. At the same time as there are no evaluations of the proposed framework presented, it is difficult to assess the cost of modules in the framework. In particular, TDMA protocol enabling MAC layer programmability needs to be evaluated. It is possible that WSN applications and WSN traffic load proposed programmable MAC layer might be costly in terms of resource consumption. These questions need to be addressed and investigated.

(9) Software-Defined Industrial IoT [34]

Proposed Framework and Critical Analysis. It is daunting to implement IoT in present-day industrial systems due to problems and challenges in terms of information based interaction. Due to lack of software-defined IoT, it becomes challenging to apprehend, analyze, and make use of all kinds of information in a coherent manner from heterogeneous sensor devices.

In order to promote malleable network resource management, the authors in [34] proposed software-defined IoT architecture. The IoT architecture, which is shown in Figure 16, has four components: machines, equipment, networks, and the cloud and terminal; proposed software-defined IoT architecture is composed of three layers: physical, infrastructure, and control and application layer. Further, software-defined IoT architecture provides three kinds of services: data collection, data transmission, and data processing. In industrial IoT, performance and communication device functioning are extremely arduous. High real-time performance is required by few applications. Few of the application tasks are performed periodically, which usually trigger events. Due to such characteristics, the difficulty is increased for practical applications of IoT.

The rules are accessed by software virtualization technology. IoT generated data is of great commercial value and securely manage devices that are spread in IoT is a great challenge to defend against the threat of the generated data. Technology itself is not the main barrier to the widespread application of IoT and it is the interoperability interfaces of systems belonging to different vendors that inhibit adoption. In order to achieve good interoperability, the key is to unify standardization of interface intercommunication among varying components possessed by heterogeneous vendors.

In comparison with conventional networks, the role of designing forwarding plane in IoT has increased. In particular, it is a demanding task in SDIoT. Data collection, data transmission, and data processing services are proposed in the architecture. In data collection service, applications are developed using API. Consider data transmission service of wireless or wired network forward perception data to an industrial cloud. Combining SDN with IoT enables modification of network devices. Furthermore, existing problems and possible solutions for software-defined IoT are discussed that are data safety and reliability, technology standardization, and practical implementation. The authors discussed their work superficially.

Pros and Cons. Management challenges for IoT are not addressed by the proposed architecture. It only considers collection and transmission of data to the SDN applications. Detailed working of the API (northbound and southbound) used for communication with heterogeneous data sources is not given. As future work, detail algorithms of the components in the proposed architecture need to be investigated. Proposed architecture was analyzed by designing a prototype platform that consists of physical equipment that are cloud data center, industrial robot, an AVG, IWN, RFID reader, conveyer, and work piece. Comparison of the traditional scheme and software-defined IoT is performed. For this energy efficiency and consumption are evaluated for the proposed architecture and conventional schemes. It was found that the proposed architecture consumes less energy. Intelligent services can be provided by the proposed

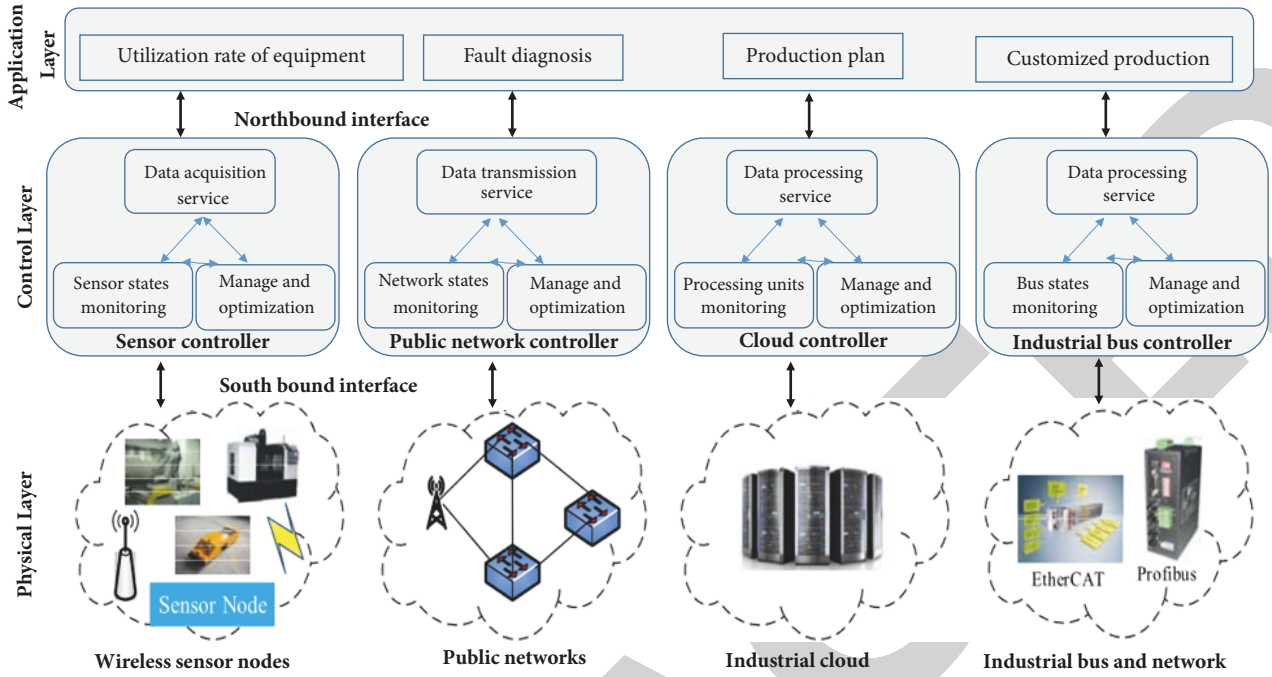


Figure 16: Proposed framework in [34].

software-defined IoT. Traditional schemes exhibit a reduction in autonomous decision.

(10) SDN-Based Architecture for Horizontal IoT Services

Proposed Framework and Critical Analysis. In order to address the concerns raised, in [35] the authors put forward an IoT architecture model, shown in Figure 17. The main ambition for progressing IoT architecture is to reutilize diverse resources and enable swift deployment for IoT services and applications. The design principles are specified as layered architecture with four layers. The lowest layer is device layer that collects huge data in a varying format for varying IoT application domains. SDN gateways and routers constitute communication layer. SDN controller accounting and billing mechanisms are performed at computing layer. Service layer contains IoT services developed by service developers and operators.

Proposed architecture consists of SDN controller, gateway and routers, data processing and storage center, accounting, and billing center. Horizontal IoT solution given in the proposed architecture aims at supporting multiple IoT services (such as meter data gathering service for smart meters, alert notification service). The proposed architecture enables any kind of IoT services to be ported and provided immediately. SDN controllers in the architecture are responsible for data forwarding along with the processing of data and perform functions of equipment management, IoT service management, topology management, operation and maintenance, security management, and implementation of northbound/southbound interfaces. Gateway routers in the architecture do data forwarding in the networks. Data processing and storage center in the architecture contain the data attained from IoT devices and the controller gave sinks in the network that are stored in this module as per the orders.

Algorithms to process data are performed in the data processing and storage center component through the controller. Accounting and billing center in the architecture tracks the consumption of resources as IoT services take up computing and storage resources, unlike traditional networking services that consume network bandwidth. Instructions on how controller control gateways/ routers are given through northbound interface. The southbound interface is used to realize vital request response paths between controller and routers. In order, to validate the proposed architecture main modules are implemented and their performance is analyzed. Controller and forwarding function is implemented in POX configuring flow tables in routers/gateway with the instructions given by the control plane. Gateway/routers are implemented using open vSwitch. SDN northbound data exchange is implemented by JSON. SDN southbound interface is implemented using OpenFlow.

For the experiment, 1 controller, 11 gateways/routers, and 9 hosts are installed as virtual machines in a server inside a distributed architecture. RTT (Round-Trip Time) and packet loss rate for a case when IoT service exists and the RTT when new IoT service is introduced is observed. With the results, it is claimed that by using proposed architecture, it is possible that the new IoT services can be provided rapidly. The authors discussed their work superficially.

Pros and Cons. Proposed architecture does not address major management challenges. Detail working of the functional modules in the proposed architecture is not given. It is not clear how the components in the architecture will interact. In particular, it is confusing as to how the new IoT services

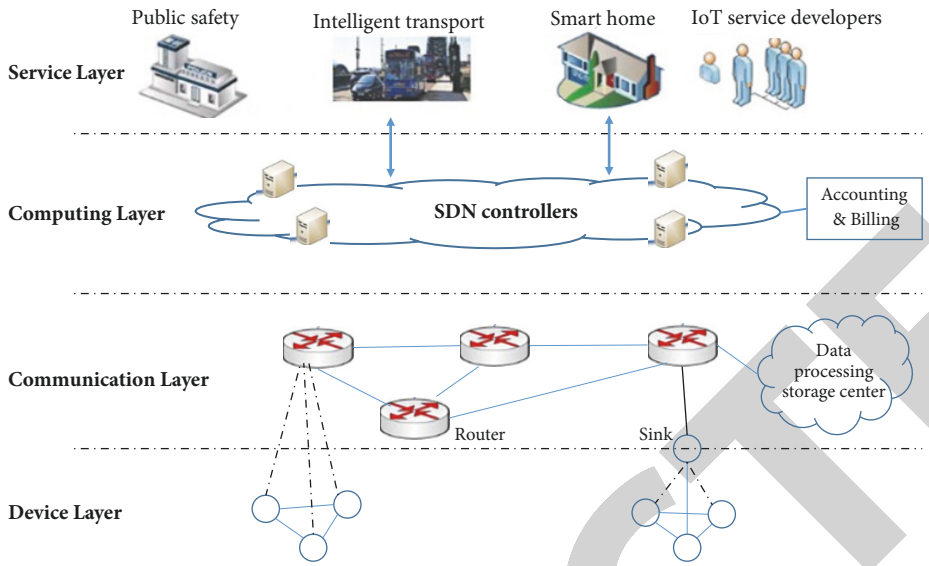


Figure 17: Proposed framework in [35].

will be seamlessly ported into the architecture. Security management is left as future work. In addition, authors plan to implement more functions and algorithms on SDN controllers and gateways such as calculating routing paths, which considers caching in gateways.

(11) Securing Networks Using Software-Defined Networking [21]

Proposed Framework and Critical Analysis. The authors of this paper [21] investigate how SDN offer novel ways to handle the principal security requirements. Existing SDN-based security research is divided into two branches that one of them focuses on securing the network and the other on provisioning security as a service. Further classification of current work has two main tracks such as threat detection, remediation and network correctness. Along with this, the authors highlighted possible challenges and various directions that can be opted for security in SDN.

SDN is presented as security as a service to secure the network with novel security functionality such as specialized network management. Previous research is classified as the research on securing the network and provisioning security as a service. Policies consolidation within central controller boosts the steadiness of configuration to avoid attacks. Network state centralized eases intrusion detection and response in the vigorous manner that alienates attack. As SDN provisions centralize traffic, monitoring packet inspection on device traffic is possible. This enables developers to write applications that can use data mining and machine learning techniques making it possible for swift recognition of security threats, while SDN allows in real-time programmatic capabilities enabling a vast variety of vigorous responses such as alarms and traffic redirection for forensics. There is the possibility of making errors by the human, which is leading to an emerging research topic of automated techniques to verify network consistency in SDN. Varying users or multiple controllers in the same domain, which potentially lead to conflicting rules and access control violations, encounter badly configured network in SDN as the network controllers are used.

Pros and Cons. Basics of security configuration with SDN are ensured by centralizing the control plane and flexible policies. SDN security as a service enables supplementary network security measures. Challenges and future directions of research in security-oriented applications are securing SDN, federating heterogeneous networks, coupling overlays, and underlays and trends beyond OpenFlow and network function virtualization. There are no proposed architecture and implementation but only thoughts on provisioning security with SDN paradigm are discussed.

(12) SDN-Based Application Framework for WSAN [36]

Proposed Framework and Critical Analysis. SDN has the ability to effectively control the communication infrastructures and reducing processing loads of forwarding nodes. Thus, by applying SDN for Wireless Sensor and Actuator Networks (WSAN), optimal control and scheduling are possible for the whole process of cooperative communication and execution of the task with the help of relevant nodes. This leads to improvement in efficiency/reliability with decreasing consumption of energy in WSAN. The WSAN elements usually work using a distributed paradigm that complicates sensing process. By applying SDN to WSAN the authors [36] claim to achieve effective control that improves reliability at the same time decreases energy consumption in WSAN.

WSAN framework, as shown in Figure 18, which is comprised of three layers, is presented with control plane sandwiched with data plane and application plane. SDN controller makes a decision, instead of distributed WSAN, for controlling packet forwarding. The traditional protocol stack of WSAN has a cooperation plane interacting with

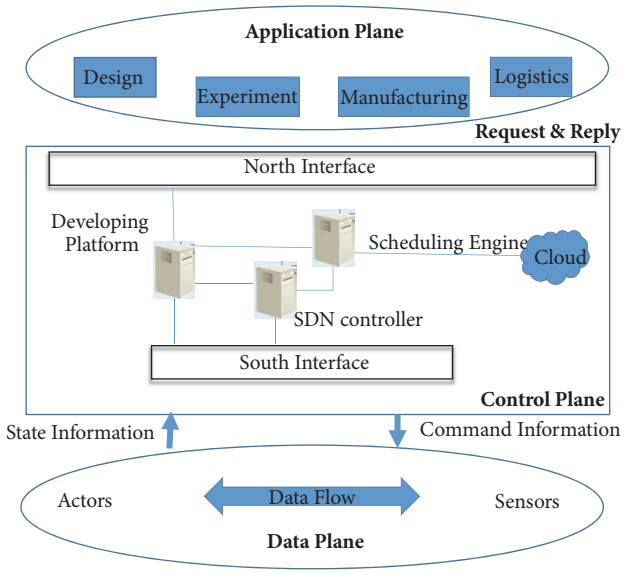


Figure 18: Proposed framework in [36].

five protocol layers (application, transport, network, medium access control, and physical) and making a decision as the elements respond to commands. The authors proposed an idea of a protocol stack for WSAN equipped with SDN working principles. Functions at network layer are divided that are categorized as the local and global function at the network layer. The controller implementing network management functions collects suitable parameters for other protocol layers. In order, to realize centralized SDN control functions effective management and optimization of distributed WSAN environment is required that addresses the issues of mobility, energy consumption, reliability maintenance, and security heterogeneity. Apart from the proposed architecture, shown in Figure 16, for managing WSAN, the authors proposed software-defined scheduling method for the tasks submitted to WSAN. Scheduling engine is implemented in the control plane that works with SDN controller to provide scheduling service. In contrast to traditional WSAN, SDN control plane responds to commands from sensors and actuators in the data plane. Hence, the SDN-based control plane can help in making decisions as regards to commands from devices. The authors discussed their work superficially.

Pros and Cons. Proposed architecture addresses major challenges of IoT that are handled by the adapted protocol stack with SDN. These challenges are mobility management and energy management that are managed by gathering protocol parameters collected by the SDN controller across the networking protocol stack layers in a cross-layer manner. However, security and fault tolerance is not addressed in the proposed architecture. This cross-layer solution to manage IoT network come at the cost of parameter passing across layers that adds an extra overhead of passing messages. At the same time, the proposed architecture requirement yields major changes in the protocol stack of the network in order to achieve an SDN adoption in WSAN. As the control plane is a middle layer that accepts a request from the application plane and data plane the authors did not consider the load on the controller and the response time of the request being fulfilled.

(13) Black SDN for the IoT [37]

Proposed Framework and Critical Analysis. The authors in [37] used black networks with SDN controller as a trusted third party to propose a secure SDN IoT network architecture. Security in black SDN is provided by encrypting header and payload at the network layer mitigating a range of attacks including traffic analysis/inference attacks. Black networks secure metadata and payload within each layer in the IoT protocol stack mitigating a broad range of both passive and active attacks that happen due to authenticated and secure communications at both link layer and network layer.

Black SDN for IoT consists of a star, mesh, and wireless IoT network that communicates with an IoT adapted SDN controller. SDN controller's effectiveness for security and routing are presented for three topological scenarios broadcast on star network, synchronized mesh network, and unsynchronized mesh network. Black network simulator is designed by the authors to simulate network performance for various topologies, sleep patterns, and implement the functionality of network layer. Impact of black network layer on the routing performance of the network is observed where black network displays optimal overhead. Node coverage is plotted against network performance when certain nodes are asleep and good network performance is observed by the proposed solution. Broadcast routing is investigated in black networks where sender and receiver metadata, which includes sender and receiver information, is encrypted. Evaluation of the proposed architecture also suggests increase security in the IoT architecture and network performance. The authors discussed their work in a very superficial manner.

Pros and Cons. Detailed algorithms and working of the components in the proposed architecture are not given. Black SDN provides confidentiality, integrity, authentication, and privacy. It is not clear how black SDN is implemented in IoT, as detailed working is not given by the authors that lead to confusion as for how SDN controller is acting as the third party for provisioning security in IoT. In addition, detailed working of black network simulator designed by the authors is not discussed in detail that leads to confusion as to how black SDN is realized for IoT. Detail mechanisms and protocols that are provisioning security are not given. Future directions as highlighted by the authors will focus on more complicated routing mechanisms and so will focus on energy efficiency.

(14) Software-Defined Security Service [38]

Proposed Framework and Critical Analysis. Motivated by the lack of a comprehensive resource management framework for IoT, the authors in [38] have highlighted security requirements of IoT that needs to be addressed in order to ensure secure data delivery and defense against attacks in IoT. Apart from security management, the authors also

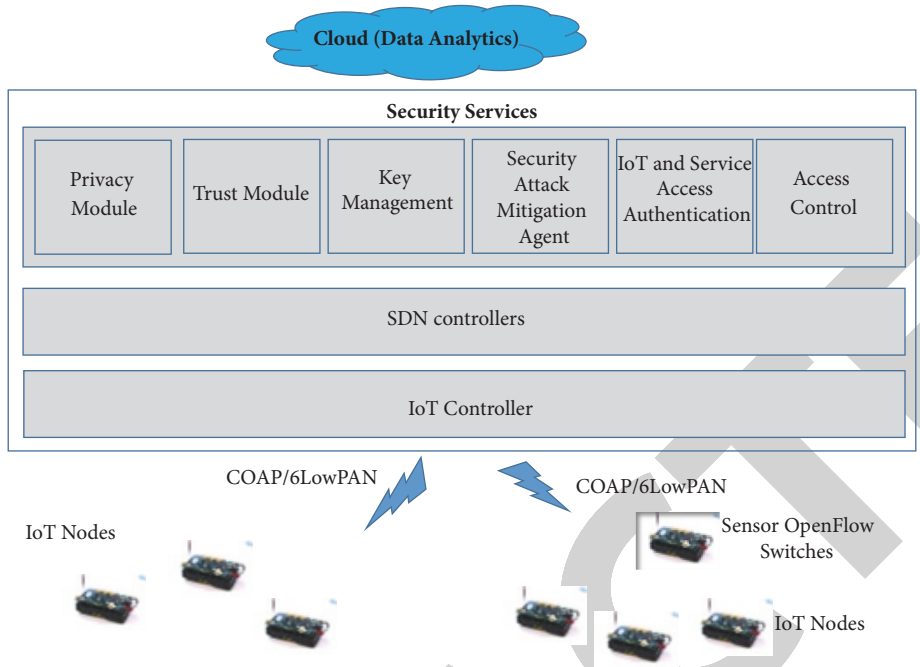


Figure 19: Proposed framework in [38].

identified other management issues in IoT that needs to be addressed by the management framework for IoT. These are fault tolerance, energy management, and load balancing. In [38] the authors proposed an SDN-based IoT framework, as shown in Figure 19, for security service provisioning that has three main components IoT controller, SDN-based security controller, and sensor OpenFlow switches. IoT controller is a middle tier that collects information from IoT devices across IoT network. SDN-based security controller runs security services that interact with SDN controller to provision security over IoT network. Modules that are privacy, trust, key management, security attack mitigation agent, IoT, and service access authentication and access control modules provide these security services. Although the authors have done a tremendous effort of developing a comprehensive framework, now no evaluations of the proposed framework are presented. As there will be security modules that will run over SDN controller, it is expected to incur overheads, which needs to be investigated and evaluated.

Pros and Cons. Security over IoT is expected to improve dramatically with the proposed framework. As security management has received very little attention until now, this work is a tremendous contribution. In particular, the potentials offered by SDN paradigm are applied to solve the security management issues in IoT. At the same time, security modules are expected to incur overheads that need to be investigated.

(15) SDN-Based Refactored Middleware for IoT [39]

Proposed Framework and Critical Analysis. In order to extend the support of SDN, the authors extended their previous work [41] to include SDN technologies. With the refactored middleware supporting SDN, management of things in IoT is claimed to be improved. The authors [39] proposed a refactored REMOA middleware [41] with SDN support as shown in Figure 20. Complex network services implemented over AP is now distributed among servers with more computing power. Three of the modules of REMOA are refactored as a transparent proxy module, things management, and gateway. AP and service servers perform the task of the transparent proxy module. AP forwards packet based on flow rules. Data collected by things are sent through IPSec tunnel to services. Things management that used to be based on SNMP is based on OpenFlow counters. Counters are available through APs by Things Flow application that makes it available with a timestamp that indicates when it was retrieved. Services at the same time retrieve counters, which are stored in Thing Flow that implements management mechanisms to things. Gateway is packet forwarding featured by AP. Forwarding is defined by OpenFlow rules. Middleware capabilities are extended that enable us to provide multiple services in a single AP. No evaluation of the proposed work is presented. It is not clear what will be the overhead of the messages between the modules after refactoring the REMOA middleware. In addition, what will be the complexity added by the new modules in the whole architecture?

Pros and Cons. By refactoring the middleware, development of each service is not impacted by AP constraints. At the same time, developers can use any programming language, library, framework, or programming techniques for developing features for services, which are complex or impossible to develop on AP. It seems by decomposing the middleware into heterogeneous components that communication complexity has increased as more number of messages will be exchanged among the components.

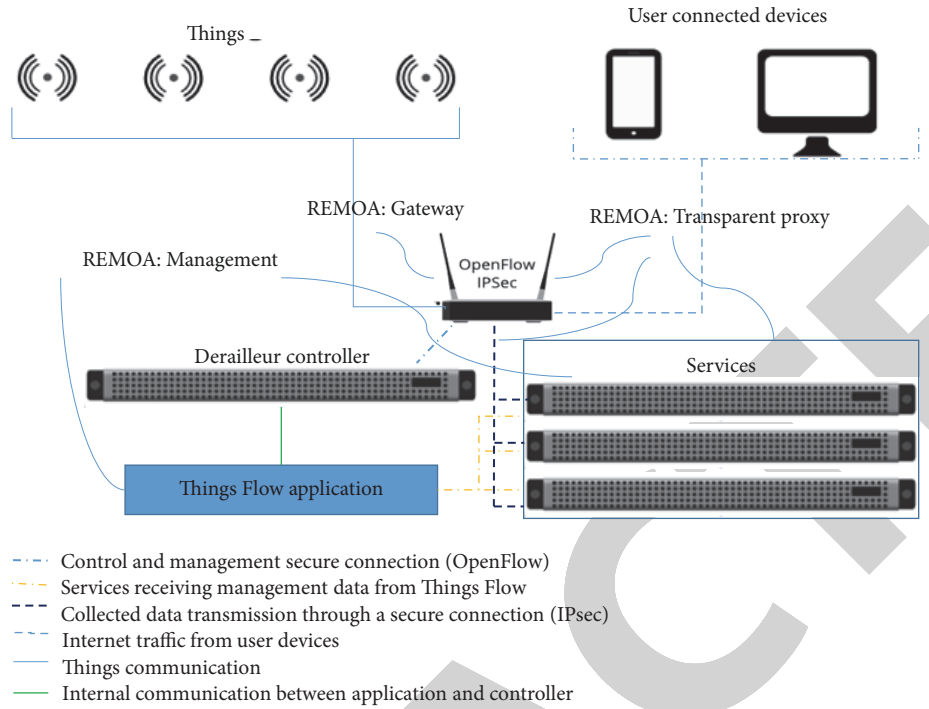


Figure 20: Proposed framework in [39].

4.3. SDN Architecture and OpenFlow Adaptation Based Management Framework. In order to adopt SDN paradigm for IoT, there are efforts where researchers have adapted SDN architecture and OpenFlow for IoT that have constraints and requirements different from the traditional network. Such work is classified as SDN architecture and OpenFlow adaptation based management framework.

4.3.1. Sensor OpenFlow. Background, motivation, and discussion of proposed framework with critical analysis on [42] are given as follows.

Background. Since the advent of WSN, they are conceived as application specific. The applications specific WSN are subject to resource underutilization that is due to different application-specific WSN deployed on overlapping terrain that can be achieved by a single versatile WSN. At the same time, heterogeneous vendors develop WSN in isolation deprived of reusing common functionalities to accelerate prototyping and production.

Motivation. It is hard to change policies in WSN. Another problem with WSN is that it is hard to manage. Motivated with this fact the authors take a step forward to propose an SDN-based architecture for IoT by adapting OpenFlow protocol.

Proposed Framework and Critical Analysis. Authors in [42] have proposed software-defined WSN, shown in Figure 21, which have a clear separation between the control plane and data plane with sensor OpenFlow as an accepted communication protocol enabling communication between two planes. The data plane has sensing nodes that perform packet

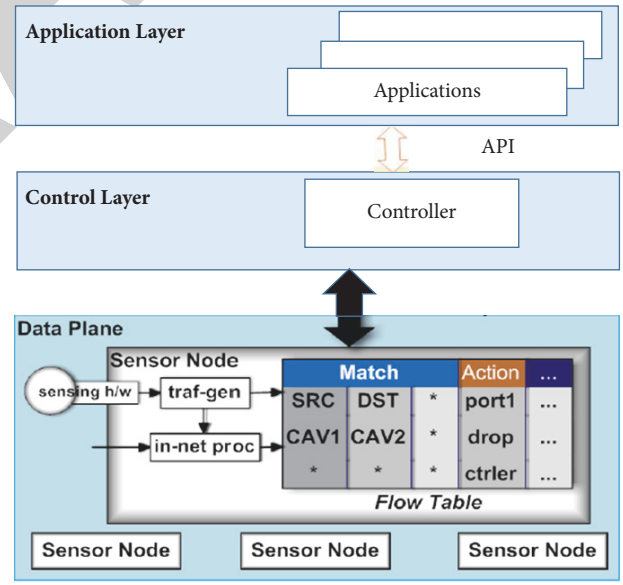


Figure 21: Proposed framework in [42].

forwarding based on the flow table. Network intelligence is centralized at the control plane. It is possible to make network programmable that manipulates flow table on each sensor. Incorporating SDN in WSN solves inherent problems and makes WSN as versatile, flexible, and easy to manage. OpenFlow is designed for a wired protocol and hence needs to be adapted for WSN that the authors proposed as sensor OpenFlow.

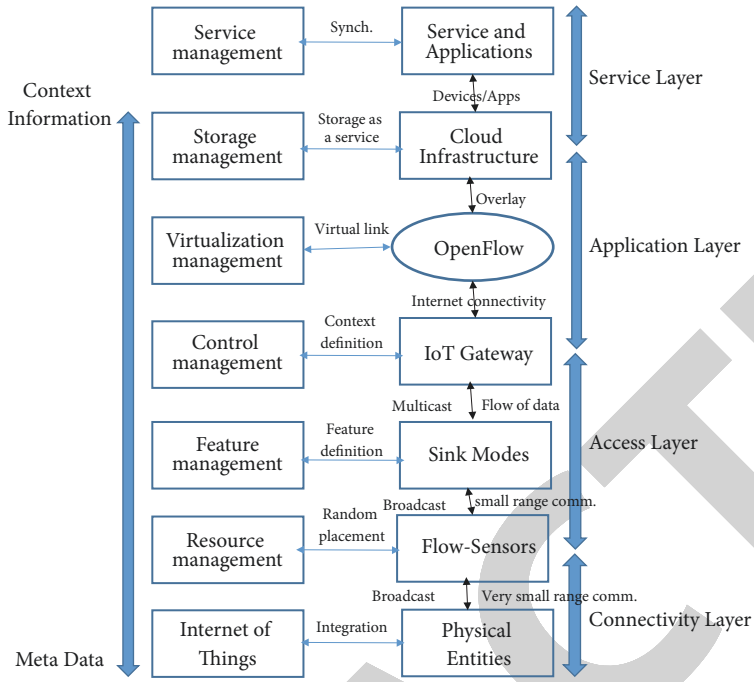


Figure 22: Proposed framework in [43].

Sensor OpenFlow control channel is similar to an OpenFlow control channel. In SDN OpenFlow, a channel is out of a band that is not realistic for WSN and sensor OpenFlow channel is hosted in the band that means WSN have to carry additional control traffic. This is further exacerbated by the fact that control traffic in WSN tends to be largely due to the high network dynamics, which means that WSN can get overloaded. WSN needs to process data-by-data aggregation to conserve network resource and bandwidth that is absent in SDN. The authors give two solutions for flow creation in sensor OpenFlow; the first solution is to rewrite flow tables. Secondly, IP can be augmented with WSN. Similarly, for the control channel, if someone chooses non-IP solution, the sensor OpenFlow channel is equipped with superimposing transport protocol precisely over wireless sensor networks. If an operator chooses WSN with IP, then sensor OpenFlow channels are self-supplied. Traffic-gen module is added to each sensor (as an interrupt routine) that can run in blocking, callback, or round robin manner. This module is responsible for handling sensory data generated that consists of two steps: reading data from the hardware and converting the data if needed. In-net proc module is added for in-network processing. If the packet does not need the processing the module simply passes packet intact to the flow table. The authors discussed their work superficially.

Pros and Cons. Proposed architecture does not give any details of how it will address major challenges of managing IoT. It only addresses how SDN principles can be implemented in current IoT network. No implementation and evaluation of the proposed architecture are presented. Overhead of adapting OpenFlow for IoT requires the authors to not consider investigation as practical feasibility. Inherently, IoT is constrained in resources; it is desired that adapting OpenFlow for this environment should be lightweight. An extensive evaluation is needed to assess the computational complexity of OpenFlow in IoT.

4.3.2. Framework for IoT Virtualization via OpenFlow [43]

Background. IoT industry is looking forward to virtualizing and setup common platform for pervasive computing with context information to be shared [43]. This context information will be distributed to the huge amount of physical entities. At the same time, such context information will create collaboration among multiple services irrespective of centralized system. In other words, it can be said that it follows the distributed structure.

Motivation. Motivated by a need for a virtualized framework for IoT devices, the authors in [43] proposed a conceptual framework where the devices generate context information of raw data captured from surroundings. Proposed framework standardized IoT infrastructure through that it can receive e-services based on context information, which at the same time leaves current infrastructure unchanged.

Proposed Framework and Critical Analysis. The proposed framework [43], as shown in Figure 22, consists of four layers. These layers establish a generic framework that does not change the current network infrastructure and instead forms interface among services and entities through network virtualization. The four layers are connectivity layer, access layer, abstraction layer, and service layer. Connectivity layer has physical devices in the framework and their

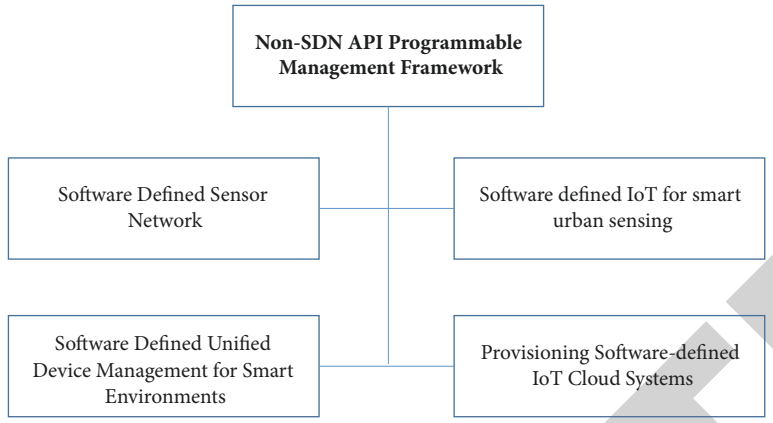


Figure 23: Non-SDN programmable frameworks.

interconnection. This layer checks the availability of physical resources of all devices and networks involved in network infrastructure. Access layer consists of topology definition, network initiation, and the creation of domains. In addition, this layer also includes connection setup, intra/inter domain communication, scheduling, packet transmissions between flow-sensors, and IoT gateway. Important characteristics of OpenFlow are to add virtual layers with the present layers leaving established infrastructure unchanged. Such virtual link can be created for different networks and a common platform is developed for a various communication system. Storage and management layer involves data storage and supervision, software services and business management, and operations. All of them are included in one layer that is the service layer. Performance evaluation of the framework is performed in three different scenarios: internetwork communication, intradomain communication, and interdomain communication. In all of the scenarios, a significant gain in throughput is observed within the conceptual framework.

Pros and Cons. It is no doubt the proposed framework combines Internet of Things and OpenFlow through that infrastructure as a service can be attained and cloud computing can be exploited. On the other hand, an effort of virtualizing IoT devices with the conceptual framework proposed in [43] is composed of four layers. With so many layers, too much processing is expected on the packets owing to the proposed conceptual framework. The authors discussed their work in a superficial manner.

5. Non-SDN API Based Programmable Management Framework

In contrast to SDN-based management framework, there are efforts of designing custom APIs for managing IoT programmatically, which do not adopt SDN principles. As SDN also proposes its own APIs for managing its network programmatically. We discuss the most important and the popular as non-SDN API based programmable management framework so it has categorized in Figure 23.

5.1. Software-Defined Sensor Network [44]

Background and Motivation. "Sensing as a service" is introduced as a new paradigm integrating WSN resources into cloud computing. To realize sensing as a service software-defined sensor network (SDSN) plays an important role in this service paradigm. In the study of [44], the authors claim that the SDSN has great potential, but that there is no major work in this area.

Proposed Framework and Critical Analysis. Motivated with software-defined sensor networks (SDSN) the authors in [44] proposed SDSN architecture, shown in Figure 24. The proposed architecture incorporates a novel sensing programming services that will enable programmable sensor nodes. The sensor nodes configured with a particular kind of activity that they perform as temperature sensing, pressure sensing, data aggregator, and cluster head roles. The functions to be activated on the sensor node are described in terms of roles that are generated by the compiler based on a scenario description, which describes the functions. Appropriate sensor nodes are delivered programs via wireless communication in order for the sensor nodes to behave as desired. Logically, SDSN comprises the physical, networking, and application layers. Key enabling technologies of SDSN are Software-Defined Radio, SDN, wireless sensor operating system, over the air programming technique, and field programmable gate array technique.

Pros and Cons. The proposed architecture is very abstract and needs detailed working along with algorithms. Implementation and evaluation are not presented. Description of layers in SDSN is very vague; main functionalities of each layer are not given. As future work, implementation and feasibility of the proposed framework are required. In addition, investigation of how the proposed architecture will address major challenges of managing IoT is needed.

5.2. Software-Defined IoT for Smart Urban Sensing [45]

Background. Increasing number of inhabitants in the city is increasing day by day that result is the urban straining

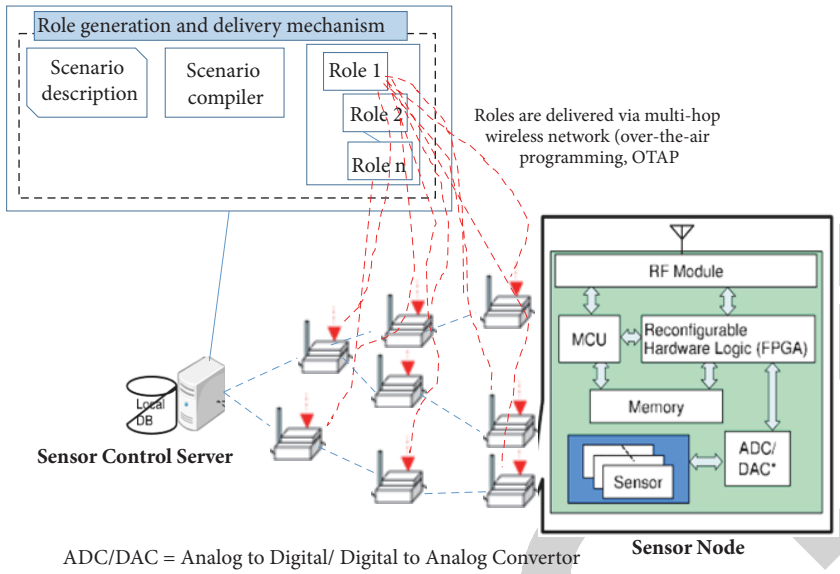


Figure 24: Proposed framework in [44].

of transportation, security, healthcare, and environmental pollution. Urban sensing is one of the profound solutions to solve this problem. IoT can make urban sensing a reality. Current IoT architecture lacks concrete resource management framework, especially for urban sensing application.

Motivation. The centralized control offered by SDN can address the issue of managing urban sensing application. Addressing the problem of inapplicability of traditional Internet protocol in IoT, the authors in [45] presented software-defined IoT architecture for intelligent urban sensing.

Proposed Framework and Critical Analysis. The authors in [45] discussed the state of the art in IoT urban sensing. Moreover, the SDIoT architecture, shown in Figure 25, is proposed that comprises physical infrastructure, control, and application layer. Urban sensing application can personalize transmission, data acquisition, and processing by using well-defined service APIs data transmission, data acquisition, and processing service. Data acquisition service provides APIs specifically for applications specifying their data requirements. Data transmission service provisions API to specify destination and QoS parameters. Data processing service enables applications to specify required resources. Well-defined API services are implemented at control layer that are enabled with northbound and southbound interfaces. Mobility management is provided by the control layer, which supports mobile sensor platform from one gateway to another. Benefits of SDIoT are illustrated with quantitative analysis over 5∗6 rectangular urban areas taken on three type of sensor platforms fixed sensor platform, user smartphone-based sensor platform, and mobile vehicle-based sensor platform. Five urban sensing applications street view, weather monitoring, noise monitoring, environmental monitoring, and dust monitoring are used for evaluation. 2500 vehicles generate data with constant probability set to 1/1000. Seven forwarding devices connect three data centers. Five applications process the data on VMs and results are shown by visualization software. Proposed architectures have shown efficient management of physical infrastructure. Traffic is distributed equally over multipaths lessening maximum link load.

Pros and Cons. Detailed algorithms of the proposed architecture are not presented. It is not clear how the components interwork with each other. The proposed architecture addresses mobility management, while fault tolerance and security are not considered. The authors do not give an extensive description of how mobility is handled by SDIoT when the handover process occurs. Clear details and working of the northbound and southbound interface to work with three services of data transmission, data acquisition, and data processing need to be given. As a future work, a study is needed as to how the proposed architecture can be comprehensive management framework for IoT.

5.3. Provisioning Software-Defined IoT Cloud Systems [46]

Background. Contemporary techniques of IoT cloud systems virtualizing physical sensors focus on data and device utilization. The authors argue it in [46] that designers and operation manager have to face challenges to make possible huge challenges in order to provision and serve large-scale IoT system. In order to gain the most of cloud computing the IoT cloud system should virtualize IoT resources and IoT capabilities that are encapsulated using an application-programming interface (API) with varying levels of abstraction. API enables central management of IoT devices.

Motivation. Motivated with the need of introducing programmability in IoT network, the authors in [46] proposed the idea of software-defined IoT units, a new idea of IoT cloud computing that envelope IoT resources and capabilities accessible using API. In order to make it feasible,

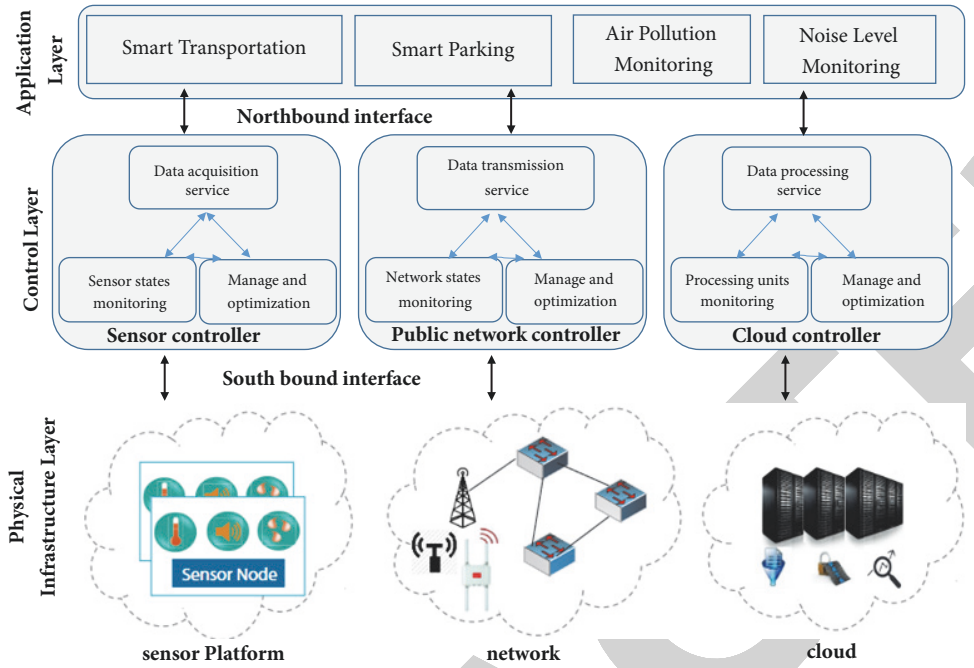


Figure 25: Proposed framework in [45].

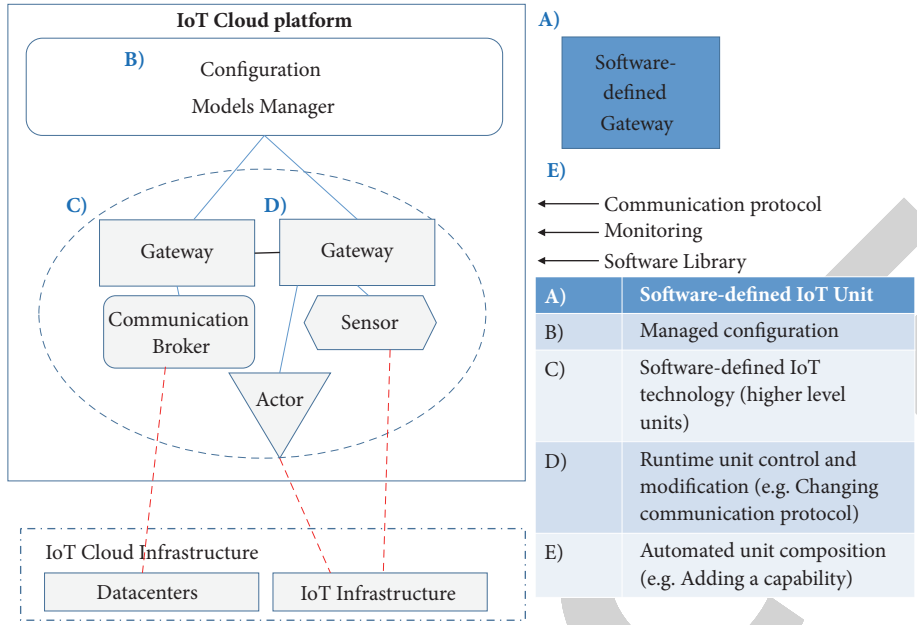
the authors highlighted that such large-scale and geographically distributed setup have requirements of tools that will automate development, provisioning, the operation of processes, and reconfiguration of policies dynamically and centrally.

Proposed Framework and Critical Analysis. Software-defined paradigm is explained by the principle of abstracting low-level components managed by API. Software-defined IoT system consists of resource components hosted in the cloud reconfigured in runtime.

IoT resources, gateway, and capabilities are defined as software-defined IoT units. Main concepts of software-defined IoT units are composed of API encapsulation, fine-grained consumption, policy-based specification and configuration, automated provisioning, cost awareness, and elasticity support. Units encompass within them functional aspects and nonfunctional aspects of the IoT resources. The proposed idea maps the virtual resources with the underlying infrastructure. The concept of the unit prototype is usually constructed using OS-level virtualization or kernel supported virtualization. The governance API that is exposed by units performing control operations at runtime comprises commencing or concluding the unit. Internal topological structure of software-defined IoT unit is classified into atomic, composed, and complex software-defined IoT units. Software-defined IoT cloud system is provisioned by self-operating software-defined IoT units and centrally managing configuration model and policies.

The proposed framework architecture is shown in Figure 26, which have presentation layer enabling various configuration models to be specified. The main functionality of framework is found in cloud core services. Policy processor and unit management services make use of the prototype for composing and managing the units. Initialization manager configures and composes more complex unit. Cloud system wrapper enables deployment across various cloud providers. Deployment manager manages and distributes the dependency references. Units' persistence layer caches and manage software-defined IoT units. The authors discussed their work superficially.

Pros and Cons. Proposed architecture does not address security, fault tolerance that is a major challenge of a framework managing IoT. Sensor and actuators are managed in a cloud of IoT by the software-defined gateway. The authors do not address scalability and reliability of software-defined gateway. As the gateway is susceptible to fail, how will it be handled by the proposed architecture? Furthermore, as the number of IoT nodes joining the cloud increases how the joining process will be handled by the gateway to ensure smooth operation of the architecture? The authors do not discuss the detail of protocol and API to enable communication between software-defined gateways. A prototype of the proposed framework is implemented in an OpenStack cloud and Ubuntu 12.10 cloud image is used. In order to make a fleet management system for evaluating the proposed framework a number of software-defined IoT units are developed. The proposed framework shows promising results claimed by the authors but no results are presented to justify it. Apart from the evaluations discussed, extensive evaluations are required that measures the network performance as required for any management framework for IoT. For future, the authors look forward to extending the prototype in various dimensions.



A)	Software-defined IoT Unit
B)	Managed configuration
C)	Software-defined IoT technology (higher level units)
D)	Runtime unit control and modification (e.g. Changing communication protocol)
E)	Automated unit composition (e.g. Adding a capability)

Figure 26: Proposed framework in [46].

5.4. Software-Defined Unified Device Management for Smart Environments [47]

Background. Smartness is now part of various IoT applications. The smart environment consists of heterogeneous types of embedded sensing and actuating devices customized to run on specific protocols and technologies suited to the needs of IoT [47]. Smart applications can be augmented by cloud computing that provides distributed and scalable computing infrastructure on demand. Cloud can meet the needs of IoT when massive computing power to manage billions of IoT devices is needed.

Motivation. Motivated by the need of smart management platform for devices, the authors in [47] proposed a smart device management platform.

Proposed Framework and Critical Analysis. The proposed smart device management platform [47], as shown in Figure 27, consists of IoT controller that drives things to attach virtual control unit (TAVCU). IoT controller provides a unified common management user interface based on Northbound REST API. Applications use a controller to aggregate network intelligence. They also run algorithms to run analytics and use the controller to distribute new rules across the network. Things attach fabric (TAF) is a programmable virtual mesh that has multiple TAVCU. The controller is modular that has topology manager with maintains devices, their capabilities, reachability, etc. Device manager generates topology database for the topology manager. Statistics manager maintains statistics and counters that are related to usage. Trust manager handles security keys and infrastructure related security and trust. No evaluation of the proposed management framework is presented. The authors discussed their work superficially.

Pros and Cons. Proposed framework enables smartness in IoT network. However, it seems that the framework will have overheads of communication among different modules. These overheads may introduce the unnecessary delay in communication between application and IoT devices. At the same, single controller will cause the load to be concentrated in IoT controller. Apart from topology manager and device manager, there is a desire need of load manager that will balance the control and data traffic within the IoT network with smart device management framework.

6. Comparison Results and Evaluation

IoT technology every day gets worse. The security issue is the first important challenge and open research issue in these structures that are in wireless and Internet-based communications together or servers for different aims such as big data analysis, monitoring, and smart city; security is always a very important issue in wireless communications. However, the security issue has so much increased because they have a working mechanism that can more directly affect human lives. Other important research topics are fault tolerance, energy management, and load balancing. Because these systems are often distributed and heterogeneous systems (of course, they can be in centralized architecture and consist of homogeneous devices). Indeed, they must be stable and continue to work in the task environment even if some devices are disabled or failed. Therefore, the fault tolerance of system is critical issue. Other major issue is energy management because the devices may have to work for long periods without the possibility of any charging. The system resource management is other significant criterion. Therefore, there must be absolutely balance between the uses of resources. For example, you have to spend too much energy to reach a high data reliability ratio, so, these parameters

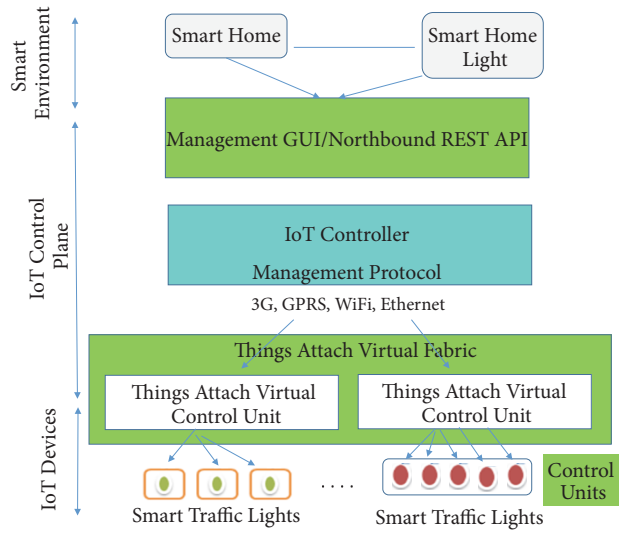


Figure 27: Proposed framework in [47].

have tradeoff together. These types of parameters are most important and difficult to balance between resources. Both, these requirements can be provided inside of the network, also, inside of the server and software layer. Therefore, all studies examined in this paper will be evaluated on these four parameters. In addition, these parameters are the research topics on the agenda and in the future will be very important, specially security.

Management challenges addressed by the proposed framework are summarized as follows. Figure 28 summarizes what functionalities are supported by the existing proposals on SDIoT management framework by focusing on the four parameters, while Figure 29 discusses pros and cons of existing proposals on SDIoT management framework only. Summary from Figure 28 is discussed as follows. Our methodology of the evaluation of the paper is based on their own data on all the methods described. We examined the pros and cons of each. Moreover, we took care of the four parameters (fault tolerance, energy management, load balancing, and security management) in all presented work-studies with their advantages and disadvantages, so they are summarized in Figures 28 and 29.

Fault tolerance is one of the important factors and we used it in our evaluation based on the approach that is explained in [29] by partitioning the region with IoT infrastructure into a portion called share. Scheduling algorithm balances the traffic in a different share of the IoT network hence provisioning load balancing in IoT network. In addition, fault tolerance is provided by identifying faulty nodes in the share and then rerouting the traffic from defected nodes to other nodes running normally in the IoT network. Apart from [29] other has classified in SDN-based management framework lacks in fault tolerance factor.

Energy management is another crucial parameter in IoT. SDN controller in the SDN paradigm can assist in scheduling the flows in the network resulting in energy conservation. Centralizing the view of the network assists in energy efficient data aggregation. Hence, as a future research direction, it is essential to explore SDN paradigm and its potentials as it provides a centralized network view, which makes it a hot candidate to solve complex management issues of IoT. None of the proposed frameworks addresses energy management issue in IoT.

Load balancing of traffic in IoT can be aggravated by SDN principle based techniques and architecture. The general definition of this parameter is a feature that distributes network traffic among multiple servers or virtual machines within a cluster to avoid overloading any one host and improve performance. SDN controller can enable the whole view of the IoT network that can be monitored. Centralizing control can help in balancing traffic across IoT. Load estimation techniques and algorithms at the controller can assess the predicted load in IoT that can influence the flow traffic in IoT network. SDN controller can propose routing decisions that are forwarded to IoT nodes to balance traffic in the network. Detailed investigation of load estimation algorithms at the controller is required. Probabilistic techniques can be used to predict the load on the IoT network and nodes. There is a need to propose algorithms and techniques, which use current load statistics to routing decisions to neutralize traffic in the IoT network. Partitioning the region with IoT infrastructure into a portion called share is suggested in [29] by the authors. Share enables load balancing in the network. Apart from [29], other classified SDN-based management frameworks lack in load balancing.

Security provisioning over IoT can be found in proposals of the framework that employed network function virtualization to manage the network. This is probably the most popular and important parameter in the field. There are many IoT-based arenas (such as vehicle ad hoc network, privacy in personality data analysis systems, and government-based intelligent systems). Largely, security service provisioning over a low constrained IoT network is a daunting task as traditional security protocols and mechanisms are inapplicable in the domain. In [3, 22] minimal security service provisioning exists in the framework where attack mitigation, key management, and other essential security services are not handled by the proposed architecture. SDSec is employed in [26] to provide security in IoT network and bare minimum discussion can be found in other middleware-oriented frameworks [21, 27, 37], while none of the security provisioning exists in API based programmable management framework, SDN architecture, OpenFlow adaptation based framework, and message-oriented publish-subscribe model-based framework. In results shown in Figure 28, it is evident that all of the proposed frameworks are centrally reconfigurable network exploiting paradigm of SDN. This property of SDN makes it a hot candidate to provide management services across the IoT network. Fault tolerance, load balancing, and energy management require rigorous programming from the central agent. SDN controller in SDN paradigm has a centralized view of the network that can enable reconfiguration adapted to network snapshot at a particular instant of time. Future work section in our paper gives details as to how emerging SDN paradigm will address the management challenges of IoT. Most of the existing proposals lack implementation and

		Existing works	Security Provisioning	Fault Tolerance	Energy management	Load Balancing	Centrally Reconfigurable Network
SDN-based IoT-Management Framework	Network Function Virtualization based Management Framework	[3, 22]	Minimal Functionality	N	N	N	Y
	Middleware with SDN principles for IoT	[26, 38, 27, 33, 34, 28, 31, 35, 21, 30, 6, 39]	Minimal Functionality	N	N	N	Y
		[37]	Exclusive Functionality	N	N	N	Y
		[29]	Minimal Functionality	Y	N	Y	Y
	Message Oriented Publish Subscribe Middleware based Framework	[23–25]	N	N	N	N	Y
	SDN architecture and OpenFlow adaption based management Framework	[42, 43]	N	N	N	N	Y
	Non-SDN API based Programmable Management Framework	[44–47],	N	N	N	N	Y

Figure 28: Comparison of SDIoT and non-SDN-based current frameworks (Yes = Y, No = N).

evaluation. Figure 29 discusses strength and weaknesses of all the frameworks.

7. Conclusion and Future Works

In this paper, we have presented taxonomy and details of all the existing works of Software-Defined Networking based management framework for IoT. We have classified the management framework as network function virtualization based management framework, middleware-based management framework, API based programmable management framework, SDN architecture and OpenFlow adaptation based management framework, and message-oriented publish-subscribe model-based framework. Details of all the work classified according to the taxonomy are presented. There is no work-up till now, which has provided a concrete management solution for IoT network. Most of the works lack the detail working of functional components in the proposed architectures for managing IoT network. Few of the proposals until now have evaluated their proposed architecture extensively proving their effectiveness. Comparison of all these efforts is given in terms of requirements of IoT management framework, which are security provisioning, fault tolerance, energy management, and load balancing. Few of the proposals [3, 22, 26] provide minimal functionality of security service provisioning, while there is wide room to address the issue. One such work, which has addressed the issue of security service provisioning in IoT comprehensively, can be found in [38]. At the same time, there exist few proposals [29] to address load management and fault tolerance in IoT network, but largely most of the existing works do not address the issue, while proposing SDN-based management framework.

In future works, as explained in detail in the evaluation section, fault tolerance, energy management load balancing, and security constitute important research areas. In addition, they are both on the system side and on the software layer side, especially security requirements. In our future works, our goal is to focus on these needs at the software side and suggest new solutions. Designing a management solution for IoT is not a trivial task. SDN paradigm offers potential to design a comprehensive management solution that enables reprogramming of the devices in the network. As discussed, management challenges, which require to be addressed, are of fault tolerance, energy management, load balancing, and security in the network. Efforts in adopting SDN principles for constructing management solution have already been initiated but none of the solutions at the moment address most or all of the challenges needed to be addressed by the solution for IoT. The proposals, which exist in the literature up until now lack, detail working of functional components in the proposed architecture. In addition, only a few proposals are extensively evaluated, while most of them lack prototype implementation and evaluation. For future work, a deep investigation is needed as to how the centralize controlled IoT network by SDN controller can recover faulty nodes in the network. Also, centralize view can help in managing energy of the nodes by balancing traffic in the network. Security provisioning is an essential requirement especially in IoT as it is used in various utility applications. Network status view by the SDN controller can assist in provisioning security services such as attack mitigation, privacy, and lightweight key management for IoT. Hence, security service provisioning by using SDN principles on the Internet of Thing network needs in-depth investigation. Details of what is needed to be addressed in provisioning IoT management functions

		Existing works	Pros	Cons
SDN-based IoT-Management Framework	Network Function Virtualization based Management Framework	[3]	Virtualization decouple hardware from network functions lowering cost	Proposed architecture is generic
		[22]	Network functions are virtualized, which run heterogeneous processes	Overhead of virtualizing the functions
	Middleware with SDN principles for IoT	[26]	Multiple type of SDN controllers simplifying network management	Integrating various SDN supported controllers in resource intensive
		[37]	A comprehensive framework that manages security over IoT network	Proposed framework does not address to other management challenges of IoT such as load balancing or fault tolerance
		[38]	Proposed middleware is expected to support multiple services within a single home network, which shares the same network infrastructure	The middleware is composed of multiple components, which means in order to perform its operation. There will be numerous messages that will be exchanged among the components
		[27]	Stateful solution for IoT	Management challenges addressed by SDN wise is not clear
		[33]	Compatible central architecture according to the proposed system	Synchronization and coordination of the central and local controllers are not addressed
		[34]	Intelligent services can be provided by the proposed software defined IoT (SDIoT)	Detail working of the API is not given
		[28]	Proposed architecture enables negotiation of IoT services parameters	Details of the functional components are not given
		[29]	Proposed architecture address fault tolerance and load balancing	Architecture have multiple controllers, which leads to consistency and synchronization problems
		[31]	None	Idea is very abstract
		[35]	Proposed architecture enables novel IoT services	Major management challenges of IoT are not addressed
		[21]	None	Lacks in implementation and prototype for evaluation
		[30]	None	Major management challenges of IoT are not addressed
		[36]	Proposed architecture addresses mobility and energy managements	Security and fault tolerance are not considered
		[39]	Black SDN provides confidentiality, integrity, authentication and privacy	Detail of the mechanism is not presented
	Message Oriented Publish Subscribe Middleware based Framework	[23]	Proposed architecture addresses the challenges of managing IoT	Costs and layer numbers are increased
		[24]	Proposed architecture focuses on the scalability and delay issues	Increasing overhead due to computing based trees
		[25]	Takes advantage of crowd sensing for flexible and energy efficient acquisition of sensor data taking into consideration the real time requirements	Major component of the proposed middleware may act as central point of failure. There is also a need to balance load at the cloud broker
	SDN architecture and OpenFlow adaption based management Framework	[42]	OpenFlow adaption for IoT	Does not have evaluation of adapting proposed method
		[43]	Proposed framework enables communications across heterogeneous IoT devices and protocols	There are too many layers that cause to increasing processing delays
Non-SDN API based Programmable Management Framework		[44]	None	Proposed architecture need to details
		[45]	The proposed architecture addresses mobility management	Details of the architecture is not given
		[46]	Programmable architecture for managing IoT	Proposed architecture does not have focus to security and fault tolerance
		[47]	Proposed a generic management framework to support heterogeneous applications in smart environment	Things are attached to virtual control unit and have interconnection as heterogeneous but the details of it is not presented

Figure 29: Pros and cons of SDIoT, non-SDN-based frameworks in literature.

are fault tolerance, energy management, load balancing, and security issues.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] H. Huang, J. Zhu, and L. Zhang, "An SDN based management framework for IoT devices," in *Proceedings of the 25th IET Irish Signals & Systems Conference 2014 and 2014 China-Ireland International Conference on Information and Communities Technologies (ISSC 2014/CIICT 2014)*, pp. 175–179, 2014.

[2] O. Flauzac, C. Gonzalez, A. Hachani, and F. Nolot, "SDN based architecture for IoT and improvement of the security," in *Proceedings of the 29th IEEE International Conference on Advanced Information Networking and Applications Workshops (WAINA '15)*, pp. 688–693, March 2015.

[3] N. Omnes, M. Bouillon, G. Fromentoux, and O. Le Grand, "A programmable and virtualized network & IT infrastructure for the internet of things: How can NFV & SDN help for facing the upcoming challenges," in *Proceedings of the 2015 18th International Conference on Intelligence in Next Generation Networks, ICIN 2015*, pp. 64–69, February 2015.

[4] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, "A software defined networking architecture for the internet-of-things," in *Proceedings of IEEE/IFIP NOMS 2014*

- *IEEE/IFIP Network Operations and Management Symposium: Management in a Software Defined World*, pp. 1–9, May 2014.

[5] P. Hu, "A system architecture for the software-defined industrial internet of things," in *Proceedings of the 2015 IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB)*, pp. 1–19, 2015.

[6] V. Rao, "Software defined networking: Redefining the future of internet in IoT and cloud era," in *Proceedings of the International Conference on Future Internet of Things and Cloud*, pp. 296–301, 2014.

[7] N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: an intellectual history of programmable networks," *ACM Sigcomm Computer Communication*, vol. 44, no. 2, pp. 87–98, 2014.

[8] L. Mainetti, L. Patrono, and A. Vilei, "Evolution of wireless sensor networks towards the Internet of Things: a survey," in *Proceedings of the 19th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pp. 2–7, 2011.

[9] C. Alcaraz, P. Najera, J. Lopez, and R. Roman, "Wireless sensor networks and the internet of things: Do We Need a Complete Integration?" in *1st International Workshop on the Security of the Internet of Things*, pp. 1–8, 2010.

[10] D. Christin, A. Reinhardt, P. Mogre, and R. Steinmetz, "Wireless Sensor Networks and the Internet of Things: Selected Challenges," *Proceedings of the 8th GI/ITG*, vol. 146, no. 7, pp. 31–33, 2016.

[11] S. Khera, N. Mehla, and N. Kaur, "Applications and challenges in wireless sensor networks," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 5, no. 6, pp. 448–451, 2016.

[12] K. Tejasvit, "Challenges in integrating wireless sensor networks into the internet," *International Journal of Engineering and Management Sciences*, vol. 5, no. 1, pp. 7–11, 2014.

[13] H. Lamaazi, N. Benamar, A. J. Jara, L. Ladid, and D. El Ouadghiri, "Challenges of the internet of things: IPv6 and network management," in *Proceedings of the Eighth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, pp. 328–333, 2015.

[14] A. P. Athreya and P. Tague, "Network self-organization in the Internet of Things," in *Proceedings of the 2013 10th Annual IEEE Communications Society Conference on Sensing and Communication in Wireless Networks (SECON)*, pp. 25–33, New Orleans, LA, USA, June 2013.

[15] M. Abomhara, "Security and Privacy in the Internet of Things: Current Status and Open Issues," in *Proceedings of the International Conference on Privacy and Security in Mobile Systems (PRISMS)*, pp. 1–8, 2014.

[16] H. Suo, J. Wan, C. Zou, and J. Liu, "Security in the internet of things: a review," in *Proceedings of the International Conference on Computer Science and Electronics Engineering (ICCSEE '12)*, pp. 648–651, Hangzhou, China, March 2012.

[17] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: the road ahead," *Computer Networks*, vol. 76, pp. 146–164, 2014.

[18] L. Paradis and Q. Han, "A survey of fault management in wireless sensor networks," *Journal of Network and Systems Management*, vol. 15, no. 2, pp. 171–190, 2007.

[19] J. A. Khan, H. K. Qureshi, and A. Iqbal, "Energy management in wireless sensor networks: a survey," *Computers and Electrical Engineering*, vol. 41, no. 1, pp. 159–176, 2015.

[20] D. Wajgi and N. Thakur, "Load balancing algorithms in wireless sensor network: a survey," *IRACST International Journal of Computer Networks and Wireless Communications*, vol. 2, pp. 2250–3501, 2012.

[21] M. Ammar, "Internet of Things: a survey on the security of IoT frameworks," *Journal of Information Security and Applications*, vol. 38, pp. 8–27, 2018.

[22] J. Li, E. Altman, and C. Touati, "A General SDN based IoT Framework with NVF Implementation," *ZTE Communications*, vol. 13, no. 3, pp. 42–45, 2015.

[23] A. Hakiri, P. Berthou, A. Gokhale, and S. Abdellatif, "Publish/subscribe-enabled software defined networking for efficient and scalable IoT communications," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 48–54, 2016.

[24] Y. Wang, Y. Zhang, and J. Chen, "SDNPS: A load-balanced topic-based publish/subscribe system in software-defined networking," *Applied Sciences*, vol. 6, no. 4, pp. 1–21, 2016.

[25] A. Antonić, M. Marjanović, K. Pripužić, and I. P. Žarko, "A mobile crowd sensing ecosystem enabled by CUPUS: cloud-based publish/subscribe middleware for the Internet of Things," *Future Generation Computer Systems*, vol. 56, pp. 607–622, 2017.

[26] Y. Jararweh, M. Al-Ayyoub, A. Darabseh, E. Benkhelifa, M. Vouk, and A. Rindos, "SDIoT: a software defined based internet of things framework," *Journal of Ambient Intelligence and Humanized Computing*, vol. 6, no. 4, pp. 453–461, 2015.

[27] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for WIreless SEnsor networks," in *Proceedings of the 34th IEEE Annual Conference on Computer Communications and Networks, IEEE INFOCOM 2015*, pp. 513–521, May 2015.

[28] Jacquenet C. and M. Boucadair, "A software defined approach to IoT networking," *ZTE Communications*, vol. 1, pp. 1–12, 2016.

[29] D. Wu, D. I. Arkhipov, E. Asmare, Z. Qin, and J. A. McCann, "UbiFlow: Mobility management in urban-scale software defined IoT," in *Proceedings of the 34th IEEE Annual Conference on Computer Communications and Networks, IEEE INFOCOM 2015*, pp. 208–216, May 2015.

[30] K. Sood, S. Yu, and Y. Xiang, "Software-defined wireless networking opportunities and challenges for internet-of-things: a review," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 453–463, 2015.

[31] C. Orfanidis, "Ph.D. Forum Abstract: Increasing Robustness in WSN Using Software Defined Network Architecture," in *Proceedings of the 15th ACM/IEEE International Conference on Information Processing in Sensor Networks, IPSN 2016*, pp. 1-2, April 2016.

[32] A. Hakiri and A. Gokhale, "Rethinking the design of LR-WPAN IoT systems with software-defined networking," in *Proceedings of the 12th Annual International Conference on Distributed Computing in Sensor Systems, DCOSS 2016*, pp. 238–243, May 2016.

[33] M. Jacobsson and C. Orfanidis, "Using Software-defined Networking Principles for Wireless Sensor Networks," in *Proceedings of the 11th Swedish National Computer Networking Workshop (SNCNW)*, pp. 1–4.

[34] J. Wan, S. Tang, Z. Shu et al., "Software-Defined Industrial Internet of Things in the Context of Industry 4.0," *IEEE Sensors Journal*, vol. 16, no. 20, pp. 7373–7380, 2016.

[35] Y. Li, X. Su, J. Riekki, T. Kanter, and R. Rahmani, "A SDN based architecture for horizontal internet of things services," in *Proceedings of the 2016 IEEE International Conference on Communications, ICC 2016*, pp. 1–7, May 2016.

[36] J. Zhou, H. Jiang, J. Wu, L. Wu, C. Zhu, and W. Li, "SDN-Based Application Framework for Wireless Sensor and Actor Networks," *IEEE Access*, vol. 4, pp. 1583–1594, 2016.

[37] S. Chakrabarty, D. W. Engels, and S. Thathapudi, "Black SDN for the internet of things," in *Proceedings of the 12th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2015*, pp. 190–198, October 2015.

[38] F. Khan and S. Hameed, "Software defined security service provisioning framework for internet of things," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 12, pp. 22-23, 2016.

[39] L. M. Arbiza, L. M. Bertholdo, C. R. dos Santos, L. Z. Granville, and L. M. Tarouco, "Refactoring Internet of Things middleware through Software-Defined Network Categories and Subject Descriptors," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pp. 640–645, Salamanca, Spain, April 2015.

[40] A. Darabseh, M. Al-Ayyoub, Y. Jararweh, E. Benkhelifa, M. Vouk, and A. Rindos, "SDStorage: A software defined storage experimental framework," in *Proceedings of the 2015 IEEE International Conference on Cloud Engineering, IC2E 2015*, pp. 341–346, March 2015.

[41] L. Margarida, R. Tarouco, L. Bertholdo, L. Zambenedetti, L. Mendes et al., "Internet of Things in healthcare: Interoperatibility and security issues," in *Proceedings of the 2012 IEEE International Conference on Communications, ICC 2012*, pp. 6121–6125, June 2012.

[42] T. Luo, H. Pink, and T. Quek, "Sensor openflow: enabling software-defined wireless sensor networks," *IEEE Communications Letters*, vol. 16, no. 11, pp. 1896–1899, 2012.

[43] T. Kanter, R. Rahmani, and A. Mahmud, "Conceptual framework for internet of things virtualization via openflow in context-aware networks," *International Journal of Computer Science Issues (IJCSI)*, vol. 10, no. 6, pp. 16–27, 2013.

[44] D. Zeng, T. Miyazaki, S. Guo, T. Tsukahara, J. Kitamichi, and T. Hayashi, "Evolution of software-defined sensor networks," in *Proceedings of the 9th IEEE International Conference on Mobile Ad-Hoc and Sensor Networks, MSN 2013*, pp. 410–413, December 2013.

[45] J. Liu, Y. Li, M. Chen, W. Dong, and D. Jin, "Software-defined internet of things for smart urban sensing," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 55–63, 2015.

[46] S. Nastic, S. Sehic, D.-H. Le, H.-L. Truong, and S. Dustdar, "Provisioning software-defined IoT cloud systems," in *Proceedings of the 2nd International Conference on Future Internet of Things and Cloud (FiCloud '14)*, pp. 288–295, August 2014.

[47] M. Varun, "Software defined unified device management for smart environments," *International Journal of Computer Applications*, vol. 121, no. 9, pp. 30–34, 2015.