# Secure, Context-Aware and QoS-enabled SDN Architecture to improve Energy Efficiency in IoT-based Smart Buildings

Akram Hakiri[1]✉[0000−0001−7151−5499], Bassem Sallemi[1,2][0000−0001−6869−3518], Fatma Ghandour[3][0000−0003−0757−6232], and Sadok Ben Yahia[2,4][0000−0001−8939−8948]

[1] University of Carthage, SYSCOM ENIT, ISSAT Mateur, Tunisia
[2] University of Tunis El Manar, Faculty of Sciences, Dept of Computer Sciences, Tunisia
[3] Planning Department, Tunisie Telecom, Les jardins du Lac II, Tunis, 1073, TN
[4] Tallinn University of Technology, Akadeemia tee 15a, Tallinn, Estonia
akram.hakiri@enit.utm.tn, sellami.bassem@gmail.com, fatma.ghandour@tunisietelecom.tn, sadok.benyahia@fst.rnu.tn

**Abstract.** Nowadays, buildings are increasingly energy intensive, as they represent almost 40% of total energy consumption and more than 35% of CO2 emissions. The excessive and unnecessary use of planet resources and the use fossil fuel and a non-renewable energy source urged government and industry to explore new research directions and utility-driven energy improvement programs to drive advances in energy-efficient. Energy efficiency in smart buildings can be achieved by introducing a context-aware Internet of Things (IoT) approach, where sensors can learn from their surrounding environment to control the actuators in a coordinated network. However, the IoT network requirements are constantly changing in unpredictable fashion, which needs faster and frequent on-demand network reconfiguration. Software Defined Network (SDN) has been envisioned as a new approach to enable a flexible and agile network programmability in diverse IoT scenarios. However, the focus has primarily been on the design of the SDN computation logic, i.e. controllers, while the dynamic delivery and operations service-inferred IoT resource allocation has been postponed.

To address this plethora of challenges, this paper we first extend Software Defined Network (SDN) with Network Function Virtualization (NFV) to support distributed IoT sensing devices automation and orchestration in micro-grid data center at the network edge of smart campus building. Second, we introduce a novel IoT data management model based on data-centric middleware IoT message broker that implements a hierarchical containment tree for retrieving sensor data from remote IoT devices. Then, we introduce a context-aware knowledge learning approach that maps raw sensing data into a meaningful context and transform them into the appropriate context representation models. Finally, we provide a proof of concept to demonstrate successful deployment and provisioning of virtualized services in the context of Smart Campus research project.

**Keywords:** Energy efficiency · Smart Building · Software Defined Networking · Internet of Things · Context-Awareness · Service Function Chaining.

## 1 Introduction

Energy consumption has increased drastically at global scale due to the growing urbanization in cities. Buildings account nearly 40% of the global energy consumption [4] [5] and more than 35% of CO2 emissions [1] in many countries. Governments and industry are exploring new research directions and utility-driven energy improvement programs to drive advances in energy-efficient, by motivating the awareness of society to become energy conscious and adopt energy conservation and energy efficiency measures, and developing micro-grids systems that can be managed intelligently to save energy within a building or even within a city. Despite the promise, recent behavioral studies [26] have shown that it is still a challenging issue to incentive human behavior from an energy perspective [22]. For example, in campus housing and residential education buildings, which represent 9% of operating micro-grids [9], users are often unaware that changing their daily routines, e.g. doing laundry by night or setting at maintenance temperature their Heating Ventilation and Air-Conditioning (HVAC) equipment's, turning off unnecessary lighting and projectors, etc. could impact their energy consumption patterns. Consequently, achieving energy efficiency goals despite the challenges of changing users habits becomes a key challenging issue.

The proliferation of diverse Internet of Things (IoT) technologies such smart meters, embedded sensing and networking at various levels of power grids, which makes it possible to achieve energy saving and information sharing among sensing devices and actuators [31]. However, because power-grids generate enormous amounts of raw data from different zones in buildings, these data embrace uncertainty and imperfection due to the inaccuracies and imprecision inherent in data sources. That is, in order to implement an efficient smart energy management system, collected data should be reasoned to unearth knowledge and reflect a meaningful context [23]. The context-awareness should be used to react and trigger specific events to change the behavior of IoT devices, e.g. automating and adapting the indoor lighting, closing curtains, switching off the light, ensure the comfort level by controlling the HVAC equipment's, etc.

Additionally, the heterogeneous and dynamic aspects of IoT sensors generating these raw data pose major challenges for the underlying network by requiring support for handling heterogeneity, dynamic changes, device discovery as well as context-awareness. Aggregating heterogeneous sensory data from different types of sources needs an agile infrastructure that embraces message brokers, sensor virtualization and softwarization for flexible, cost-effective, secure, and private IoT deployment for diverse applications and services. Existing standardized communication protocols [12], such as IEC 61850, IEEE P1547.8, and Modbus, are designed in isolation to solve a specific problem and are often retrofitted to address a new requirement. They often lack the right abstractions that address the interoperability requirements of IoT communication. Thus, if network resource utilization is a concern, the network must be flexible enough to be reprogrammed in accordance with any change in IoT application needs. Current network provisioning approaches neither address the dynamicity of IoT applications nor care about resource utilization.

To address these key challenges, there is a need to enhance the future micro-grids industry through intelligence to enable a successful deployment and realization of powerful and trusted IoT networks. Software Defined Network (SDN) [25] [7] [11] and

Network Function Virtualization (NFV) [30] [2] show a significant promise in meeting micro-grids communication needs. SDN can achieve fine-grained resource management, enforce traffic forwarding policies and keep the micro-grid network overhead as simple and minimal as possible. SDN can also solve interoperability issues in smart micro-grids communication, as it can deal with heterogeneous devices exchanging data formats and diverse protocols for M2M data exchange. It can also improve cooperation and capability mismatch between IoT devices to handle simultaneous connections of various communication technologies. With the help of NFV, SDN can virtualize a set of network functions by deploying them into software packages, assemble and chain them whenever required to deliver chained services to IoT devices. The combination of SDN and NFV allows increasing the efficiency and capacity of smart micro-grid networks without radically making change at the hardware level. Expanding the micro-grids network capacity can be achieved using an orchestrator, which can add new services without interrupting existing one or upgrading the network with new devices. The orchestrator is the NFV management and network orchestration (MANO) tool, which is responsible for controlling and managing NFV compute, storage, and network resources.

Besides, SDN and Data analytics are needed for context-aware data processing, filtering, aggregation, and mining to capture the knowledge and generate high-level abstracted context information. It can also be used to react and trigger specific events to change the behavior of IoT devices, e.g. automating and adapting the indoor lighting, closing curtains, switching off the light, ensure the comfort level by controlling the HVAC equipment's, etc. SDN and Big Data analytics could also be used to offload some functions from sensors, i.e. by allowing the creation of software sensors inside mediation servers, etc. Therefore, first we need a powerful approach to allow smart devices understanding their own context and adapt themselves "on-the-fly" for optimal energy-efficiency and performance under all communication scenario. Second, we also need to bring the computation from cloud computing infrastructures the network edge in single hop proximity of IoT sensing devices to create micro data centers, i.e. cloudlets. Such cloudlets should support context semantics to achieve QoS prioritization among distributed HVAC sensors and/or actuators and differentiate the traffic exchanged among them. Third, we need a centralized management and configuration of the cloudlets infrastructure, orchestrate the communication in a flexible and agile manner.

In this paper, we introduce an intuitive system, less computation intensive, easy to implement on small modular low-cost Single Board Computer like Raspberry Pi, and amenable to online adaptation to the variations in ambient temperature, solar heat input through windows etc. In the context of our energy-efficiency management system in micro data center, we make the following contributions in this paper:

- We present a novel IoT network virtualization approach based on SDN/NFV to offer a high degree of automation in service chaining delivery for IoT devices. Our solution enables offloading expensive computation at the network edge and offers a high degree of automation in service chaining delivery for IoT devices.
- We introduce an IoT data model that provides a data collection facility to accommodate HVAC sensors and actuators, where data are generated and consumed locally in smart campus buildings.

- We describe a novel Context-Awareness Model to represent the functional intelligence that identifies a set of contexts, transition rules, dependencies, and relations between contexts, to control energy appliances in smart buildings .
- We provide a novel approach to realize context information enhanced with Time Series Data Repository (TSDR) approach to model hierarchical structures and relationships for collecting, storing, querying, and maintaining time series data in the SDN controller.
- We present a prototype implementation to show the proposed architecture framework can be deployed in both educational and residential smart campus buildings using low-cost hardware and lightweight Docker virtualization.

The remainder of this paper is organized as follows: Section 2 describes the related work on context-aware energy management IoT systems and highlights the use of SDN for context IoT data delivery. Section 3 devolves into the proposed architecture, which integrates our context-aware algorithm into a SDN infrastructure to realize efficient energy management in the network edge. Section 4 presents the details of a proof-of-concept implementation along with two use case which will benefit from the proposed architecture. Section 5 presents concluding remarks alluding to lessons learned and future work.

## 2 Related work

This section describes the related work on context aware sensing towards an efficient energy management systems and highlights the researches promoting SDN for programmable provision of context aware data delivery.

### 2.1 Context-aware Sensing in Smart Buildings

Service oriented middleware such as has been introduced for integrating heterogeneous IoT domain context data into a unified context broker. For example, COLLECT [20] receives contextual complex events, performs reasoning with context rules, and provides advises about the actions to take on the detected event patterns. A context aware sensing approach for efficient energy communication in IoT is described in [24] to allow IoT devices learning contextual information, e.g. channel conditions, QoS demand, battery condition, etc. from their own experience and adapt themselves "on-the-fly" for optimal energy efficiency. Additionally, a semantic reasoning system based on Semantic Web technologies was proposed [15] [6] to facilitate interoperability among diverse IoT applications operating in a realistic context-aware environment. Likewise, [17] introduced a Semantic Web server for logging temperature values and store the gathered raw data in a time-series database.

The CAEMS framework [13] uses an off-line ontology model for sharing the context knowledge on behalf of the energy consumption, and predict the expected future context information based on the historical data values. Venkatesh et al. [29] introduced an ontology-based context engine to reduce the compute redundancy and computational complexity of the IoT distributed infrastructure. A middleware layer is used to translate low-level heterogeneous data gathered from wearable computing sensors, e.g. GPS

location, blood pressure, heartbeat, etc. to a single higher-level abstracted context. The ContQuest [21] service oriented middleware uses an ontology-based context reasoning agent captures the context-related characteristics and maps these resources into a uniform description model. Narendra et al. [18] proposed a goal-driven context-aware data filtering algorithm was aligned with the semantic sensor network ontology.

Nonetheless, Semantic Web technologies is strongly coupled with applications and its expressive power does not allow advanced capabilities to express the logical and semantic reasoning context. Ontology representation is resource intensive and can be complex to perform information retrieval. Moreover, data need to be modeled in a compatible format (e.g. OWL, RDF), which limits its performance and numerical reasoning. Our approach used key-value modeling to facilitate data retrieval from Big Data store and introduced a transformation model for collecting, storing, querying, and maintaining context-aware data in the SDN controller.

### 2.2 SDN for Context Data Delivery

SDN has been used a context-aware network service layer to enable self-adaptation of virtual network function deployment [16]. The authors in [27] enhanced the SDN controller to create contextual overlay network graphs and monitors the network links and nodes as well as performing data analytics and edge processing and storage. They also introduced soft sensors [28] to obtain spatial and temporal distribution of content requests to perform optimal content placement in the cloud for mobile users. Similarly, Haw et al. [8] proposed a context-aware content delivery scheme that combines SDN and Content Centric Networking (CCN) to improve content delivery in mobile cloud networks. Luc et al. [14] introduced a context-aware traffic forwarding service to create a composition scheme between the SDN controller and the upper network applications while providing an abstracted forwarding functions for the lower layer.

Likewise, Du et al. [3] extended the OpenFlow model to IoT gateways at the network edge to attach contextual information, e.g. geo-localization data from GPS sensors and wearable devices, to OpenFlow packet header. Kathiravelu et al. [10] introduced the Cassowary middleware based on AMQP as a northbound binding to integrate sensing devices into the SDN network seamlessly. An AMQP broker extracts features available from sensing devices such as motion, temperature or humidity detection and uses a context awareness reasoning model to filter and process the collected raw data. Nova et al. [19] introduced the Capillary Network Platform for interconnecting and virtualizing IoT devices over the cellular backbone.

Despite the promise, these contributions lack of a precise context reasoning model since the context is not well-defined or vaguely introduced. Our approach provides an efficient data aggregation and data query to enabled centralized topology view that can be leveraged by multiple data-driven IoT applications. It used SDN northbound interface to query data from data store, and leverages Grafana to visualize the collected data. Moreover, it provides NetFlow data retrieving to collect SDN Flow data and logging from the underlying network elements in the micro data center in the edge domain.

## 3 Design and Implementation of The proposed solution

This section presents the architectural details of our proposed solution to realize our SDN-enabled efficient energy management system.

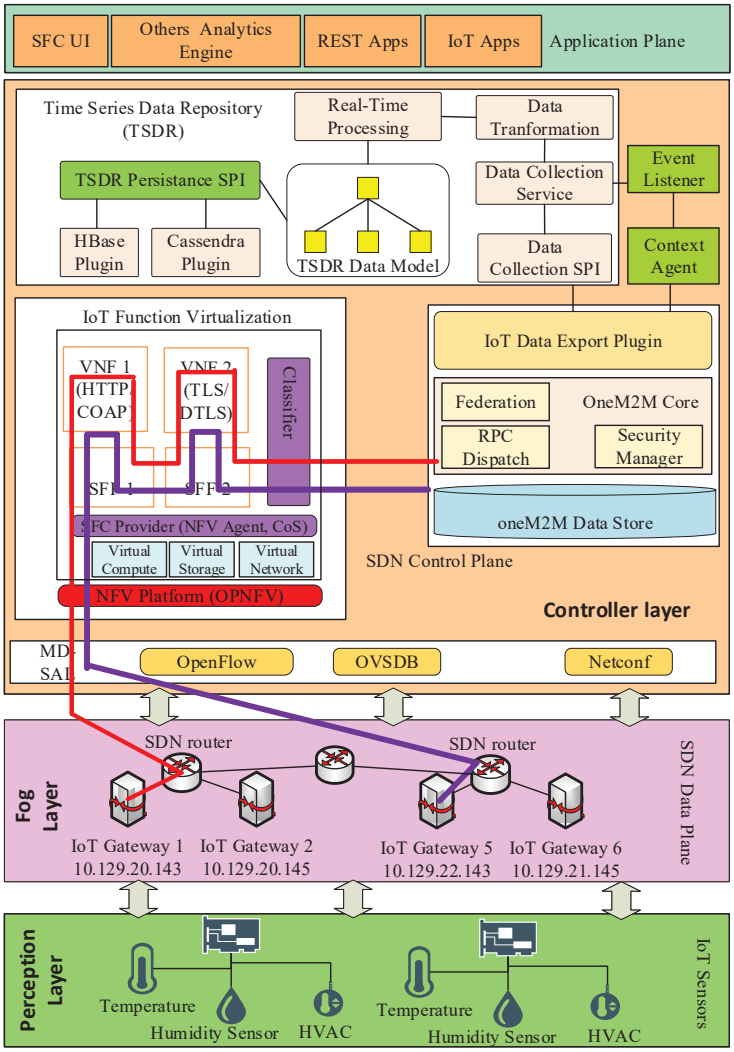


Fig. 1: Layered Architectural Overview of the SDN-enabled Framework

### 3.1 Architectural Overview

Figure 1 illustrates the architecture of our framework, which is composed of three layers: at the bottom, is the *perception layer*, which comprises the sensing layer and the

aggregator. The sensing layer encompasses all smart IoT sensors, e.g. temperature, humidity, air quality sensors, and the HVAC actuators, which have direct connectivity to the network via short range PAN technologies such RF, Bluetooth, ZigBee, etc. Aggregator sinks collect data from sensing layer and act as bridges between sensor nodes and IoT gateways, which act as a network proxy with the rest of the network. IoT sensors and actuators use message queuing client (MQTT) for publishing and subscribing data to/from IoT gateways. For example, IoT sensors gather temperature readings periodically, listen to network events through MQTT protocol, and send commands to IoT gateways to control fans and HVAC system. These IoT devices use four MQTT messages to publish-subscribe data: i) *Connect* message allows client to connect to remote M2M broker inside IoT gateways. It trigger a callback function to handle any incoming MQTT messages on the subscribed topics; ii) *Send* message is used by sensors to publish data to IoT brokers and receive command data back to it; *Store* message allows MQTT traffic incoming to brokers to be stored in time series database for retrieval in the future, and iii) *Use* message allows using Restful API for client applications to make use of their stored data. Furthermore, these IoT devices make of three QoS levels (i.e. QoS 0, QoS 1, and QoS 2) to create different priority levels for the published data.

Next, we have the *fog layer*, which is located in single hop proximity of the IoT sensing devices and include all the network equipment to realize the micro-grid communication infrastructure. It consists of IoT gateways (shown in Figure 2) that interface with the perception layer to receive raw data from the sensors, and send commands to control the actuators. The gateway concept is prevalent in home ADSL models and WiFi access points. The design of IoT gateway is different since it should be able to integrate heterogeneous smart objects and expose their resources and make them available to the rest of IoT network. Additionally, IoT gateways are connected to the SDN network through SDN routers (i.e. virtual and physical), which embed an OpenFlow agent that has the capabilities to add, remove, update, and delete packets inside these routing devices.

The SDN routers (OpenVSwitch virtual routers in Figure 2) are connected to a *SDN control layer* as shown in Figure 1, i.e. the SDN controller, which embeds all the intelligence and maintains the network-wide view of the data path elements and links that connect them. The controller contains several modules we develop to integrate to the smart micro-grid network. First, the *IoT function virtualization module* (will be described in Section 3.3), which expands the micro-grids network capacity by deploying virtualized IoT functions into software packages that can be assembled and chained whenever required to deliver chained services to IoT devices. Thanks to NFV platform (e.g. OPNFV in Figure 1) that encompasses an orchestrator (i.e. SFC provider) which can add new services without interrupting existing one or upgrading the network with new devices. The orchestrator is the NFV management and network orchestration (MANO) tool, which is responsible for controlling and managing NFV compute, storage, and network resources.

Second, the SDN control layer contains the IoT management model to perform communication with IoT sensors through IoT data management and service capability's module i.e. *IoT Data Export Plugin* (will be described in Section 3.2) for accomplishing M2M operations at scale. It also includes a *context agent* (will be discussed in Section 3.4) embeds the context-aware model to perform context reasoning needed for

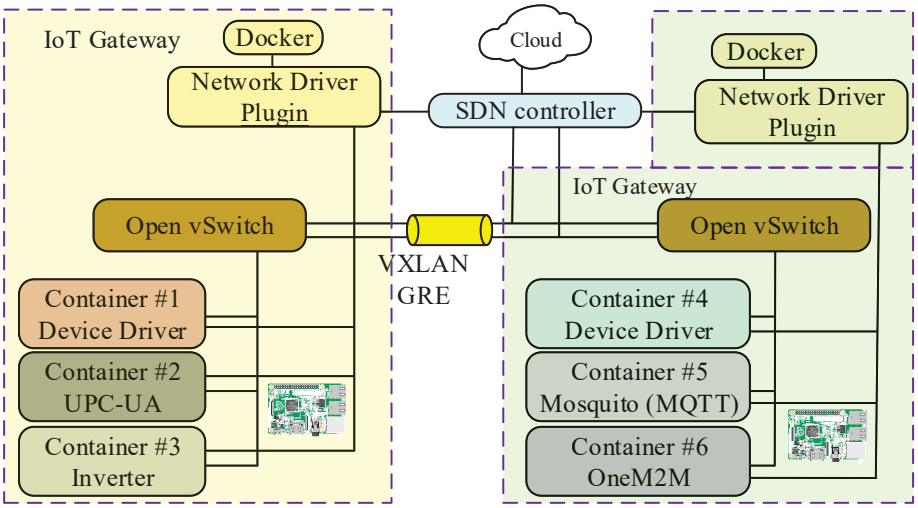


Fig. 2: IoT Gateways

data processing, filtering, and aggregation, and to capture the knowledge and generate high-level abstracted context information. Third, the controller layer comprises a data processing module that make use of a Time Series Data Repository (TSDR) module to perform real-time data processing and analytics, data transformation and collection services: upon received at the data store, the IoT Data Export Plugin triggers CRUD handling operations to enable writing data in the data-store before connecting them to the context-aware middleware to perform data processing. The TSDR module connects to Cassandra and HBase NoSQL database management system through plugins.

### 3.2 IoT Data Management Model

The IoT data management model is based on a data-centric IoT message broker that make use of standardized oneM2M API to allow authorized sensors and applications retrieving the data stored by other devices. In particular, the model implements a hierarchical containment tree where each node in the tree represents an IoT resource. As depicted in Figure 3, the tree contains different data and measurements of IoT devices and their associated attributes. Each node in the tree represents a specific resource an IoT device can interact with using either the Message Queuing Telemetry Transport (MQTT) broker or direct HTTP-like message exchange. Those attributes provide resources description in a form of meta-data that includes information about the resource creation, access rights, creator of the resource, content size, creation time and date, etc.

As shown in Figure 1, the oneM2M Data Store represents the back-end database where raw sensor data are stored. The proposed architecture supports CRUDN (create, retrieve, update, delete, and notify) operations for collecting sensing data from remote IoT sensors. Such an approach allows us to easily perform inventory with life-cycle

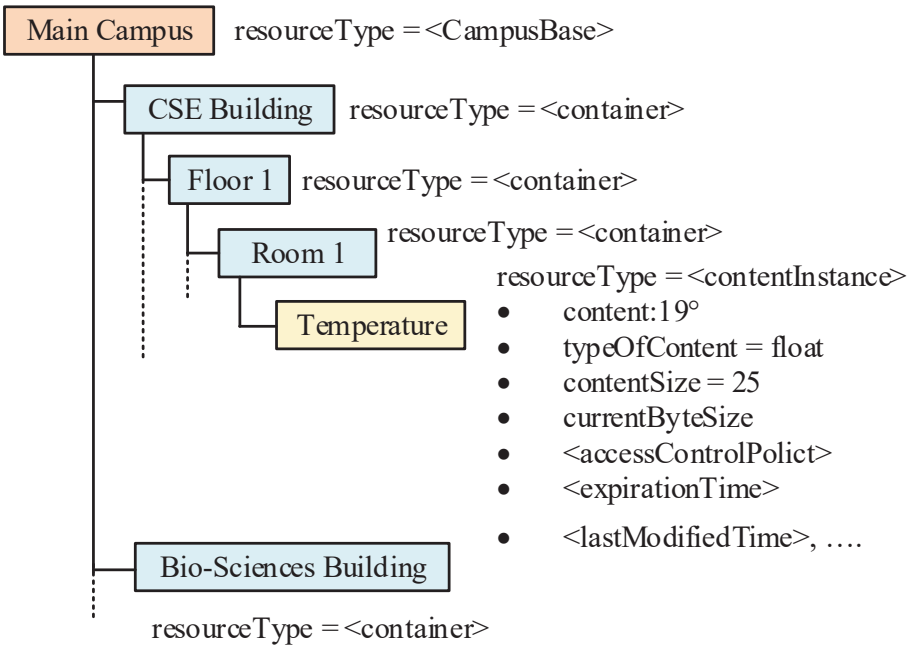


Fig. 3: Resources tree comprising IoT data.

management of IoT devices and perform big data analytics on raw sensor's data, retrieve and transform them into appropriate context representation.

### 3.3 IoT Service Chaining

In Figure 1, the *IoT Function Virtualization (IoT NFV)* module shows our proposal for mapping IoT sensors into virtualized functions that follows the ETSI guideline for NFV architectural framework. The SDN controller is merged with the IoT management platform (i.e. NFV Platform) to perform communication between IoT gateways and their remote sensors and enforce the security of state information. The IoT-NFV module uses lightweight containers to enable the creation of multiple isolated virtual IoT gateways inside a single physical one. For example, VNF1 and VNF2 in Figure 1 represent two independent virtualized functions chained to form a single IoT service. Constrained Application Protocol (CoAP) messages in VNF1 coming from sensors in the sensing layer are chained with DTLS service to enforce the security of IoT resources. Similarly, other group of sensors in the sensing layer that make use of HTTP/REST services in VNF1 can see their messages chained and secured with TLS in VNF2 to optimize the cooperation between IoT devices, intermediate infrastructure and the rest of the IoT network. The virtualization layer is based on lightweight containers using Docker. Thus, it becomes fast to create, install, run and deploy independent micro-services and provide simple service composition facilities.

Routing the packets among these VNF components is completely managed and controlled by the SDN controller. Thanks to Pipework and the "overlay" mode of Open-VSwitch software router. The former allows connecting together multiple containers in arbitrarily complex scenarios. The later provides a kind of private IP addresses that are only valid internally. Each IP address $P$ identifies a service deployment in a separate chain, so that the SDN controller can program the flow table with the required flow entries $F_P$ to define the following component $B = F_P(A)$ in the chain for which the traffic will be forwarded. The controller creates for each flow entry $F_P$ the forward table entries that matches received packets against the forwarding ports $A$ they should follow with P as the destination address.

### 3.4 Context-Awareness Model

The context reasoning model identifies a set of contexts, transition rules, dependencies, and relations between contexts. Figure 4 illustrates the context-awareness model which includes a 3-tuple ($A_k$, $T_k$, $D_k$), Where is $A_k$ is the action knowledge, $T_k$ is Transitional knowledge, and $D_k$ declarative knowledge. The action knowledge represents the functional intelligence for a given environment, which is coded using logic rules or machine learning algorithm. For example, given a current temperature inside a room, the system should switch on or off HVAC system of the building.
The transition knowledge specifies when a passage to another context should be performed. It can be expressed for example as IF(conditions) then(activation) transition rules, fuzzy rules or neural network probability condition. Finally, the declarative knowledge provides description of some aspects of the context to include some pre-acquired knowledge for the context. For example, number of invited guest in convocation hall, room size, room usage schedule, etc.

The context reasoning model is used to subdivide the knowledge into multi-level hierarchy, each level is used to represent vertical relationship between sub-levels. Figure 5 illustrates an example context hierarchy for an educational building. The vertical multi-level hierarchy describes the relationship between groups in a given set $G = \{G_1, G_2, ..., G_n\}$, where a given group $G_i \in G$ contains a set of mutually exclusive contexts $C_i = C_0^i, ..., C_a^i, ..., C_n^i$. An active context $C_a^i$ in a subset of group $G_i$ is active within the context of its parents. That is, to make the inheritance active, transitional and declarative knowledge from selected contexts in groups that are hierarchically above $G_i$.

In our case, Figure 5 illustrates the reasoning model where Campus buildings are the first group $G_1$. The second group $G_2$, we identify different types of buildings, e.g. educational, residential, administrative, laboratory buildings, etc. To apply this context-aware reasoning model to our case, we should activate the set of contexts that best suits to a given situation. The active context will control all the execution process, define behaviors of our actuators, specify the constraints and all other context-dependent characteristics. For seek of simplicity, we define an example of our context reasoning model from Figure 5 as follows:

- $C^t_{Active}$ = {Buildings, Educational, Spring, Working-Day, Day}.
- $C^{t+1}_{Active}$ = {Buildings, Residential, Spring, Working-Day, Night}.

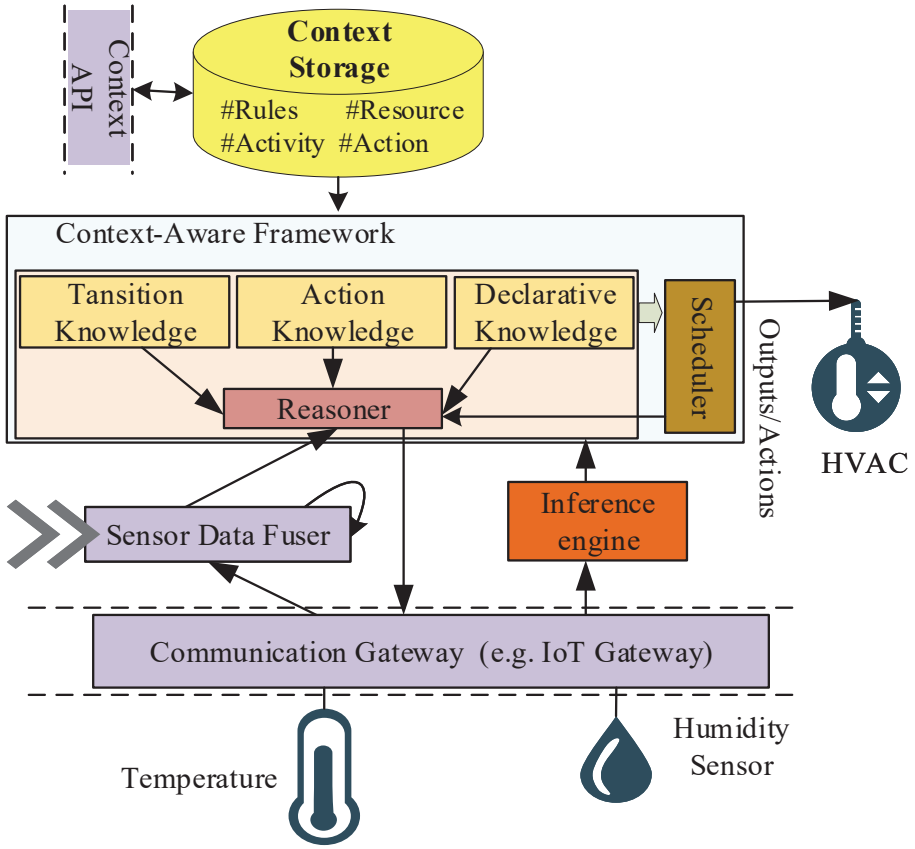


Fig. 4: Context-Awareness Model

– $C^{t+2}_{Active}$ = {Buildings, Educational, Winter, Holiday, Day}.

Specifically, given active contexts $C^{t}_{Active}$ and $C^{t+1}_{Active}$ at certain instances $t$ and $t+1$, it might be seen easy or even not much complicated to identify the process to manage context operation, in particular given the slow evolution of sensor variables such as temperature, humidity and CO2 level inside rooms in education buildings. However, given different situation that occur in a given group $G_i$, we identify a domain of services $s_i$ which has its own execution thread(s) and its control is a function of context $C^{t}_{Active}$. The control of service $s_i$ can be defined by equation 1:

$$\texttt{Control of } s_i = \Gamma(C^{t}_{Active}) \tag{1}$$

Where $\Gamma$ is the context reasoning framework operating within $s_i$. For example, room temperature behave differently if a door/window is open/closed or if the room is empty or occupied, electrical lights and fans behave accordingly to these situations.

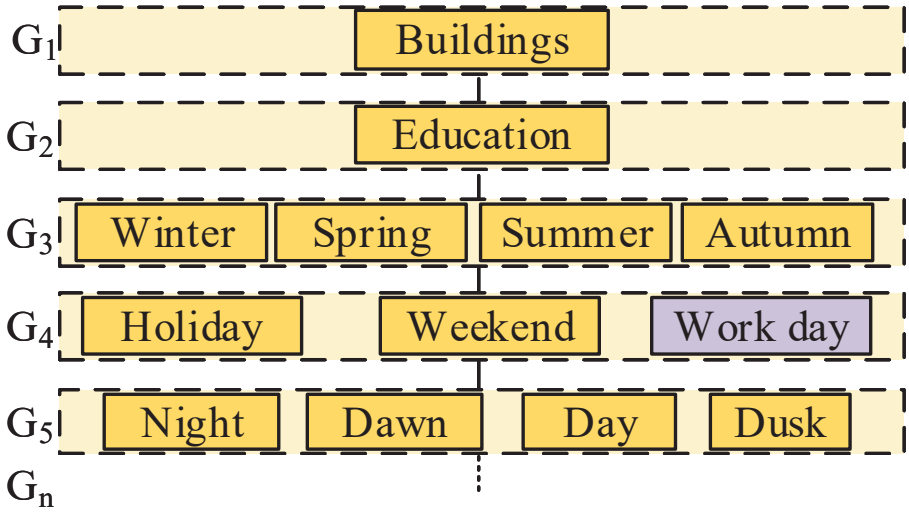


Fig. 5: Example of context hierarchy levels for an education building

### 3.5 Machine Learning Engine

The machine learning engine capabilities are twofold: first, they help the context reasoning framework to infer the context decisions. Second, they feed back the environment changes to the SDN controller to perform automatic traffic steering and policy placement. For the former, the machine learning engine monitors current sensor's data delivery and predicts future data and learn the optimal policy for the network management. In particular, the energy demand is variable in time and space, since its consumption varies qualitatively and quantitatively on the time of days, e.g. working days, holidays, week-end, etc. or the location where IoT devices are deployed, e.g. laboratories, classrooms, office building, etc. The machine learning engine provides better personalized experience and give priorities to specific IoT devices that should communicate relevant information to the SDN controller. It also enables storing and processing the collected data to analyze the data-sets based on certain parameters such as location, time, and historical data.

For the later, Figure 6 depicts the machine learning module where the SDN controller continuously learns from data generated by virtual routers and becomes aware of the runtime status of the network. The controller collects OpenFlow statistics (circle 1), applies ML algorithm i.e. multi-layer perceptron (circle 2), and take the right decisions that adjust the policies (i.e. traffic flow redirection from $A \Rightarrow D$ to $A \Rightarrow B \Rightarrow D$ or $A \Rightarrow C \Rightarrow D$) for traffic classification and traffic shaping, dynamically change these policies according to the analytics results, and feed back these results to the forwarding path for automatic steering and policy placement. The controller can also use the machine learning capabilities to establish normalized profiles to predict traffic pattern, and

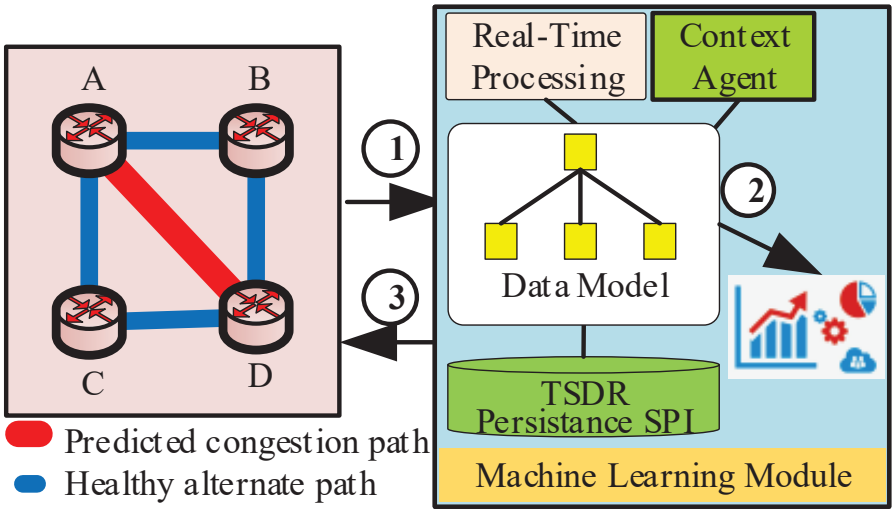


Fig. 6: Traffic congestion prediction with automated control.

perform routing optimization based on predetermined or dynamically update learned rules.

## 4 Proof of Concept Implementation

This section describes the primary prototype implementation we realized to verify the feasibility of running the platform. We first introduce the platform implementation, then we highlight two application test cases.

### 4.1 Platform

Figure 7 depicts our target application. IoT gateways are based on a single-board computer Raspberry Pi 3 with 1 GB of memory and a quad-core ARMv8 BCM2837B0 Cortex-A53 ARM Cortex-A53 CPU running at 1.2 GHz. The gateway connects to the network using its integrated 2.4GHz 802.11n interface. It also contains *hostapd* user space daemon software we used to create virtual wireless networks inside the same physical one. IoT gateways are equipped with 40 pins GPIO interfaces to connect to wide range of sensors and actuators.

We deployed several sensor boards that act as Cluster Head (CH) nodes to collect raw sensor data and measure the surrounding air quality from all Cluster Members (CM). CMs sensor nodes periodically measures temperature, humidity and Co2 levels, compare them with the last measurement, if results had changed, they send an advertisement message (ADV) to the cluster head. Examples of CMs we used include *DHT22*

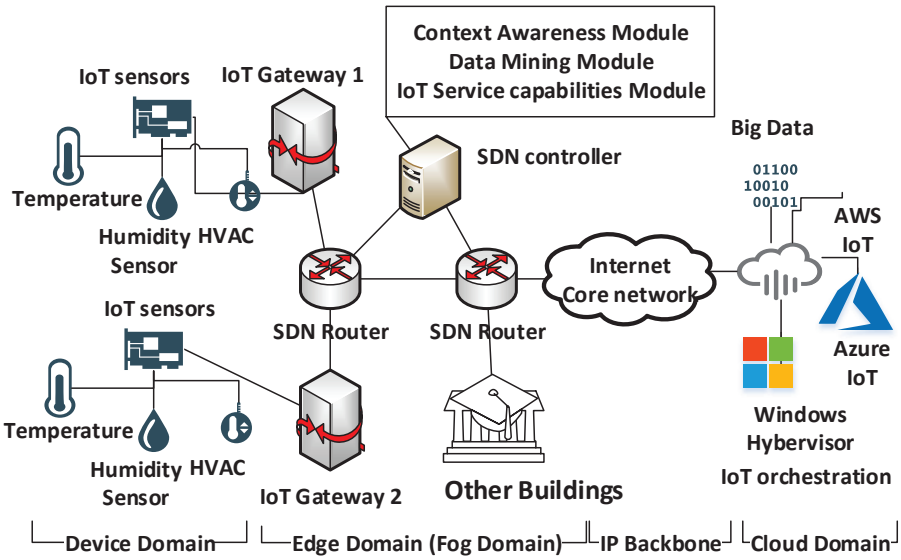


Fig. 7: Network topology used as the target application

temperature and humidity sensors; the *MQ-135* Co2-Gas sensor; and the *K30 CO2* module that gathers the level of oxygen inside our campus buildings and student housing residence. We also used smart energy meter to detect and report power consumption to CH nodes and IoT gateways. As Cluster Heads (CH) we used multiple NodeMCU IoT platforms running on-top of the ESP8266 Wi-Fi SoC, which also integrate a TCP/IP protocol stack that allows access to our virtual Wi-Fi network running on IoT gateways. We successfully connected these CHs through the MQTT broker running on IoT gateways. We used also CoAP API to connect some other sensors to ESP8266 board. COAP servers are deployed inside IoT gateways in a form of lightweight containers.

Additionally, our SDN controller is based on the OpenDaylight (ODL) Project, which has several SDN and application management capabilities. The ODL platform can be configured to run as a highly scaled up and out distributed cluster with IoT, SDN and NFV functions. It can also be integrated with OpenStack and deployed to communicate with a high traffic data center. We used OpenVSwitch (OVS) as our distributed virtual multi-layer SDN switch managed by our SDN controller. We used two OVS modes: NAT and bridge. The NAT mode is used to connect the SDN routers with outside world. The OVS bridge mode offers virtual interfaces we used to connect with docker instances to multi-host networking, i.e. we used both internal IP addresses as our pseudo-IP and MAC addresses.

### 4.2 Service Function Chain Composition in Education Building

For the education buildings, we consider diverse CH nodes in charge of collecting temperature, humidity levels and CO2 concentration from CM sensors of occupied indoor spaces. We consider three VNFs implemented inside docker images: $Tensor_{VNF}$,

$Mosquitto_{VNF}$, and $OneM2M_{VNF}$ as described in Figure 8. The first VNF monitors the current sensor's data delivery, uses TensorFlow machine learning models to predict future data from correlated sensor's readings, and learn the optimal policy for the network management by avoiding redundant readings. The second VNF gathers sensors readings periodically, listen to network events through MQTT protocol, and send commands to control fans and HVAC system. Finally, the third VNF establish the access to IoT resources through the hierarchical containment tree described in Section 3.2.

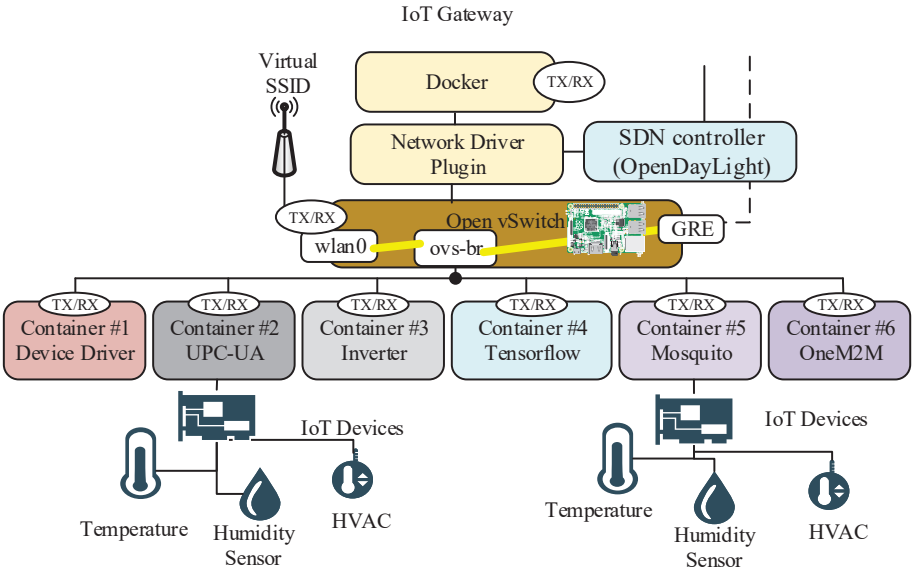


Fig. 8: Network topology used as the target application.

In order to interconnect different micro-services running in containers, we used Docker engine v19.03. Docker engine offers advanced network capabilities to manage connectivity between containers. As shown in Figure 8, we used Docker Network Driver Plugin to create virtual docker network that connects to virtual SDN routers (OpenVSwitch) and handles all coordination between virtual hosts. We also configured persistent docker volume to store alarms and other debugging events outside docker containers. We also used Vagrant for automating the creation of virtual machine instances, building and maintaining portable virtual Docker containers.

We deployed Docker swarm as our container's orchestration tool to manage different VNFs we deployed in our tests. Docker swarm offers a high level of availability for the running application. We defined one of our containers as a leader to manage membership and delegation among the other containers, which we configured as workers. It is worth noting that we can use Kubernetes (K3s) system for automating deployment, scaling, and management of VNFs. We configured Docker Swarm to use docker-compose configuration files and scripts we created to tell the docker daemon

(running inside each IoT gateway) how to pull the appropriate container image from the Docker Hub repository, how to establish networking between VNFs, how to mount storage volumes, and where to store logs for a given container.

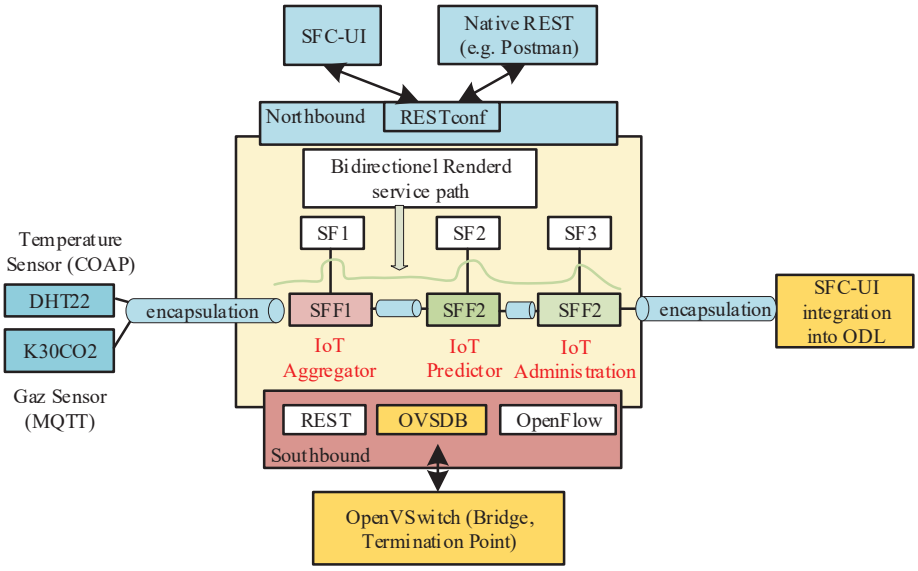


Fig. 9: SFC composition for Temperature and CO2 measurements as an IoT service.

Figure 9 describes the approach we used to configure the OpenDaylight SDN controller Service Function Chaining (SFC) to define an ordered list of network services (i.e. $Tensor_{VNF}$, $Mosquitto_{VNF}$, and $OneM2M_{VNF}$) which we stitched together to create a service chain. This SFC is arranged as: i) physical network function (PNF) that contains the IoT temperature and gas sensors, ii) three IoT VNFs depicted as Service Function Forwarders (SFF) in Figure 9, and iii) the SFC-UI integration into the SDN controller. When the IoT SF1 is updated with new sensor measurements, an update SFC composition is triggered by the docker swarm orchestrator to pull and install a new image (or migrate an existing one) for the required VNF ($Mosquitto_{VNF}$, $Tensor_{VNF}$, or $OneM2M_{VNF}$), respectively. Our implementation shows that our approach is able to create, instantiate and deploy new customized on-demand virtualized IoT services that gather, process, estimate and supervise the air conditioning inside campus buildings. Our approach successfully collects room temperature, send these values to IoT VNFs through IoT gateways, which forward them to the SDN controller to switch ON or OFF the HVAC appliances based on temperature and CO2 threshold. This result is very flexible and reconfigurable as it can instantiate and deploy VNFs as needed for the scalability and allows saving up to 70% of the energy consumption in the campus buildings.

### 4.3 Activity Management in Residential Building

Monitoring users activity in residential building is very critical to understand how they interact with IoT home appliances (e.g. TV boxes, PlayStation (PS), washing machine, Laptop, Lights, etc.) because different users activities usually call for different services. Therefore, we add an Activity Recognition (AR) model to the context-awareness model described in Section 3.4.

Table 1: Daily Occurring User Activity in smart residential building

| Activity | Sleep | Use Laptop | Study in office | Watch TV | Read Book | Play PS | Prepare food | Washing clothes | No One at home |
|---|---|---|---|---|---|---|---|---|---|
| Occurrence % | 33 | 10 | 8.3 | 12.5 | 6.8 | 7.1 | 9.2 | 4.3 | 8.8 |

The AR model describes the user behavior inside smart residential home as shown in table 1. The AR model is trained using semi-supervised learning model (which is included in the ML engine in Figure 10) to forecast the appliances a user is using during the activities describes in table 1. For example, if a user activity is "*Sleep*", only night light of bedroom should be ON and all other appliances should be OFF. Similarly, if the weather is getting warmer, the HVAC or fans should be ON automatically; if no one at home all lights and appliances should be OFF.

Figure 10 depicts the IoT gateway, which can be easily turned into a virtual SDN router using OpenVSwitch and Docker Network Driver Plugin services. The SDN controller can manage the whole Home Area network. The controller monitors link states periodically via the link layer discovery protocol and creates the network topology. Additionally, the approach we develop in this paper allows slicing the Home Area network into several separated logical networks, each network partition can deal with different QoS requirements. For example, a network partition with strict QoS requirements can be configured for CCTV Camera and video streaming, etc. This result of architecture can successfully discover a lot of useful data from the AR model. It also success in creating several separated logical networks, which improve the network agility, availability and performance. Moreover, the proposed architecture performs energy saving by automatically switching ON/OFF unnecessary home appliances based on the context-awareness and AR models. The SDN controller collect useful data for context-aware service provisioning and manages the forwarding tables of SDN routers to provide comfort and assistance to building occupants, and apply powerful learning models on collected data to derive behaviors that impact high energy consumption.

## 5 Conclusion

In order to implement a smart energy management system in smart micro-grids, in this paper we proposed a comprehensive architectural design that is devised to empower

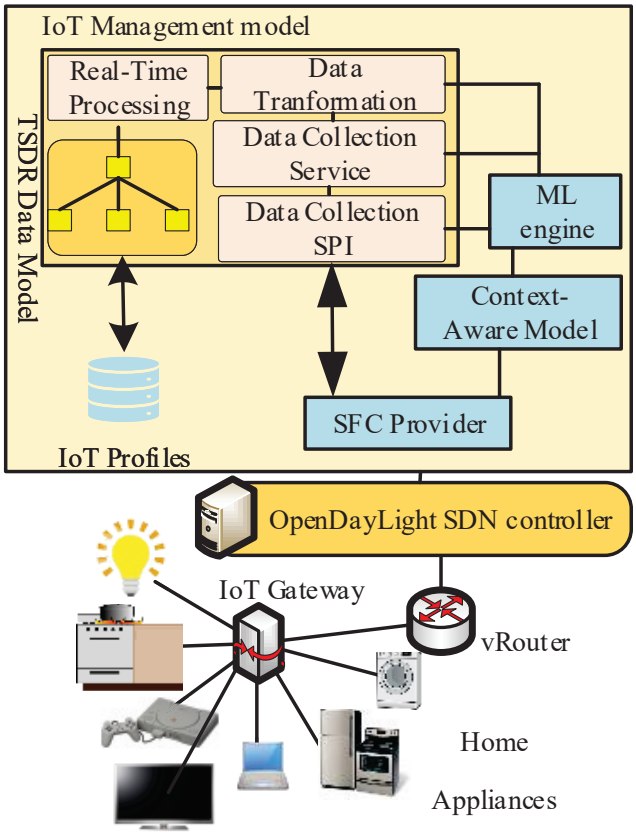


Fig. 10: Typical smart home network as an IoT service.

SDN-enabled Context-Aware IoT systems and networks to create a flexible, agile, and reconfigurable framework to improve energy efficiency in smart buildings and enable automated building operations and control. We analyzed different parameters affecting energy consumption in campus building and proposed a context-awareness model that allows an optimum prediction of users behaviors according their daily consumed energy profiles in both educational and residential buildings. Then, we introduced a IoT service chaining solution for creating and deploying customizable IoT services on-demand. Finally, we provided a machine learning engine that helps the context reasoning framework to infer the context decisions, and feed back the environment changes to the SDN controller to perform automatic traffic steering and policy placement. The architecture will also open up new perspectives towards the mass adoption of efficient energy management systems based on robust and scalable IoT services.

## Acknowledgments

This work was partially funded by the the Tunisian Ministry of Higher Education and Scientific Research (MES) under the Young Researchers Incentive Program (19PEJC09-04) and the CV Raman research program 2017675. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of MES or CV Raman program.

## References

1. Agency, E.E.: Progress on energy efficiency in europe (2019). URL https://bit.ly/2OygVJN
2. Alam, I., Sharif, K., Li, F., Latif, Z., Karim, M.M., Biswas, S., Nour, B., Wang, Y.: A survey of network virtualization techniques for internet of things using sdn and nfv. ACM Comput. Surv. **53**(2) (2020)
3. Du, P., Putra, P., Yamamoto, S., Nakao, A.: A context-aware iot architecture through software-defined data plane. In: 2016 IEEE Region 10 Symposium (TENSYMP), pp. 315–320 (2016)
4. (EIA), E.I.A.: International energy outlook (2019). URL https://bit.ly/2CFN9QK
5. of Energy's, U.D.: Increasing efficiency of building systems and technologies (2015). URL https://bit.ly/2CodZgd
6. Guner, A., Kurtel, K., Celikkan, U.: A message broker based architecture for context aware iot application development. In: 2017 International Conference on Computer Science and Engineering (UBMK), pp. 233–238 (2017)
7. Hakiri, A., Gokhale, A., Berthou, P., Schmidt, D.C., Gayraud, T.: Software-defined networking: Challenges and research opportunities for future internet. Computer Networks **75**, 453 – 471 (2014)
8. Haw, R., Alam, M.G.R., Hong, C.S.: A context-aware content delivery framework for qos in mobile cloud. In: The 16th Asia-Pacific Network Operations and Management Symposium, pp. 1–6 (2014)
9. Hirsch, A., Parag, Y., Guerrero, J.: Microgrids: A review of technologies, key drivers, and outstanding issues. Renewable and Sustainable Energy Reviews **90**, 402 – 411 (2018)
10. Kathiravelu, P., Sharifi, L., Veiga, L.: Cassowary: Middleware platform for context-aware smart buildings with software-defined sensor networks. In: Proceedings of the 2Nd Workshop on Middleware for Context-Aware Applications in the IoT, M4IoT 2015, pp. 1–6 (2015)
11. Kreutz, D., Ramos, F.M.V., Veríssimo, P.E., Rothenberg, C.E., Azodolmolky, S., Uhlig, S.: Software-defined networking: A comprehensive survey. Proceedings of the IEEE **103**(1), 14–76 (2015). https://doi.org/10.1109/JPROC.2014.2371999
12. Kumar, S., Islam, S., Jolfaei, A.: Microgrid communications - protocols and standards, pp. 291–326. Energy Engineering. Institution of Engineering and Technology (2019)
13. Kyselova, A.G., Verbitskyi, I.V., Kyselov, G.D.: Context-aware framework for energy management system. In: 2nd Int. Conf. on Intelligent Energy and Power Systems (IEPS), pp. 1–4 (2016)
14. Luo, S., Wu, J., Li, J., Guo, L., Pei, B.: Context-aware traffic forwarding service for applications in sdn. In: IEEE Int. Conf. on Smart City SocialCom SustainCom (SmartCity), pp. 557–561 (2015)
15. Maarala, A.I., Su, X., Riekki, J.: Semantic reasoning for context-aware internet of things applications. IEEE Internet of Things Journal **4**(2), 461–473 (2017)
16. Martini, B., Paganelli, F., Mohammed, A.A., Gharbaoui, M., Sgambelluri, A., Castoldi, P.: Sdn controller for context-aware data delivery in dynamic service chaining. In: Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft), pp. 1–5 (2015)

17. Najem, N., Haddou, D.B., Abid, M.R., Darhmaoui, H., Krami, N., Zytoune, O.: Context-aware wireless sensors for iot-centeric energy-efficient campuses. In: 2017 IEEE International Conference on Smart Computing (SMARTCOMP), pp. 1–6 (2017)
18. Narendra, N., Ponnalagu, K., Ghose, A., Tamilselvam, S.: Goal-driven context-aware data filtering in iot-based systems. In: 2015 IEEE 18th International Conference on Intelligent Transportation Systems, pp. 2172–2179 (2015)
19. Novo, O., Beijar, N., Ocak, M., Kjällman, J., Komu, M., Kauppinen, T.: Capillary networks - bridging the cellular and iot worlds. 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT) pp. 571–578 (2015)
20. de Prado, A.G., Ortiz, G., Boubeta-Puig, J.: Collect: Collaborative context-aware service oriented architecture for intelligent decision-making in the internet of things. Expert Systems with Applications **85**, 231 – 248 (2017)
21. Pötter, H.B., Sztajnberg, A.: Adapting heterogeneous devices into an iot context-aware infrastructure. In: 2016 IEEE/ACM 11th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), pp. 64–74 (2016)
22. Rashid, H., Mammen, P.M., Singh, S., Ramamritham, K., Singh, P., Shenoy, P.: Want to reduce energy consumption? don't depend on the consumers! In: Proceedings of the 4th ACM International Conference on Systems for Energy-Efficient Built Environments (2017)
23. Sen, S.: Invited - context-aware energy-efficient communication for iot sensor nodes. In: Proceedings of the 53rd Annual Design Automation Conference (2016)
24. Sen, S.: Invited: Context-aware energy-efficient communication for iot sensor nodes. In: 2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC), pp. 1–6 (2016)
25. Singh, S., Jha, R.K.: A survey on software defined networking: Architecture for next generation network. Journal of Network and Systems Management **25**(2), 321–374 (2016)
26. Staddon, S.C., Cycil, C., Goulden, M., Leygue, C., Spence, A.: Intervening to change behaviour and save energy in the workplace: A systematic review of available evidence. Energy Research & Social Science **17**, 30 – 51 (2016)
27. Tosic, M., Ikovic, O., Boskovic, D.: Sdn based service provisioning management in smart buildings. In: 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 754–759 (2016)
28. Tosic, M., Ikovic, O., Boskovic, D.: Soft sensors in wireless networking as enablers for sdn based management of content delivery. In: 2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 559–564 (2016)
29. Venkatesh, J., Chan, C., Akyurek, A.S., Rosing, T.S.: A modular approach to context-aware iot applications. In: IEEE First Int. Conf. on Internet-of-Things Design and Implementation (IoTDI), pp. 235–240 (2016)
30. Zhang, T.: NFV Platform Design: A Survey. arXiv: 2002.11059v2 (2020)
31. Zhu, Y., Wang, F., Yan, J.: The potential of distributed energy resources in building sustainable campus: The case of sichuan university. Energy Procedia **145**, 582 – 585 (2018)