Cluster Computing (2022) 25:1619–1636
https://doi.org/10.1007/s10586-021-03402-4

Toward integrating software defined networks with the Internet of Things: a review

Mohammed Al Ja'afreh[1] · Hikmat Adhami[1] · Alaa Eddin Alchalabi[1] · Mohamed Hoda[2] · Abdulmotaleb El Saddik[1]

Received: 14 May 2021 / Revised: 13 August 2021 / Accepted: 26 August 2021 / Published online: 7 September 2021

Abstract

Due to the outbreak of Covid-19 pandemic, activities in most sectors- be it business, education or even healthcare- are taking place in an online rather than in an inline style, and as a result, Internet traffic has increased drastically. Recent studies have highlighted that internet traffic has grown by 70% to 300% since March 2020. According to a recent CNN news article (https://www.cnn.com/2020/03/19/tech/netflix-internet-overload-eu/index.html), popular content providers such as Netflix and YouTube are slowing down in North-America and Europe to keep the internet from breaking. With that being addressed, the existing network deployment and solutions, even with the fifth generation mobile communication (5G) partial deployment, are currently under a huge burden. This work intends to review the integration of two of the most innovative network research areas, Software-defined Networks (SDN) and the Internet of Things (IoT). The IoT aims to interface questions over the Internet while the SDN offers orchestration for network management by decoupling the control plane and the data plane. In this article, we present the state of the art of Software-defined networking and the Internet of Things discussing the integrated architectures, challenges, and designs. Also, we discuss two proposals targeting the QoS Key Performance Indicators (KPIs) in IoT via SDN mobile edge computing along with a few directions of possible research that could fill in gaps in these domains.

Keywords Software defined networks (SDN) · Internet of Things (IoT) · 5G · Tactile Internet (TI) · SWAY · OpenFlow · Quality of Service (QoS) · Ontology · Key Performance Indicators (KPIs)

1 Introduction

With the advancements in communication technologies since the 1990s, the Internet has become the largest international computer network worldwide. However, the new protocols and standards that have been added to refine the performance of the Internet have caused network ovation and its reformatting. For instance, traditional networking is established inbehaviors to grow not only extraordinarily complicated but also challenging to anticipate. At the same time, the existing Internet infrastructure seriously curbs the network inn fixed-work networks i.e., a couple of switches and routers. These devices each have certain capacities that work well together and bolster the network. On the off chance that the network's capacities are actualized as hardware constructs, in case of any change in the network conditions, even with traditional QoS mechanisms such as Integrated Services (IntServ) and Differentiated Services

✉ Mohammed Al Ja'afreh
jaafreh@uottawa.ca

Hikmat Adhami
hadha068@uottawa.ca

Alaa Eddin Alchalabi
aalch040@uottawa.ca

Mohamed Hoda
mhoda053@uottawa.ca

Abdulmotaleb El Saddik
elsaddik@uottawa.ca

[1] EECS, University of Ottawa, Ottawa, Canada

[2] College of the North Atlantic, Doha, Qatar

(DiffServ), these devices will fail to be reinforced in a way that meets the new conditions. As such, adaptability and flexibility are repetitive obstacle for regular IP networks. Many Application Programming interfaces (APIs) are uncovered for provisioning and most switching hardware and software are proprietary. Hence, traditional networks frequently function admirably with restrictive provisioning programming, yet this product cannot be immediately adjusted as required.

On the other hand, Software-defined Networks (SDN) is characterized by "the decoupling of control and packet-forwarding planes in the network" [1]. It empowers the network to legitimately associate with applications through applying programming interfaces, supporting application execution and security, and creating a uniquely adaptable network design that can be changed as required. Apparently, evolving to be the most regularly utilized method for application deployment, SDN is now utilized by enterprises to send their applications more quickly, in the meantime cutting down deployment and operating expenses. IT heads utilizing SDN can oversee and provision their network services from an incorporated point. A network model that yields programmatic management, control, and network asset optimization, SDN applies open APIs to help keep up network control. This network control is established when SDN decouples the network design and traffic engineering, isolating them from their central network infrastructure. This splitting permits the utilization of OpenFlow and other open protocols. These open protocols can access network switches and routers that regularly utilize exclusive and generally closed firmware by applying globally aware software control at the network's edge.

SDN helps clients virtualize their hardware and attempts to make a computer network by separating the system [2] into the accompanying separate planes: The control plane offers the performance and fault management of NetFlow according to the type of the deployed protocols, hence it is often utilized for managing devices designs that are remotely associated with the software-defined network. The data plane advances traffic to its ideal destination. Before traffic arrives at the data plane, the control plane directs what path streams it will take by utilizing the flow control. This enables network administrators to effectively work with the software-defined network and manage the network.

At the point when it was first deployed by huge enterprises, like Google and Amazon, SDN helped them make adaptable server farms, encourage network resources and new cloud based server development, as well as decrease the workload for IT directors. SDN streamlined the efficiency of the up-scaling procedure for these huge organizations and immediately drew the consideration of other huge organizations that quickly embraced SDN to improve their up scaling effectiveness. The framework of SDN is summarized and depicted in Fig. 1.

Additionally, Internet of Things IoT includes broadening web availability beyond standard devices, for example, work areas, workstations, cell phones, and tablets, to any scope non-web enabled physical devices and regular items. Embedded with technology, these devices can convey and communicate over the web, and they can be remotely observed and controlled. As illustrated in Fig. 2, hard sensors, such as fire sensors, ambient light sensors, humidity, and cameras, are physical hardware-based that can be connected to the user. These include wearables and personal sensory devices, gathering mainly physiological data, and providing valuable perceptions on health and well-being. Hard sensors can also be ambient sensors, supplying data on the user's environment at any given time. Moreover, continuous tracking entails data gathered by soft sensors. Soft sensors collect data mainly from Social cloud media (SNs) such as Facebook, Instagram, YouTube, Twitter, LinkedIn, etc. They are referred to as soft sensors because they are software-based, where information is entered into the platform by humans. Both IoT and SNs provide large amounts of sensory data and varied views on environments and people's state of health and well-being [3]. In the context of communication, The raw data collected from the sensors is sent to an IoT gateway, which is installed in the user's home. The gateway will send, via the promising regular network or ultimately using the ultra-reliable with ultra-low latency 5th generation of mobile communication, the sensory information to the cloud server. The server will do data analysis from the received raw data and send semantic, context-aware, and user-friendly messages to the end-user's mobile app or platform [4].

The aforementioned scenario, depicted in Fig. 2, represents the ultimate and supreme use case of deploying IoT in

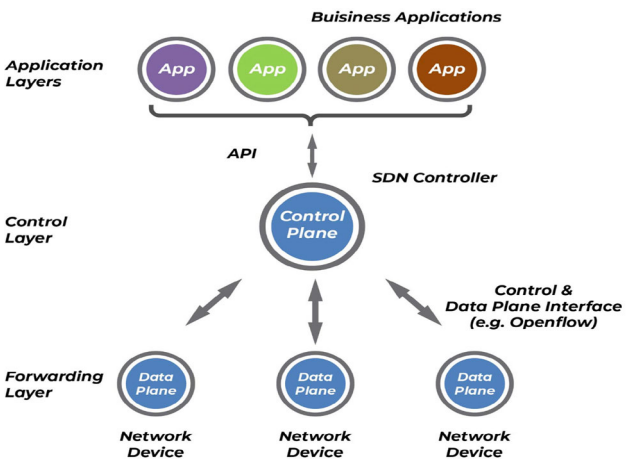

Fig. 1 SDN architecture

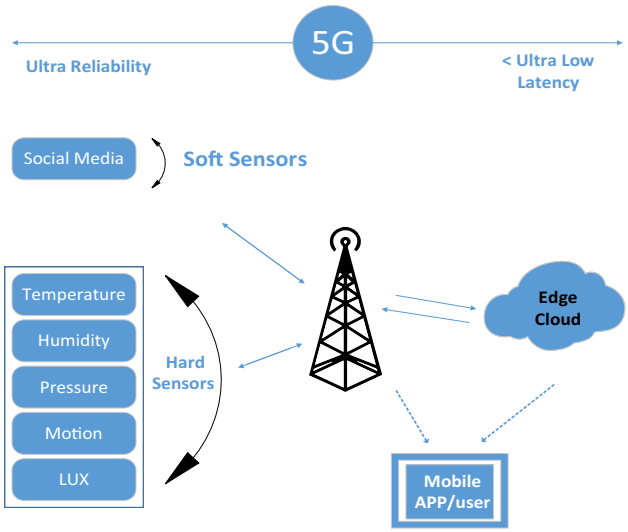

Fig. 2 IoT architecture

every day human's life. Currently and traditionally, the IoT communications comprise nearly all the communication technologies of wireless communications and wired communications. As for wireless communication technology, there are GSM, CDMA, LTE, Wi-Fi, RFID, Bluetooth, and ZigBee. The use of some kind of control architecture is utmost required to ensure a smooth QoS for IoT. Nowadays, associated devices are a part of a situation in which each device converse with other related devices in a domain to robotize home and industrial tasks, and to impart usable sensor data to clients, organizations and other interested parties. IoT devices are intended to be deployed for people at home, in the industry, and the manufacturing domain. After surveying the literture, these hardware-based devices can be classified into three fundamental gatherings: consumer, enterprise, and industrial. User connected devices include smart TVs, speakers, toys, wearables, and well-informed machines. Smart meters, business security frameworks, and smart city technologies. For example, those used to monitor traffic and climate conditions - are instances of industrial and enterprise IoT devices. Different technologies, including air conditioning, thermostats, smart lighting and smart security, enterprise and industrial uses.

In a smart home, for instance, a user arrives home, and his car communicates with the garage to open the entryway. Once inside, the indoor regulator is as of now acclimated to his preferred temperature, and the lighting is set to a lower force and his picked color for relaxation, as his pacemaker information indicates it has been an unpleasant day. In the enterprise, smart sensors situated in a gathering room can enable a worker to find and schedule an accessible room for a meeting, guaranteeing the best possible room type, size, and highlights. When meeting participants go into the room, the temperature will change as indicated by the occupancy, and the lights will diminish as the fitting PowerPoint loads on the screen and the speaker starts his presentation. Consequently, IoT is now an essential and hot research field for both academic and industrial stakeholders. Nevertheless, in the current IoT network, tons of devices are connected and have communication tasks. The access network will face a major strain in the near future.

The main contribution in this article, is to explore the suitability of existing network solutions, architectures and researches in integrating Software Defined Network implemented via edge computing with the IoT architecture. Also, we have tackled the technical challenges to deploy IoT applications with their vital theories and algorithms related to this context. Unlike other papers which present ideas and anticipations of SDN and IoT, we aim at offering the reader advice of the implementation of those infrastructures and progress in solving their challenges. The rest of the article is as the following. Section 2 presents the state of the art of software defined networks (SDN) along with the SDN-Internet of Things architecture. Section 3 argues the necessity for the integration of SDN and IoT from within the applications that have constraints on the KPIs such as End-to-End delay, bandwidth and packet loss. Then, in Sect. 4, we present some future research venues adding our analysis to tackle the challenges in this context. We end up with a concise conclusion in Sect. 5.

2 Software-defined IoT: an overview

With the development of communication technologies, the Internet of Things has revolutionized the network architecture, from industrial to military applications. This revolution includes many aspects such as healthcare, home automation, earthquake warning, traffic control, and industrial processing monitoring. In terms of different use cases, dedicated platforms and applications are built by different providers. As a result, there is a lot of redundancy in IoT devices, data, operations, and system management [5]. Again, generally speaking, an IoT network is composed of a group of sensor and actuator networks, as well as end-users' SNs, which work all operate at the edge network. Meanwhile, the edge network is supported by some gateways and access points (Access Network). Datacenter network and core network also play an important role in IoT network. Besides, SDN technologies can be used in the different types of network in terms of use cases. SDN-based schemes for efficient data collection and network flow monitoring in edge networks have lots of applications in IoT [6] which can be summarized as follows.

– *Data aggregation* In OpenFlow-based flows placement, the SDN controller plays a key role in the network, leveraging the global view of the entire network. At this point, flows can be monitored and analyzed for making improved decisions. As IoT network comprises heterogeneous devices, with SDN, it becomes possible and attainable to control all the devices in a uniform pattern [7].
– *Network monitoring* SDN can provide a global view of the entire network. With this property, the network monitoring in SDN can be achieved in two ways: probing by the controller and reporting from switches when changes are detected in the network. For the probing method, the SDN controller will send probe messages to the switches and routers to get statistics from them periodically. It is proved that optimized operational expenditure (OPEX) is greatly improved in a larger network [8]. Nevertheless, if we rely on switches to send network statistics actively, overhead could drop, while accuracy is compromised. Therefore, we notice that there is a tradeoff between control overhead and accuracy.

2.1 SD-IoT, access network

2.1.1 Access-core integration by simplifying network architecture

The integration of a heterogeneous access network into a single platform will facilitate seamless data exchange among multiple devices. Active remote node (ARN) [9] will play a crucial interface between the end-users and backhaul network. ARN is responsible for short-range communication, mainly the wireless. For long-range passive optical networks (PONs), it works in the backhaul network to provide log-range connectivity. With SDN, service providers will experience conveniences such as dynamic bandwidth allocation, service differentiation, network monitoring, and dynamic spectrum management.

2.1.2 Pub-sub-based architecture

Pub-sub architecture provides more scalability in a dynamic network topology. In this architecture, source nodes will update messages without detailed information of the destination nodes. Besides, the destination nodes (subscribers) also express their interest in receiving a different message without details of source nodes (publishers). This will ensure the number one aspect of expansible networks since it can dramatically reduce the overhead of communication establishment between IoT devices. As shown in [10] a pub-sub SDN architecture can enable scalable and distribution services. In addition, a pub-sub SDN architecture can provision an abstraction layer which reinforces interoperability i.e, independent of specific networking protocol and technology. Therefore, a dedicated application can be dynamically deployed.

2.1.3 SDN based mobile or optical access network

To apply SDN in an optical access network or in a mobile communication infrastructure such as 4G or LTE-A, several aspects need to be taken into consideration. Users can provide feedback to network service providers and the latter will use the information to make adequate decisions to improve the QoS of a network. With this mechanism, the SDN controller will use per-flow analysis to accordingly explore the optimum path for data forwarding, within QoS concerns, consequently improving the overall Quality of experience from the end-user's perspective.

2.2 SD-IoT, core network

2.2.1 Adequate security mechanism at core network

Each enterprise network puts security issues at a high priority. In general, distributed and centralized solutions are two common architecture. The centralized security uses a network intrusion detection system (NIDS) at the core network. The performance is not satisfying in some aspects. For instance, additional dedicated middle-ware is required, which might increase the overhead; besides, a network operator has only limited views of the network since it is a traditional network architecture, which means the host-band solution approaches are OS-specific and this is likely to lead to different contention between solutions [11]. As a result, SDN should be considered as an alternative approach at the core network to achieve different KPIs of a network.

2.2.2 Adequate network traffic distribution

Due to the presence of heterogeneous devices in IoT, routing requests should be handled properly for different application-specific use cases to fulfill the user requirement. Traffic should be distributed through redirecting application-specific requests when they are received within the intermediate nodes while minimizing the associated cost, network load, and delay.

2.3 SD-IoT, data center network

2.3.1 Efficient flow handling

In the data center network there are two types of flows, long-lived and short-lived, which are also known as the elephant- and mice-flows respectively. It is necessary to handle these two flows efficiently without disrupting the network performance. SDN can help with this issue. More details are discussed in [12].

2.3.2 Traffic-aware NFV deployment

Virtual Machine (VM) is the core computing entity in data center networks. With SDN, the controller has a global view of all the traffic, and monitoring is possible for each VM, while these VMs are running IoT analyzing application in the data center. This will ensure scalability in SD-IoT as while meeting the best practice of Network Function virtualization (NFV). Therefore, VMs must be deployed dynamically and efficiently, including placement problems and resource allocation constraints.

2.3.3 Energy-efficient data center networking

Datacenter network accumulates thousands of computing nodes and the energy consumption is a great concern for each DCN deployment scenario. To reduce the consumption issue, properly utilized resource allocation policies should be applied in the deployment mechanism. Consequently, adequate techniques need to be proposed for energy-efficient data center networking.

2.3.4 Over-and-under-subscription of services

Customers are always tends to subscribe more resources in higher priority to meet their real-time requirement. Naturally, the real-time resource is far more expensive. As a result, some data centers are likely over-and underutilized [13] due to the specialized real-time service requests especially in IoT application requests. Leveraging SDN-based dynamic request mapping technique to distribute the requests among data centers, and load balancing are very common to tackle these issues.

2.3.5 Seamless mobility of VMs

Migration of VMs in data centers is always challenging when it is between different DCN vendors. Providing seamless connectivity is a key aspect of IoT. This needs to be achieved in DCN by creating requests by VMs or containers. SDN and NFV are good examples to provide efficient service migration solutions and meet the requirement of QoS and service-level agreement (SLA).

2.4 SD-IoT: a model

In the past decade, IoT emerged as a special skeleton/model of wireless sensor networks (WSNs). blue WSNs suffers from challenges that limit their functionalities. They require a tailored customization for concrete applications which makes them harder to deploy and limits their manageability and flexibility. In order to overcome that, combining the advantages SDNs bring to the game with WSNs creates SD-IoT. In addition, SDN-WSN suffers from security threats that adversely impact the vulnerability and the QoS performance of the system. For instance, Denial of Service Attacks or malicious attacks with false data and parameters would outcome system unreliability [14]. Therefore, the large-scale deployment of WSNs is difficult and poses some problems; therefore, operators need special adaptation procedures when they use specific applications that require flexibility and particular management. To tackle these problems, the proposal of integrate SDN into WSNs, is introduced and a new SD-IoT model is born and surfaced, Fig. 3.

In the aforementioned model and when communication takes place the two layers of SDN (control plane and data plane) use a Sensor-based OpenFlow (SOF) communication protocol. The transmission of data packets is performed by the sensors based on the flow assigned table. The control plane is composed of one or several network controller(s). Intelligence in the network can be accomplished, and network control can be realized (like routing and QoS control). In this design, users can run primary flow table by SOF to attain the programmability of the WSN. SD-IoT has the following features: (1) multifunctionality: as it can handle multiple plug and play applications so that the sensor no longer depends on the operating system of the application. Keep in mind that the control plane holds the logic functions of the network. (2) Flexibility: changing the entire approach in the network setting is a simple task for SD-IoT. This helps network operators and network equipment manufacturers, to escape from incompatible local strategies. (3) Manageability: the network management systems only requires open API provisioned by the control plane. This is likewise the means of adding new applications, without necessarily tweaking the actual primary code. Adaptability in the network can be accomplished using artificial intelligence based network controller thus smart routing and QoS control might be achieved [15]. As illustrated in Fig. 3, the deployed SD-IoT model is composed of three layers: the physical layer, the control layer, and the application layer.

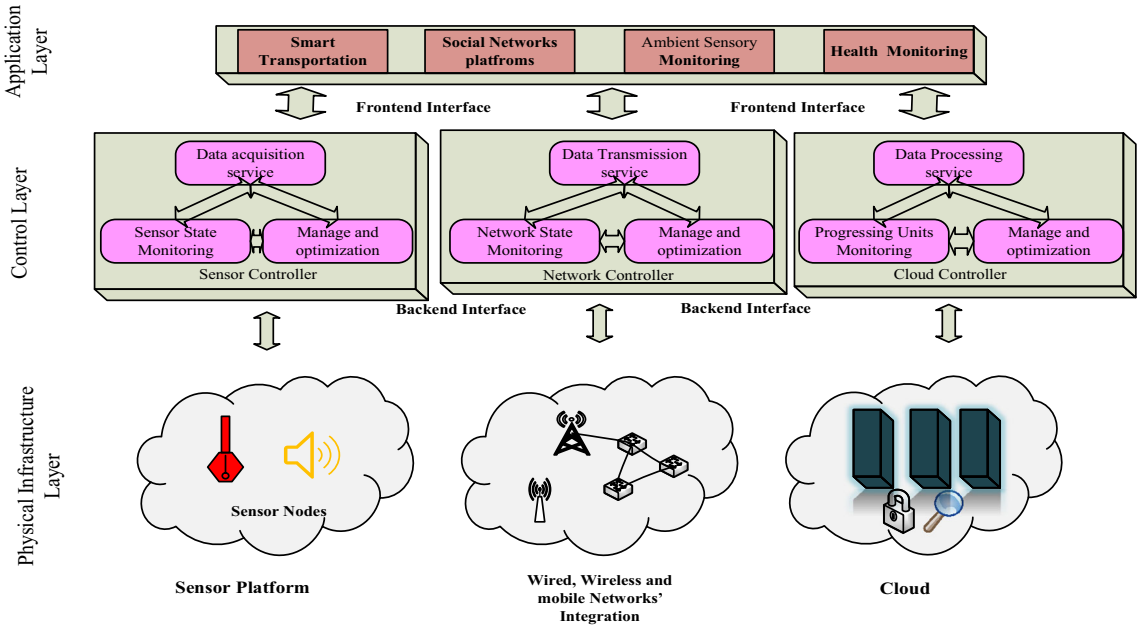

Fig. 3 Structural model of software defined Internet of Things (SD-IoT)

2.4.1 The physical layer

In this layer, all the assets and hardware devices are placed. These devices will be responsible for the main data collection and communication tasks only without the participation of the controlling process. More specifically, it is composed of a sensor network cluster (SN), database pool cluster (DB). For sensor network clusters, IoT devices (sensors contained) are the main function points. The sensors are responsible for recording the data from the surrounded environments in order to use it in different applications. Agents work as assistants or interfaces to communicate with upper layer devices and controllers. The board sensors will combine the accumulated data from bottom sensors and send them by a bridge or gateway to the SDN controller. On the other hand, database pool cluster provides different types of data storage. As a result, a dedicated database will be created to store each IoT device information for further processing and analysis.

2.4.2 The control layer

Similar to SDN architecture, SD-IoT has an SDN controller in the control layer. Besides the basic SDN components, it also contains the specific IoT controller, which focuses on IoT applications and its data forwarding and processing issues. SD-Storage controller, SD-Security controllers are encapsulated in this layer in order to deal with urgent situations. Those controllers form an integrated middle-ware layer; eastern and western bound interfaces are also integrated to ensure the data communication between controllers. In a wide range system where the system is physically distributed, an edge controller should be considered in this middleware layer to handle all the requests from the bottom and the top.

2.4.3 The application layer

SD-IoT provides a DaaS (Data as a service) application layer. User-applications are generated in this layer to facilitate accessing and acting with the stored and analyzed data. Similar to SDN, northbound or front-end APIs should be introduced to enable data communication between the upper layers and controllers.

2.5 The overall workflow of the SD-IoT model

Figure 4 presents the overall workflow SD-IoT model in the control layer.

A service request is created by an IoT device, or more specifically a sensor. Pieces of information are generated and are forwarded to the control layer by gateways and bridges. Before the requests are accepted by the SDN controller or the IoT controller they need to pass the authentication investigation handled by Software Defined Security (SDSec). If this process fails, the requests will be discarded. Otherwise, it will be allowed to enter the message queue to wait for further data processing. After the message processing operations are successfully executed, the IoT controller will add labels and tags to the request packets to a dedicated receiver. The labeled request will be handled by the SDN controller for further forwarding. A

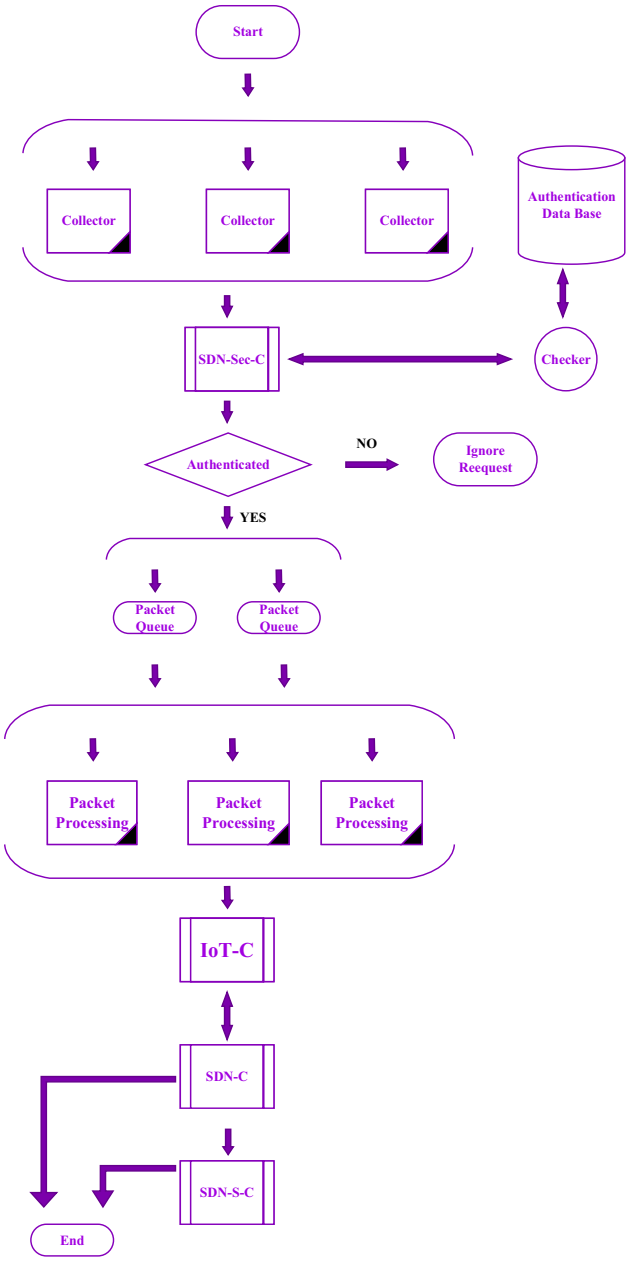

Fig. 4 The workflow of (SD-IoT) to handle service requests from the application layer

specific flow forwarding algorithm will be applied in the SDN controller. Meanwhile, the SDN controller should coordinate with IoT controller in case of updated policies or information changes and challenges. Besides, it is responsible for releasing the flow rule to broadcast the final routing and forwarding information to the bottom devices that are in the sensor network cluster. The SDN controller will then inform the SD-storage controller when new flow rules are created. These new flow will be stored in the DB cluster for additional processing if needed. The workflow briefly introduced how to combine IoT to SDN network. As mentioned earlier, there are plenty of challenges in different inter-networking scenarios in terms of use cases and format. Hence, this what we will explore in the following sections

3 SDN based IoT models for delay/bandwidth sensitive and packet loss-sensitive applications

The vision of the Internet of Things (IoT) is to have smart devices connected over the internet as a backbone infrastructure. More specfically, the network from smart devices to the IoT platform is made up of many communication domains and different technologies are used in inter/intra domain. For example, in the access part (access domain) of the network, a smart device can connect to the IoT gateway using different access technologies like Bluetooth, Zigbee, Wifi, etc. Similarly, the IoT traffic on its path to the IoT platform may go through the optical network, metro network, and even mobile networks like (4G or 5G). Moreover, different IoT applications have different QoS requirements to be met from the network. Software-defined network (SDN) has some tools in its arsenal to cope with the heterogeneity of IoT backbone network and QoS requirements. SDN can utilize its logically centralized control plane to achieve unified control over heterogeneous networks and likewise, the flexibility/programmability feature can be used to monitor the network and make dynamic run time changes in the network to satisfy the QoS requirements of different IoT applications. There are many QoS metrics like throughput, bandwidth, jitter, delay, reliability, etc but in [16], Jin et al. categorized IoT applications into two broad categories:

– *Delay sensitive IoT applications* This class of applications requires packets to be received by the IoT platform with strict packet delay constraints. They can still work properly if there is some packet loss as long as the non-dropped packets are received within a delay threshold. Examples of these kinds of applications include video surveillance like (Closed-circuit television) CCTV infrastructure, smart connected car systems for autonomous driving, and real-time health monitoring systems, etc.
– *Packet loss-sensitive IoT applications* Unlike delay-sensitive applications, packet loss-sensitive applications require packets to be received with no packet loss and can withstand delay. Most of the IoT applications use UDP as a transport protocol which does not provide end to end guarantee, so reliability is implemented at the higher layers, but due to the resource-constrained nature of the IoT devices, it is the responsibility of

the network to provide no packet loss in order to save energy, processing power, and memory usage on IoT devices. For the packet to be delivered reliably, it must be stored in the buffer of the IoT device so that in case of packet loss, it can be retransmitted. It is worth noting that some IoT devices have limited buffer space to store packets; in this case the SDN has to communicate with IoT gateway to enforce some queuing mechanism at the expense of latency. Examples of these applications include wireless sensor network for monitoring different environmental parameters and wireless body area network (WBAN) [17].

3.1 SDN based IoT aware edge/cloud architecture for a bandwidth-constrained application

Analyzing raw data collected by IoT devices is a need for any IoT infrastructure to provide useful and concise information for the application's user. In some applications, sensors/actuators collect data in huge amount and require a solid network to carry this data to the data centers located in the core of the network for analytics. In [18], the authors mention that the data collected by a connected car system and sent to the cloud for analysis is approximately 25 Gb/hour and will exceed this data rate if more sensors are added to the self-driven cars. A connected car has many sensors that may collect the surrounding information, the driver's behavior pattern, and telematics to provide vehicle efficiency and passenger safety. Transferring this huge amount of data from edge networks through metro and core networks to data centers for analytics will cause congestion in the network, and hence will hinder the performance of delay-sensitive applications. Besides, transferring such data requires huge bandwidth which is very expensive to obtain. Therefore, there is alarming demands to find an alternative architecture that does not require data sent to the core data center all the time. Edge and Mobile edge computing is a good candidate approach to tackle this issue.

3.1.1 Edge computing in IoT

To solve the processing of large amounts of data at a core data center, the idea of edge computing was introduced. The basic goal of edge computing is to bring computation and storage resources closer to the user i.e., the edge of the network, which will provide real-time data processing, real-time message response and temporary data storage. The computing and storage resource at the edge network is called as a micro and small data center. First analytics is done at the edge side to extract useful information, and then this data is sent to the core data center for storage purposes to be used by IoT applications. This requires a dynamic distribution of IoT analytics between the core and edge data centers in order to reduce the dependency on the network, hence minimizing the impact of bandwidth and latency constraints of the core network. Authors in [19] survey edge computing in the context of IoT. Dynamic distribution of analytics requires programmability/flexibility, which is a feature of SDN. Moreover, this implies tight coordinations between IoT platform, SDN infrastructure, and cloud/edge infrastructure. The authors in [20, 21] extend the architecture proposed in [19] by providing an SDN-based edge data center (micro and small data center) that uses virtualization of computing and storage i.e, utilizing NFV for better resource utilization. Further, they demonstrate the proposed architecture for distributed analytics using CCTV applications and congestion avoidance using a dynamic distribution of analytics. Key features of this architecture are:

1. flexible transition between cloud and edge resources based on SDN based (Open Flow) real-time monitoring of the network resources.
2. Transferring the request control of edge resource from IoT gateway to a logically central IoT analytics platform.
3. Hierarchical implementation of the control plane that integrates the control plane of both different networks and cloud/edge computer.
4. Virtual machine-based cloud computes and container-based edge computing

Figure 5 explores the connectivity diagram of different components of the unified control and communication SD-IoT edge based network framework. The whole architecture can be divided into three layers:

– *Infrastructure layer* Data is collected from the smart things/sensors and aggregated by IoT gateway in case of static devices and via base stations if devices are mobile (such as smart vehicles or drones and haptic-robotic). The aggregated data is then forwarded to the edge network which includes the SDN edge switches/routers for data forwarding and micro/small DC for edge compute and storage resource. Similarly, The SDN packet network and Fiber-Wireless (FiWi) network representing a metro network and optical fiber wireless network respectively. They collectively deliver connectivity between edge networks to core data-centers. IoT flow monitors are placed on the transit links between edge network and packet network and also between packet network and broadband channel for the sake of adaptively observing the average bandwidth utilization.
– *Control layer* includes multiple SDN controllers divided into a hierarchical pattern. At the lower layer

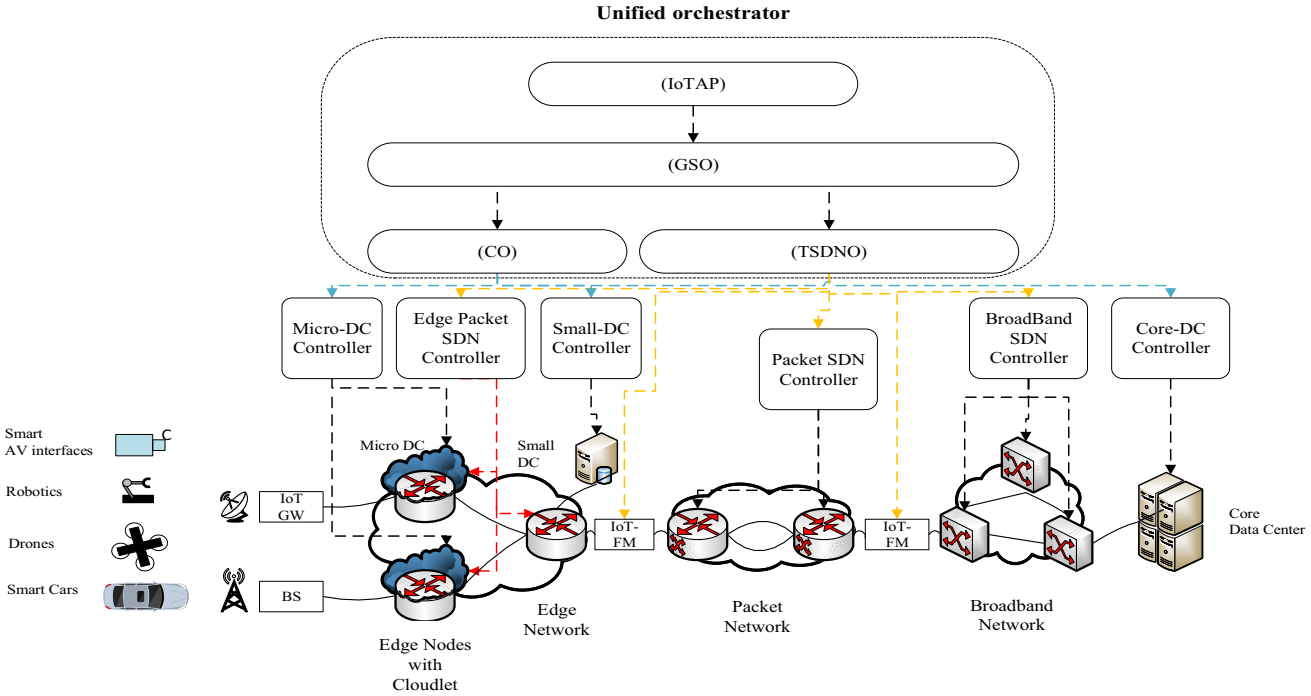

Fig. 5 Unified communication architecture of IoT, SDN and cloud computing

of hierarchy, there are a Micro DC controller, Small DC controller, and Core DC controller to provide control plane functionality for computing and storage resources at micro DC, small DC, and core DC respectively. Similarly, the multiple SDN controllers (Edge Packet SDN controller, Packet SDN controller, and broadband SDN controller) provide control functionality of packet forwarding for the edge, packet, and broadband network. At the second layer of hierarchy, a cloud orchestrator (*CO*) which provides a higher-level abstraction for data centers (micro, small, and core) and controls micro, small, and core DC, controllers. In addition, The IoT aware transport SDN orchestrator (*TSDNO*) is connected to each SDN network controller at the lower level and provides a unified transport network operating system. *TSDNO* uses a transport API [22] to interact with SDN controllers of heterogeneous transport networks to provide end-to-end connectivity services by using higher-level abstracted controls from lower-level SDN controllers. Another function of TSDNO is to provision tags for IoT flows and subscribe to their real-time periodic information from *IoTFM* (infrastructure level). Once *TSDNO* detects a link congestion on the transit connections, it informs the controller at higher hierarchy about all the IoT flows passing through the congested link. Consequently, *TSDNO* can be viewed as a controller of controllers. At the third layer of hierarchy, an IoT aware global orchestrator (*GO*) provides global automated configuration, management, and coordination of end-to-end services. It receives information regarding bandwidth and latency threshold breaches from *TSDNO* and forwards the request to the IoT analytics platform (*IoTAP*) to provision distributed analytic resources at edge, small, or core DC. Upon receiving the global service request from *IoTAP*, the GO decomposes it into two parts. The first portion, referred to as cloud resource request, is forwarded to the *CO* to instantiate the VM at core DC or containers at the edge/small DC. The second part i.e, end-to-end network and VM connectivity requests are forwarded to TSDNO which further relay this information to lower-level SDN controllers to provision flows. IoTAP sits in the top layer and communicates with IoT applications in the application layer. It prioritizes the traffic by deciding which IoT application should be distributed first.

– *Application layer* which sits on top of the analytic platform. It provides services to application end-users, and higher-level instructions to the control layer.

3.2 Experimental case study

Figure 6 shows a proof of concept for the distributed analytics of the unified architecture described in the previous section. For that purpose, a CCTV IoT based

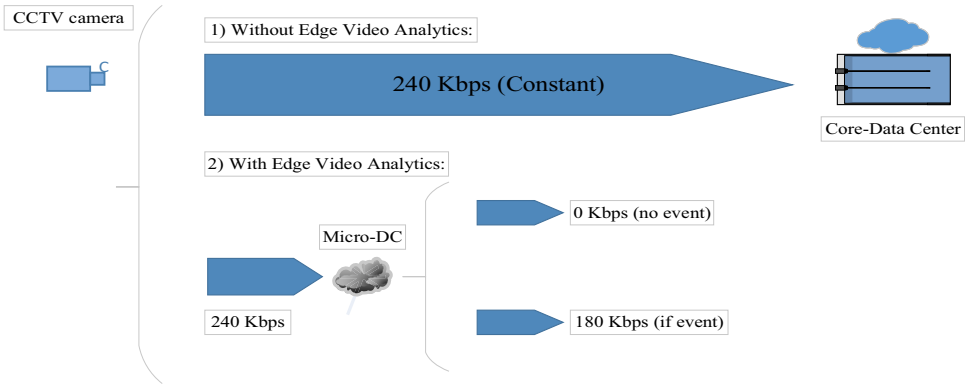

Fig. 6 Video streaming analytic use case

application was deployed using two scenarios i.e., with and without incorporating edge computing to the SD-IoT system. Normal analytics of the video stream will be done at the edge datacenter aka (micro DC) and then in case of any mishap, the video stream will be forwarded to core DC for storage purposes. As depicted in Fig. 6, without edge computing, the raw stream requires a 240 kbps rate from the edge network to the core so that the edge computing can save the bandwidth from 25% up to 100% in the core network as long as no events more specifically motion detection are encountered.

In order to provide this kind of service, *IoTAP* asks the *GO* to provision a service that includes a VM on the core DC which in turn includes a virtual container at micro DC as well as a path between the CCTV camera to edge container (flow1). In addition, the *GO* will provision a path from edge container to VM (flow2) in core DC. The *GO* splits the request into two: one for the *CO* and another for *TSDNO*. The *TSDNO* part consists of end-to-end connectivity requests while the *CO* part consists of instantiation of VM and container. *TSDNO* selects the lower level SDN controllers whose networks are involved in the path asking them to add tag flows on each switches at the provisioned path. Similarly, the *CO* selects the corresponding DC controllers and asks them to spin up VM and containers. In normal cases, the analytics will run on a container at the edge node and as soon as the analytics detect activity, the video stream will be forwarded using flow2 to VM in core DC for storage purposes as depicted in Fig. 7a. In order to provide proof of concept for dynamic switching of core analytics to edge analytics after detecting congestion, the same setup as described above was used. To get the bandwidth usage, the *IoTFM* functionality is integrated on the OpenFlow switches and *TSDNO* can periodically get packet rate on a specific port using OpenFlow messages. Initially, *IoTAP* deploys three services, each with its corresponding flow and VM in core DC more specifically ($S1$ uses $VM1$ and flow1, $S2$ uses $VM2$ and flow2 and $S3$ uses $VM3$ and flow3). The traffic rate threshold used for the transit link is 20 Kbps and the time threshold is 10s. Next, dummy packets are generated for each of the three applications using a packet generator and are passed to the edge network. $S1$, $S2$, and $S3$ are operating at 20Kbps, 10Kpbs, and 15Kbps respectively as shown in Fig. 7b. As we can see the link between the metro network and optical network carries 45Kbps in total, which exceeds the threshold for 40Kpbs, so congestion will be detected by *TSDNO* (by monitoring the output of $IoTFM3$). *TSDNO* will inform *GO* about this congestion and flows passing through the link. In response, the *GO* will forward the same information to *IoTAP*. Based on the information for *GO*, the *IoTAP* platform decides to provision edge analytics for $S1$ by creating a new service $S4$ and discarding $S1$. This information is forwarded to *GO* which again splits the requests into two parts one for *CO* and the other for *TSDNO*. $VM4$, $VM5$, flow4, and flow5 are created for new service instance $S4$ in order to complete the switching process, $VM1$ with flow1 will be discarded. The deletion and instantiation of new VM's are done by *REST API* of the *CO*. For new flow provisioning, *TSDNO* offloads the responsibility to involve lower-level controllers. New flows and VM's are shown in Fig. 7c. Figure 7d shows the timing graph against the individual and aggregated traffic flows. Flow1 traffic starts from 4s and at this point, the aggregated traffic $(IoT - FM3)$ is the traffic of flow1. Flow2 starts at 9s, and the aggregated traffic is still less than the threshold. Flow 3 starts at 19s and the aggregated traffic starts approaching the threshold. At 21s, aggregate traffic exceeds the threshold and then *TSDNO* notifies *GO* at 31s (10s time threshold) about the congested links and flows through it. *IoTAP* removes $S1$ and instantiates $S4$. It is obvious that traffic for flow1 starts decreasing and flow5 starts increasing at 34s and as a result, the aggregated traffic starts dropping. At this point, the analytics for $S4$ will be done at the edge so that the traffic on flow5 will be minimal as compared to flow4. Flow 4 traffic is not shown, as it is not monitored on a transit link.

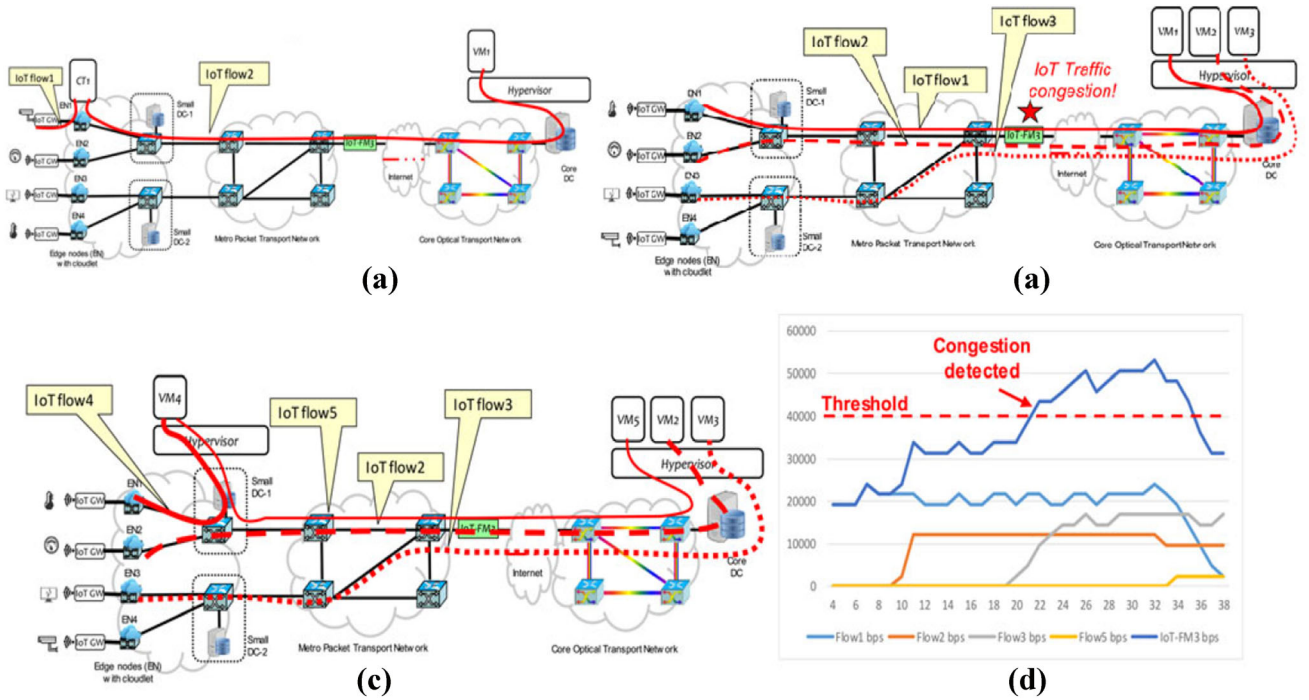

Fig. 7 Deployment and bench-marking video streaming analytic use case adapted from [21]

3.3 Traffic-aware QoS routing in software-defined IoT: SWAY

In the edge-based architecture, explained in the previous section, SDN controllers monitor only bandwidth on links between different networks. The idea of Traffic-Aware QoS Routing in Software-Defined IoT also referred to as SWAY [23–25] is to add link delay and packet loss to the list of QoS metrics for selecting a routing path. All the discussed research works [5, 20, 26–28] are based solely on a particular one QoS metric, but the authors in [24] consider a combination of bandwidth, delay and, packet loss probability. The edge-based solution can benefit IoT applications that require more bandwidth, but some applications do not have large bandwidth requirements and typically UDP as per the transport layer protocol. UDP can help in the delay-sensitive applications whereas low rate UDP can experience packet loss if it passes through the same shared network where normal TCP traffic (greedy traffic) is flowing on the internet as mentioned in [29]. Packet loss in the case of UDP puts more responsibility on the application layer of the IoT device for re-transmission, but due to IoT limited capabilities and resource constraints, it is not feasible to address that issue at the application layer. Hence, the network has to take into account delay as well as packet loss on the link before forwarding the IoT traffic. Additionally, and as mentioned earlier, network flexibility and programmability are needed to provide dynamic QoS requirements of heterogeneous networks and IoT applications.

Consequently, the authors in [24, 25] proposed simple architecture using *SWAY* as such the controller architecture is more QoS oriented.

Their work can be summarized and depicted in Fig. 8. As can be noticed, the central SDN controller manages the traffic flowing through the SDN switches in the network

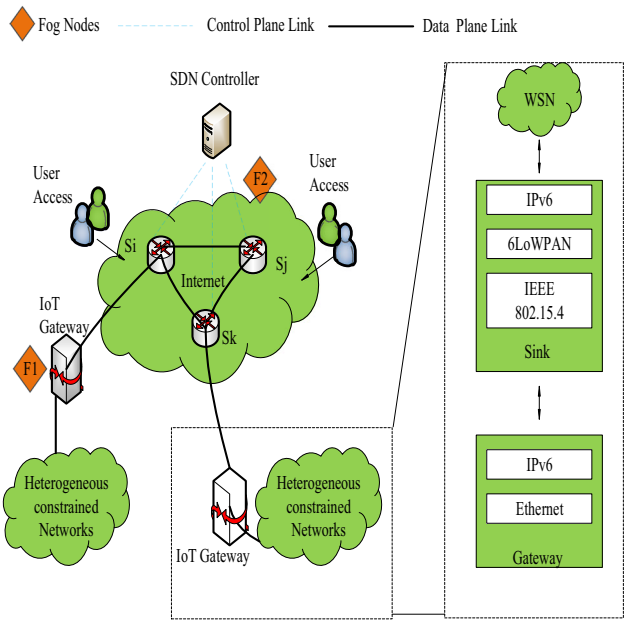

Fig. 8 SDN Controller adapting traffic aware routing (*SWAY*)

while fog nodes F_1 and F_2 provide edge services for bandwidth-hungry applications. The architecture is heterogeneous as it supports all possible wired and wireless data plane links are connected to the IoT heterogeneous gateway. IoT application will be running on top of the SDN controller which interacts using Northbound i.e front-end API while Southbound aka back-end API using OpenFlow protocol for communication with the data planes. The traffic aware routing algorithm runs on the SDN controller and can be described as follows: First, the IoT flow is categorized as delay-sensitive or packet loss sensitive or both using a flow classification module and then represented by four tuples i-e; $(s_k, t_k, \lambda_k, q_k)$, where s_k represents the source, t_k represents the destination, λ_k represents whether this flow is delay (ds) or packet loss (ls) sensitive and q_k is the bandwidth, delay and packet loss constraints provided by the IoT application running on top of SDN controller. Link delay (f_d) and packet loss statistics (f_l) are collected by the statistics collector module using OpenFlow. For link delay, the controller sends a probe message to a switch and tells it to forward it to the next switch, after which this probe packet is forwarded back to the controller by the second switch. Using the time of sending, time of receiving the probe message back, and compensating for the delay in the message forwarding, the controller calculates the link delay between switches. Packet loss is calculated from the difference in the port statistics of the switches. Additionally, customized weights are tweaked for traffic to treat the flow as more delay-sensitive (c_1) or packet loss sensitive (c_2). The topology manager module 9 provides the whole picture of the network. The packet-in module intercepts the OpenFlow messages corresponding to a new flow and sends them to the flow differentiator for classification (Fig. 9).

The heart of [24] is the QoS routing module, whose input is flow (tuples), link delay (f_d), link packet loss (f_l), customized weights (c_1 and c_2). it provides a set of optimized paths based on QoS requirements. In [30, 24], an optimization criterion based on the values of four parameters f_d, f_l, c_1 and c_2. The authors classify the process of finding an optimal path as non-deterministic polynomial-time NP-hard therefore, a heuristic approach is proposed to be implemented in the QoS routing module. In the heuristic approach, the algorithm calculates a set of shortest, loop-free paths using Yen's K-shortest path algorithm [31]. These sets of paths are evaluated against the QoS imposed by the IoT application and the best matching path is selected accordingly.

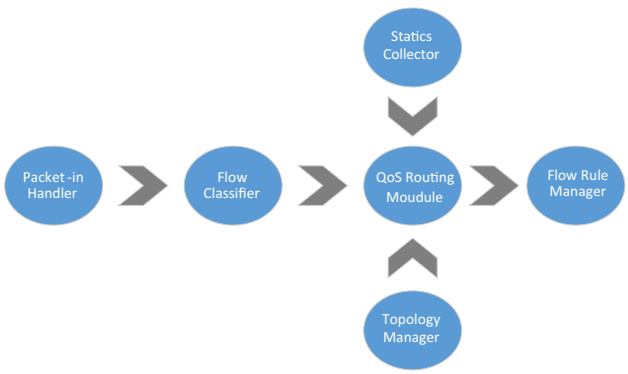

Fig. 9 SWAY SDN controller modules

It is worth noting that SWAY algorithm also considers another factor which is the amount of exiting flow entries on OpenFlow switches while deciding on the path. By doing so, Sway tries to avoid paths with switches with a high number of flow entries. This is an important factor because it may cause packet loss in case all the flows have one common switch e.g (memory-constrained issue) in the path. Other modules of the controller include the flow rule manager, which takes the output of QoS routing modules and then adds flow entries to all the switches in the path using OpenFlow protocol. It also updates the rule-capacity of each switch after adding flow entries.

In [24, 25, 32] authors compares the performance of SWAY algorithm with Shortest Path Delay (*SPD*), Minimum Occupied Rule Capacity (*MRC*). The outcome of those studies shows that the SWAY algorithm outperforms both of these previously proposed architectures. However, it should be noted that SWAY SD-IoT module does not consider another kind of QoS meterics (e.g., jitter-sensitive). Moreover, a low-level packet classification arrangement for fine-grained QoS forwarding is not taken into consideration in the Sway SDN framework.

4 Future research venues and open discussion

Even though recent research work [7, 14, 20, 33, 34] have emphasized the power of envisioning a software-defined architecture for IoT, this domain is still has lots of potential improvements and it has its challenges. In this section, a summary of the most significant challenges is provided. This includes both the challenges that have been partially tackled and the others that continue to open new research venues.

– *Reliability* A crucial condition for the success of any software development is reliability. If the system is subject to any failure, its users should be aware of it and the problem must be resolved immediately. What does software reliability mean? It is the guarantee of a proper functioning in a particular setting for a specified amount of time. In order to prevent any flaws and to increase the network availability, the configuration of the SDN

controller must validate the network management and also must validate and ensure that the minimum reliability measures are met.

The authors in [35], have studied the reliability of Open Network Operating System (ONOS) which can be used as production-grade SDN controller. The study has found a regular pattern of the releases, the number of bugs, fault detection and resolution time. In legacy networks, when devices stop working, continuity and flow control keep working since network traffic is routed through nearby alternative path/nodes. In SDN, the whole network is dependent on the central controller. If the latter fails, the whole network collapses. Consequently, vendors/ developers must harness the main controller functions if they want network reliability to be optimal. In order to overcome this challenge, a very common solution is utilizing the power of virtualization and adopting to the container culture of networks. Tools like Docker and Kubernetes (also known as K8S) [36, 37] would offer a very scalable solution to the reliability of the SDN controller. Having the ability to spawn a virtualized image of a controller in milliseconds is a great option while mitigating the failure of the whole network. Kubernetes also offers orchestration of those virtualized pods such that if a pod fails, the system will respawn a mirror image in milliseconds. Other tools like OpenStack also offer the ability to create overlay networks which could contain the SDN controller and a few replicas of the controller. This would help taking the virtualization to the next level by not only virtualizing the controller of the SDN, but also virtualizing the overlay network to have a better control on the SDN traffic.

– *Scalability* The traditional LAN is organized in multiple layers in which multiple Layer 2 networks are connected. Thanks to Layer 3 routing functionality. Unfortunately, these traditional LANs do not scale very well regarding east-west traffic because at least one Layer 3 device and probably multiple Layer 3 devices are in the end-to-end path. What is scalability? It is when a software, network or organization succeeds in growing and managing ever-increasing demand. An SDN controller should have the capacity to support a minimum of 100 switches and to alleviate the impact of network broadcast overhead as well as the rapid increase of flow table entries. In brief, as far as a system network or process is concerned, scalability means the ability to manage an ever-increasing load of work or the possibility of expanding it in order to be able to successfully deal with that load. In this context, the use of AI becomes very handy. AI algorithms can help the network predict the bottlenecks before they appear by a temporal analysis if the packet flows. When that bottleneck is identified, systems would be able to spin up hierarchical SDN controllers in order to offload some of the tasks on the main controller.
– *Low-level interface* Development is a must when it comes to the control applications of SDN network management and network policy; this means that the switch used should change into low-level configurations. The various asynchronous events observed at the switches must be coordinated by the programming interface of the SDN framework in order to carry out simple tasks.
– *Performance and security* The performance of the SDN may be compromised due to new forms of network attacks resulting from the open interfaces of the SDN network. Different distributed denial-of-service (D-DOS) attacks may stop networks from normal functioning. This in turn, may overflow the queue of the TCP protocol in the server of any organization, resulting in its outage. For that, solutions should be developed in the SDN framework in a way to handle the software integrity, remote access management, network threat detection and mitigation, authentication, and authorization of the users. IoT information security is one of the core technologies related to safe and sustainable advancement of the IoT; thus it has to be seriously considered. Simultaneously, the IoT collects more and more personal data and hence spots higher requirements on privacy protection. In addition to the conventional communication network security issues and due to the fact that the IoT is composed of a huge number of things, and a relative deficiency of intelligent management and human control, the IoT might face main security concerns related to its own specific characteristics. Accordingly, more research work has to be done in this domain to ensure a proper IoT encryption mechanism, satisfactory security design, and a dedicated communication protocol to guarantee the security, reliability, and stability of the data collected in the sensing i.e, application layer. A good example of security applied on IoT devices is the inclusion of virtualized firewalls like virtualized network functions (VNF). Those VNFs can be deployed in any layer of the hierarchy creating a more robust system to any attacks. It would be ideal if virtualized firewalls were applied on a hierarchical manner, for example, on the device level, access point level, router level, then reaching the SDN controller level. This way the probability of threat detection would be much higher which would also offload the detection off a central entity.

Another topic that contributes to the future research of the performance and security in the context of SDN-IoT systems is the Intrusion Detection Systems (IDS)

[38, 39]. There are a multitude of open issues that can be discussed such as, IoT IDS deployment strategies, and synchronization between multiple IoT IDS. The advantage of having distributed IDS is the ability to identify attacks across the IoT ecosystem in a distributed manner. Additionally, in the cases where the traffic may occasionally rise because of an unusual demand, the internet and infrastructure provider will be required to deploy multiple instances of IoT-IDS, and that deployment is an optimization by itself due to the capacity and proximity constraints which is a challenging multivariate problem [40]. Moreover, running copies of the same IDS over multiple instances in a distributed manner is a challenging task. It is known that most of network functions are stateful, which means that they frequently keep a state that is either read or updated such as the packets' statistics. Those distributed IDS over a big IoT ecosystem instances should constantly keep updating the state of the network in order to avoid any failure of operations. IDS instances should have some sort of communication with each other and synchronize the network state. This is achievable by either a synchronization protocol or by having a central server which can consolidates and distributes the same state to all other IDS. Having a central server can assists and advance the task of the intrusion detection systems by packet analysis using AI and then by making a decision and taking actions based on the analysis [38, 39].

– *IoT and Network Simulation* After the computer and the Internet, the IoT is now the new wave of innovation in the world information and industry. The investment in research and development (RD) of a next generation Information and Communications Technology (ICT) applications is growing, including mobile Internet, IoT, cloud computing, big data, software-defined network, 5G. The basic component of the IoT in the network are sensors. However, the evolving number of the terminal equipment, makes it tremendously difficult to largely implement and deploy sensory network in the long term. This is because, in most of the cases, the number usable of sensor nodes is limited. It becomes difficult, in real environment, to verify and assess the relevant theories and algorithms/protocols (Fig. 10).

To cater for the need of the large-scale deployment of sensor networks, we need to simulate IoT procedure and venture in order to scrutinize high-deployment setting demands and hardware costs. Therefore, it is of highly importance to build an IoT simulation platform. The Riverbed®Modeler [41], formerly referred to as Opnet [42], is a good candidate for creating a large scale IoT testbed as it composes a powerful feature, namely the system in the

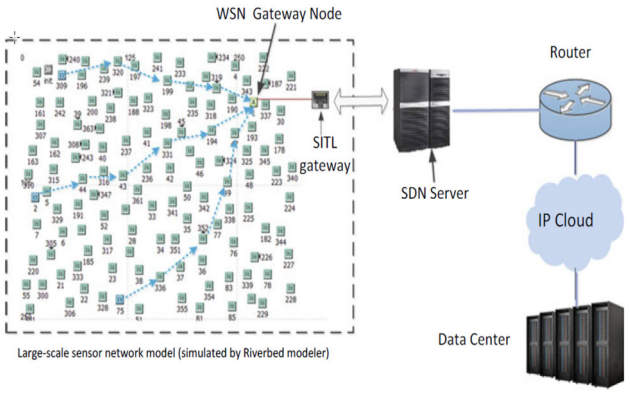

Fig. 10 Riverbed IoT wireless semi-physical model testbed

loop (STIL), which enables the exchange of data between the simulated large scale sensors and a real network as demonstrated in Fig. 11. The riverbed would also help the simulation of any network intelligence and AI which could discover early weaknesses in the AI models. In a nutshell Table 1, recaps the state-of-art on SD-IoT. Comprehensive amalgamation of the current literature discloses that there is a research gap on benefit from the flexibility of SDN to deliver QoS criteria such as either data and loss rates or latency or both according to flow-types of IoT, while concurrently when taking into consideration adds rule-capacity limitations levied by SDN-based architecture.

In our view, by integrating the edge computing architecture (presented in Sect. 3.1) and QoS aware routing algorithm (from Sect. 3.3), we will get a more scalable and robust solution for QoS aware routing and SDN-IoT model, hence enhancing IoT unified communication in general. In the case of QoS-aware routing (*SWAY*), we can see that we already have edge computing (fog nodes), but the whole seamless switching between core and edge DC was not discussed. We argue that in the case of SWAY routing, there is still a need for smooth switching of analytic resources from one edge to other DC edges. Consider we try to do edge analytics on the first edge node, and we find that all resources are completely utilized, so we will have to consider another edge node for the analytics, and this will generate traffic through the network which will require QoS aware routing like *SWAY*. Besides, *SWAY* architecture does not provide a hierarchical SDN controller architecture, which is useful in the proper delegation for responsibility and better management, but the dynamic edge analytics architecture provides this feature so combining both architectures is a win-win situation.

From the architectural point of view, we believe that including QoS aware routing (*SWAY*) including the SDN controllers located in the lower level of hierarchy as well as at *TSDNO* will yield better performance. The lower level of SDN controllers manage a specific domain of the network,

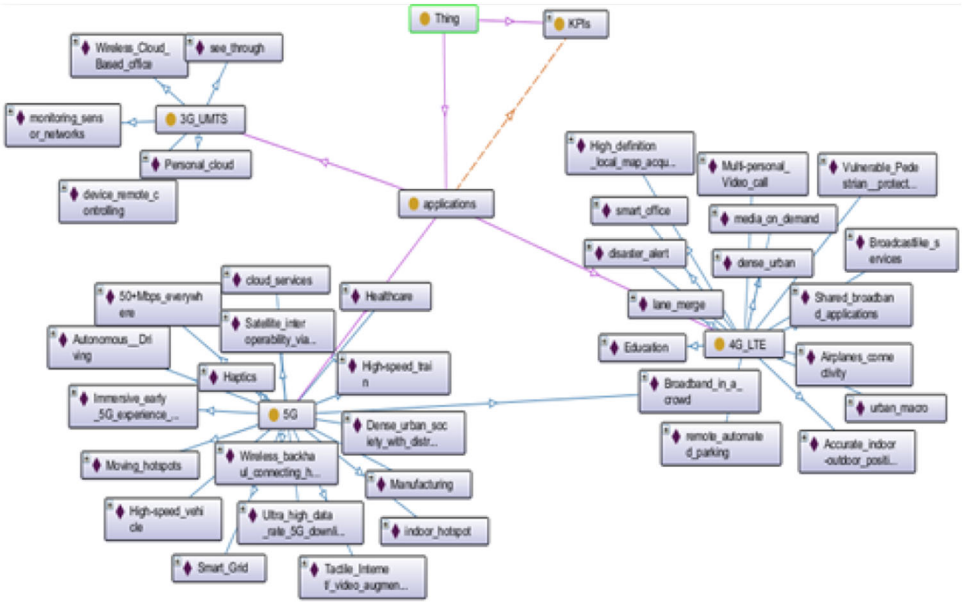

Fig. 11 Ontology which links a 5G application to the specific KPIs

Table 1 Summary of the current state-of-the-art on SD-IoT

Research work	Area of intertest
Authors in [20, 26, 28]	End-to-end orchestration, management, and control of IoT services using SDN
Authors in [43]	Traffic engineering in wireless fogs routers through SDN
Authors in [44]	Federated SDN controllers for IoT and FiWi access network
Authors in [15]	Deep learning-based adaptive channel assignment for SDIoT
Authors in [5, 24, 27]	QoS routing of SDIoT considering only one to two KPIs such as delay, throughput, and loss

so if traffic is to be routed with a specific network domain, then that controller can decide on its own. However, if we need to cross the domain to another segment like a metro/optical, we should implement *TSDNO* in the same level.

From an application point of view: considering the ontology at the SDN controller can automatically relates the future Tactile Internet e.g., IoT applications to their key performance indicators. In our recent work [45], we have addressed, through ontologies, the relationship between 5G/Tactile Internet applications and their main (KPIs). By applying the Ontology to the SDN controller, we can change the classifications of the KPIs and the applications so we can infer the most suitable network type to satisfy the Quality of Service for those applications. This can be very beneficial to the SDN network controller to dynamically optimize the communication channel needed for provisioning the required QoS thresholds.

5 Conclusion

The Covid 19 pandemic has drastically increased the internet traffic . Existing networking algorithms are not capable to effectively implementing QoS control to automatically adjust to repeatedly fluctuating network conditions. Accordingly, new network solutions and algorithms have to be developed to help solve this issue. In this paper, we have reviewed the concept of software-defined networking (SDN) and explained the features of the different specifications of the OpenFlow protocol, which is a means to implement SDN-based networks. A brief introduction of hard and soft IoT sensors and their application is already examined in this article from different technical perspectives. We have also presented the architectural points and current issues and research lacuna of combining SDN with IoT. Lastly, we have discussed the two architectures of SDN with IoT that targets the QoS attributes. Compared to other survey paper papers, we have given a more detailed and synthesis analysis of current SDN and IoT systems due to their crucial role in the functioning of the unified communication system. We believe that deployment of AI-

based SDN controllers could help expand the adoption of IoT because of the flexibility, programmability, central control provided to the heterogeneous network architecture and QoS needs of IoT application/Infrastructure.

Author contributions The main contribution in this article, is to explore the suitability of existing network solutions, architectures and researches in integrating Software Defined Network implemented via edge computing with the IoT architecture. Also, we have tackled the technical challenges to deploy IoT applications with their vital theories and algorithms related to this context. Unlike other papers which present ideas and anticipations of SDN and IoT, we aim at offering the reader advice of the implementation of those infrastructures and progress in solving their challenges.

Funding None.

Data availability The article represents a review, hence no data/dataset are provided.

Declarations

Ethical approval None.

References

1. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: Openflow: enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008)
2. Ordonez-Lucena, J., Ameigeiras, P., Lopez, D., Ramos-Munoz, J.J., Lorca, J., Folgueira, J.: Network slicing for 5g with sdn/nfv: concepts, architectures, and challenges. IEEE Commun. Mag. **55**(5), 80–87 (2017)
3. El Saddik, A.: Digital twins: the convergence of multimedia technologies. IEEE Multimed. **25**(2), 87–92 (2018)
4. Rahman, A., Chakraborty, C., Anwar, A., Karim, M., Islam, M., Kundu, D., Rahman, Z., Band, S.S., et al.: Sdn-iot empowered intelligent framework for industry 4.0 applications during covid-19 pandemic. Clust. Comput. (2021). https://doi.org/10.1007/s10586-021-03367-4
5. Qin, Z., Denker, G., Giannelli, C., Bellavista, P., Venkatasubramanian, N.: A software defined networking architecture for the internet-of-things. In: 2014 IEEE network operations and management symposium (NOMS), IEEE, pp 1–9 (2014)
6. Bera, S., Misra, S., Vasilakos, A.V.: Software-defined networking for internet of things: a survey. IEEE Internet Things J. **4**(6), 1994–2008 (2017)
7. Das, S., Sahni, S.: Network topology optimization for data aggregation. In: 2014 14th IEEE/ACM international symposium on cluster, cloud and grid computing, pp. 493–501. IEEE, Piscataway (2014)
8. Hernandez-Valencia, E., Izzo, S., Polonsky, B.: How will nfv/sdn transform service provider opex? IEEE Network **29**(3), 60–67 (2015)
9. Orphanoudakis, T.G., Matrakidis, C., Stavdas, A.: Next generation optical network architecture featuring distributed aggregation, network processing and information routing. In: 2014 European conference on networks and communications (EuCNC), IEEE, pp 1–5 (2014)
10. Hakiri, A., Berthou, P., Gokhale, A., Abdellatif, S.: Publish/subscribe-enabled software defined networking for efficient and scalable iot communications. IEEE Commun. Mag. **53**(9), 48–54 (2015)
11. Sekar, V., Egi, N., Ratnasamy, S., Reiter, M.K., Shi, G.: Design and implementation of a consolidated middlebox architecture. In: 9th {USENIX} Symposium on networked systems design and implementation ({NSDI} 12), pp 323–336 (2012)
12. Diro, A.A., Reda, H.T., Chilamkurti, N.: Differential flow space allocation scheme in sdn based fog computing for IoT applications. J. Ambient Intell. Humaniz. Comput. (2018). https://doi.org/10.1007/s12652-017-0677-z
13. Khosravi, H.A., Khayyambashi, M.R.: Load-aware virtual network service over a software defined data center network. In: 7'th international symposium on telecommunications (IST'2014), IEEE, pp 623–628 (2014)
14. Jazaeri, S.S., Jabbehdari, S., Asghari, P., Javadi, H.H.S.: Edge computing in sdn-iot networks: a systematic review of issues, challenges and solutions. Clust. Comput. (2021). https://doi.org/10.1007/s10586-021-03311-6
15. Tang, F., Fadlullah, Z.M., Mao, B., Kato, N.: An intelligent traffic load prediction-based adaptive channel assignment algorithm in sdn-iot: A deep learning approach. IEEE Internet Things J. **5**(6), 5141–5154 (2018)
16. Jin, J., Gubbi, J., Luo, T., Palaniswami, M.: Network architecture and qos issues in the internet of things for a smart city. In: 2012 international symposium on communications and information technologies (ISCIT), IEEE, pp 956–961 (2012)
17. Selem, E., Fatehy, M., Abd El-Kader, S.M.: mobthe (mobile temperature heterogeneity energy) aware routing protocol for wban iot health application. IEEE Access **9**, 18692–18705 (2021)
18. León, L.F.A.: Eyes on the road: surveillance logics in the autonomous vehicle economy. Surveill. Soc. **17**(1/2), 198–204 (2019)
19. Yu, W., Liang, F., He, X., Hatcher, W.G., Lu, C., Lin, J., Yang, X.: A survey on the edge computing for the internet of things. IEEE Access **6**, 6900–6919 (2017)
20. Muñoz, R., Vilalta, R., Yoshikane, N., Casellas, R., Martínez, R., Tsuritani, T., Morita, I.: Iot-aware multi-layer transport sdn and cloud architecture for traffic congestion avoidance through dynamic distribution of iot analytics. In: 2017 European conference on optical communication (ECOC), IEEE, pp 1–3 (2017)
21. Muñoz, R., Vilalta, R., Yoshikane, N., Casellas, R., Martínez, R., Tsuritani, T., Morita, I.: Integration of iot, transport sdn, and edge/cloud computing for dynamic distribution of iot analytics and efficient use of network resources. J. Lightwave Technol. **36**(7), 1420–1428 (2018)
22. Beilharz, J., Wiesner, P., Boockmeyer, A., Brokhausen, F., Behnke, I., Schmid, R., Pirl, L., Thamsen, L.: Towards a staging environment for the internet of things. Preprint at arXiv:210110697 (2021)
23. Fancy, C., Pushpalatha, M.: Traffic-aware adaptive server load balancing for software defined networks. Int. J. Electr. Comput. Eng. (2088-8708) **11**(3), 2211–2218 (2021)
24. Saha, N., Bera, S., Misra, S.: Sway: Traffic-aware qos routing in software-defined iot. IEEE Trans. Emerg. Top. Comput. **9**, 390–401 (2018a)
25. Saha, N., Misra, S., Bera, S.: Qos-aware adaptive flow-rule aggregation in software-defined iot. In: 2018 IEEE global communications conference (GLOBECOM), IEEE, pp 206–212 (2018b)
26. Gupta, H., Nath, S.B., Chakraborty, S., Ghosh, S.K.: Sdfog: A software defined computing architecture for qos aware service orchestration over edge devices. Preprint at arXiv:160901190 (2016)

27. Llopis, J.M., Pieczerak, J., Janaszka, T.: Minimizing latency of critical traffic through sdn. In: 2016 IEEE international conference on networking, architecture and storage (NAS), IEEE, pp 1–6 (2016)
28. Tomovic, S., Yoshigoe, K., Maljevic, I., Radusinovic, I.: Software-defined fog network architecture for iot. Wirel. Pers. Commun. **92**(1), 181–196 (2017)
29. Sawashima, H.: Characteristics of udp packet loss: effect of tcp traffic. proc of INET'97. (1997)
30. Misra, S., Saha, N.: Detour: dynamic task offloading in software-defined fog for iot applications. IEEE J. Sel. Areas Commun. **37**(5), 1159–1166 (2019)
31. Yen, J.Y.: Finding the k shortest loopless paths in a network. Manage. Sci. **17**(11), 712–716 (1971)
32. Bera, S., Misra, S., Saha, N.: Traffic-aware dynamic controller assignment in sdn. IEEE Trans. Commun. **68**(7), 4375–4382 (2020)
33. Bizanis, N., Kuipers, F.A.: Sdn and virtualization solutions for the internet of things: a survey. IEEE Access **4**, 5591–5606 (2016)
34. Mao, B., Tang, F., Fadlullah, Z.M., Kato, N., Akashi, O., Inoue, T., Mizutani, K.: A novel non-supervised deep-learning-based network traffic control method for software defined wireless networks. IEEE Wirel. Commun. **25**(4), 74–81 (2018)
35. Berde, P., Gerola, M., Hart, J., Higuchi, Y., Kobayashi, M., Koide, T., Lantz, B., O'Connor, B., Radoslavov, P., Snow, W., et al.: Onos: towards an open, distributed sdn os. In: Proceedings of the third workshop on Hot topics in software defined networking, pp 1–6 (2014)
36. Docker (2021) Docker. https://www.docker.com/. Accessed 1 Apr 2021
37. Kubernetes (2021) Kubernetes. https://kubernetes.io/. Accessed 1 Apr 2021
38. Elrawy, M.F., Awad, A.I., Hamed, H.: Intrusion detection systems for IoT-based smart environments: a survey. J. Cloud Comput. Adv. Syst. Appl. (2018). https://doi.org/10.1186/s13677-018-0123-6
39. Spadaccino, P., Cuomo, F.: Intrusion detection systems for iot: opportunities and challenges offered by edge computing. Preprint at arXiv:abs/2012.01174 (2020)
40. Alomari, Z., Zhani, M.F., Aloqaily, M., Bouachir, O.: On minimizing synchronization cost in nfv-based environments. In: 2020 16th international conference on ntwork and service management (CNSM), pp 1–9, https://doi.org/10.23919/CNSM50824.2020.9269121 (2020)
41. Riverbed (2021) Riverbed. https://www.riverbed.com/gb/products/npm/riverbed-modeler.html. Accessed 1 Apr 2021
42. Chen, M., Miao, Y., Humar, I.: OPNET IoT simulation. Springer (2019)
43. Hakiri, A., Sellami, B., Patil, P., Berthou, P., Gokhale, A.: Managing wireless fog networks using software-defined networking. In: 2017 IEEE/ACS 14th international conference on computer systems and applications (AICCSA), IEEE, pp 1149–1156 (2017)
44. Bellavista, P., Giannelli, C., Lagkas, T., Sarigiannidis, P.: Quality management of surveillance multimedia streams via federated sdn controllers in fiwi-iot integrated deployment environments. IEEE Access **6**, 21324–21341 (2018)
45. Adhami, H., Al Ja'afreh, M., El Saddik, A.: Ontology based framework for tactile internet applications. In: International conference on smart multimedia, Springer, Cham, pp 81–86 (2019)

Mohammad Al Ja'afreh received the B.Sc degree (egregia cum laude) in Computer Engineering from Mutah University, Jordan, in 2008, and M.A.Sc. (hons.) degree in Electrical and Computer Engineering from the University of Ottawa, Ontario, Canada, in 2013, where he is currently working toward the Ph.D. degree. During his graduate studies, Mohammad received some prestigious scholarships such as the International Germany Jordanian University Graduate sponsorship and Ontario Graduate Scholarship. He has been a Research Assistant in the School of Electrical Engineering and Computer Science (EECS), University of Ottawa since 2011. His research interests include: QoE Benchmarks for Tactile Internet, Haptic Communications, and Cloud based 3D Streaming. Beside research, Mohammad holds many professional certificates in the field of Network and System Engineering such as CCNP, ACMA, and MCT. He was a Network Engineer for over 6 years in the Information Technology Section at Quadra Systems Corporation, German Jordanian University, Central Bank of Jordan, and Bahrain Telecommunications Company.

Hikmat Adhami is PHD student at the University of Ottawa, Canada. He received an Engineering degree in Electricity from the Lebanese University (LUFE-1) in 1995, and a Master degree in Telecommunications from The Ecole Nationale Supérieur in (ENST-Paris) in 2003. Since his graduation from the LUFE-1, he was the Switching manager and Head of Training department at OGERO Telecom. in North Lebanon, and a lecturer at the LUFE. His current research interests are: Tactile Internet, software-defined networking, New Generation Networks, and Ontologies.

Alaa Eddin Alchalabi is a PhD candidate at the University of Ottawa. His Research interests: Artificial Intelligence, Human Computer Interaction, Cloud Gaming, Edge Computing.

Mohamed Hoda is an assistant professor at College of North Atlantic Qatar-Department of Information Technology. His research interest: Machine Learning and Cloud Computing.

Abdulmotaleb El Saddik is Distinguished University Professor at the University of Ottawa. His research focus is on the establishment of Digital Twins to facilitates the well-being of citizens using AI, AR/VR and Tactile Internet, hence allowing people to interact in real-time with one another as well as with their digital representation. He is ACM Distinguished Scientist, Fellow of the Royal Society of