Software-defined wireless sensor networks: A survey

Habib Mostafaei, Michael Menth

PII: S1084-8045(18)30224-8

DOI: 10.1016/j.jnca.2018.06.016

Reference: YJNCA 2166

To appear in: *Journal of Network and Computer Applications*

Received Date: 12 September 2017

Revised Date: 20 June 2018

Accepted Date: 24 June 2018

Software-Defined Wireless Sensor Networks: A Survey

Habib Mostafaei[a], Michael Menth[b]

[a]*Roma Tre University, Rome, Italy*
[b]*University of Tuebingen, Tuebingen, Germany*

Abstract

Software-defined networking (SDN) decouples data and control plane, i.e., forwarding elements are remotely configured by centralized controllers instead through distributed control protocols. Wireless sensor networks (WSNs) have mostly been controlled in a distributed way, but its configuration challenges are complex and can be theoretically better solved with network-wide knowledge – the solution just needs to be configured on the distributed sensor nodes. This calls for SDN in WSNs and so that software-defined WSNs (SD-WSNs) have been proposed. In this survey, we explain basics of WSN and SDN, describe fundamentals of SD-WSNs and how SDN can improve the operation of WSN. Furthermore, we outline the open challenges that need to be investigated in more detail and discuss lessons learned during the preparation this survey.

Keywords: Software Defined Networking (SDN), Sensor OpenFlow, Wireless Sensor Networks (WSNs), OpenFlow.

1. Introduction

Software-defined networking (SDN) is an emerging networking architecture that gives the opportunity to overcome the current limitations of the network infrastructure [1, 2]. It decouples the network's *control* plane and *data* plane. That means an intelligent controller configures forwarding elements with fine-grained forwarding rules for data packets of different flows. The controller obtains sufficient information to fulfill that task so that distributed control protocols among forwarding elements are no longer needed. Furthermore, the controller may interact with applications to optimize the network.

A wireless sensor network (WSN) consists of sensor nodes with communication, computing, and sensing capabilities. Sensor nodes mostly have batteries that limit their lifetimes. They are often randomly deployed over a larger area for monitoring purposes. Therefore, communication and sensing ranges are controlled to ensure communication with other nodes and to cover the entire area with the desired application. In the past, self-organized management with distributed control has been the intuitive approach for running WSNs. Thereby, energy saving was always an important goal to extend the lifetime of the network.

Software-defined WSNs (SD-WSNs) have been recently proposed with the objective that WSNs can particularly profit from SDN. The operation of sensor nodes should be simplified to save energy and to manage the WSN through a powerful controller which has a view on the entire network rather by distributed control protocols. The controller is able to manage the network and applications while saving energy and to deliberately balance the residual energy of the network to maximize its lifetime. A significant difference to SDN in a datacenter is that the controller in a WSN communicates with distant sensor nodes over possibly multiple hops rather than over a dedicated control network.

In this survey, we give an introduction to SDN in wireline networks and to non-SDN WSNs, we describe the architecture of SD-WSNs, illustrate their operation, point out advances and research challenges. We also compare SDN-based and non SDN-based WSNs. General requirements for deploying SDN in WSNs are surveyed in [3, 4]. Ndiaye et al. [4] focused on how WSN management can be performed by SDN. Kobo et al. [3] concentrated on the architectural view of SDN in WSNs. The authors of [5, 6] provided a survey on the application of SDN in wireless networks. However, non of these papers surveyed what can be controlled by SDN in WSNs and how applying SDN in WSNs is different from wireline networks

This work is structured as follows. Sec.2 reviews the basic concepts of WSNs. Sec. 3 describes the basic concepts of SDN. The basics for SD-WSN are described in Sec. 4. Advances in WSNs through SDN are reviewed in Sec. 5. Sec. 6 states challenges in SD-WSN. Lessons learned are reviewed in Sec. 7 and Sec. 8 concludes this survey.

2. WSN Basics

In this section we briefly introduce the basic concepts of WSNs by giving a general overview on the network structure, use cases, standards, and research challenges.

2.1. Network Structure

In a WSN, each sensor node has a sensing region that can sense the events and objects within that range. Additionally, each node can communicate over a wireless interface with other nodes that are in the communication range of this node. Fig. 1

Email addresses: mostafae@dia.uniroma3.it (Habib Mostafaei), menth@uni-tuebingen.de (Michael Menth)

shows a collection of sensors that are scattered over a network area to monitor events, e.g., the event E in the figure. The information gathered from this event is transferred to a base station (BS) through multihop communications. The BS sends the network data via the Internet to an application server.

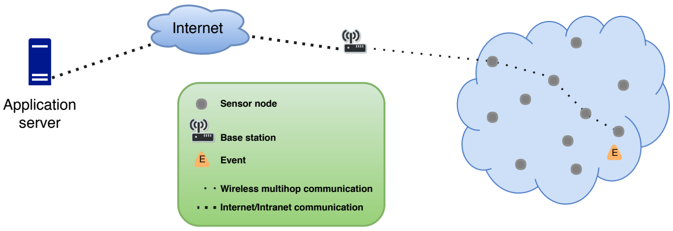

Internet
Application server
Sensor node
Base station
E
Event
Wireless multihop communication
Internet/Intranet communication
E

Figure 1: A wireless sensor network.

There are two types of WSN, namely structured and unstructured WSNs [7]. Typically, structured WSNs have a small number of sensor nodes and they are easy to manage. Sensor nodes are placed deterministically, i.e., the place of each node is determined in advance. In unstructured WSNs, many sensors are deployed in an ad-hoc manner. Therefore, the resulting WSN is more difficult to manage.

The control of WSNs can be categorized into centralized, decentralized or distributed control which are depicted in Fig. 2. With centralized control, a single node has the global view of the network and decides whether the functionality of a node is required or not, i.e., the node should be active or not. With decentralized control, the nodes are divided into groups and there is a central node for each group. The interaction among the central nodes of all groups determines the activity of each node. In distributed control, there is no central control node and all nodes interact with each other for network-wide decision making, e.g., determining the active nodes for covering the network area.

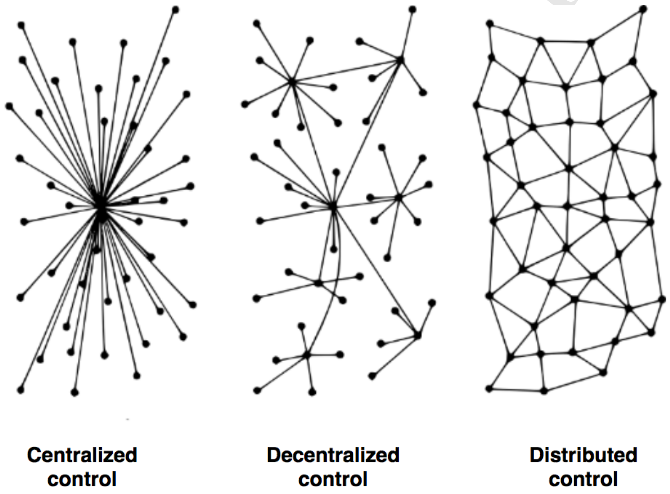


Figure 2: Different control types for WSNs.

2.2. *Typical Use Cases*

There are several types of sensors such as acoustic, thermal, visual, etc. The survey in [8] reports that sensors monitor various ambient conditions. Examples are temperature [9], habit monitoring [10], movement detection [11, 12], humidity [13], military applications [14], oil and gas monitoring [15], health monitoring [16].

The authors of [16, 8] described several application scenarios of WSNs. For example, in military applications, the sensors can be leveraged to detect the movement of vehicles or army forces in a battlefield. In health monitoring applications sensor nodes can send patient information to nursery stations or doctors to identify symptoms [8]. Generally, WSNs are mostly exploited for monitoring and tracking applications.

2.3. *Standards*

The key design challenge for wireless sensor nodes is low power consumption [16]. Standards for WSNs define sets of functions and protocols. Examples are IEEE 802.15.4 [17], Zigbee [18], 6LoWPAN [19], and ISA100.11a [20]. We briefly discuss them in the following.

IEEE 802.15.4 is designed for low-rate wireless personal area networks (LR-WPAN) [17]. The main goals of this standard are low-cost implementation, low complexity, and low power consumption. The Physical layer of this standard supports bands between 868/915 MHz and 2.4 GHz. IEEE 802.15.4 is designed for short-range communication applications that require low transmission power. In these applications, maximizing the residual power of sensors is the main challenge.

Zigbee operates on top of IEEE 802.15.4 [18]. This standard supports networks with a large number of sensors (i.e., up to 65k nodes). Sensors can monitor the environments for years thanks to low cost and low power features provided by Zigbee standard.

6LoWPAN (IPv6-based Low power Wireless Personal Area Networks) enables IPv6 over IEEE 802.15.4 [19]. In this standard, low power sensors can communicate with IPv6 speaking devices. An adaptation layer accommodates IPv6 packets into IEEE 802.15.4 frames. 6LoWPAN is mostly leveraged in embedded devices which are used in home and building automation or health-care automation [19].

ISA100.11a is designed to support low rate wireless communications for automation and monitoring applications [16, 20]. It defines the open systems interconnection (OSI) layers specification for wireless sensors. The main design goals of this standard are scalability, low energy consumption, and the capability to interact with other devices. The physical layer operates in the 2.4 GHz band. ISA100.11a provides a simple but strong security mechanism for data protection.

2.4. *Research Challenges*

As discussed, sensor devices suffer from many resource constraints such as low power transmission and low battery power. These devices are mostly used for tracking and monitoring applications [16] such temperature, noise, etc. Therefore, a variety of hardware platforms are needed to fulfill the monitoring and tracking goals. Here, we focus on research challenges that are performed on improving the nodes' efficiency in tracking and monitoring applications [21].

3. SDN Basics

In this section, we briefly overview the concept of SDN and OpenFlow which is the most widely used for SDN in wireline networks.

3.1. Concept of SDN

SDN separates forwarding and control plane in communication networks. That means, forwarding nodes do not communicate with each other to populate their forwarding tables like in traditional networks, but a controller configures their forwarding tables. The Open Networking Foundation (ONF) [22] defines a three-level architecture for SDN which is illustrated in Fig. 3. It consists of a *data* plane, a *control* plane, and an *application* plane.

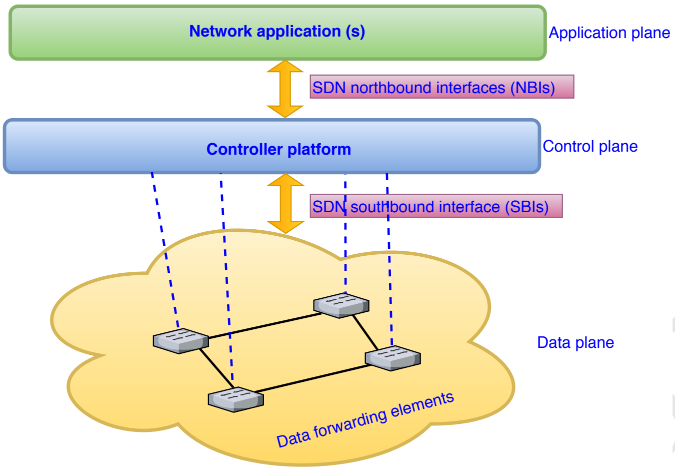


Figure 3: SDN architecture according to [22].

i) **Data plane**: The data plane is the bottom part of the SDN architecture. It comprises a set of forwarding nodes such as switches, routers, access points, etc., which are often called forwarding elements or datapaths.

ii) **Control plane**: The control plane includes a set of controllers which control the datapaths through a so-called southbound interface (SBI) whose traffic is usually carried over a secured connection, e.g., over Transport Layer Security (TLS). The most-widely utilized SBI is OpenFlow. The controllers have an overview of the network, compute suitable forwarding behavior of all datapaths, and configure them with appropriate forwarding rules. Moreover, controllers can obtain information from forwarding elements, they may be triggered by so-called network applications, and in case of multiple controllers, they may communicate with each other.

iii) **Application plane**: The application plane comprises a set of network applications that are input to the controllers to install appropriate rules on the datapaths. Examples of network applications are routing, firewalling, load balancing, network address translation, etc. Thus, the application plane defines policies which are translated by controllers into specific southbound instructions to control the forwarding behavior of network devices. Logically, the application plane communicates with the control plane over a northbound interface (NBI), but often the application plane consists of subroutines within a controller.

3.2. OpenFlow

OpenFlow [23, 24] is a SBI for SDN which has been developed at Stanford University [25]. Forwarding elements have flow tables that can hold mostly a moderate number of flow rules (aka flow entries) which are used for packet handling because they are mostly implemented with fast and expensive ternary content-addressable memory (TCAM). They consist of match fields, counters, and actions [26]. The match fields can refer to selected packet header fields like source/destination MAC/IP address and port, etc., i.e., the match fields extend over several protocols. Counters may be used to gather management information that can be leveraged by the controller. Examples for actions are forward, drop, modify, send to controller, etc. When a forwarding element receives a packet, it may be matched by a flow rule in the flow table. In that case, the specified counters and actions are applied to the packet.

The flow rules are installed by controllers on the forwarding elements. If no flow rule matches the header of an incoming packet (table miss), the behavior of the datapath depends on configuration. It may either drop the packet or send a packet digest to the controller to request the installation of another flow entry. The controller then computes new flow entries respecting the policies provided by the application plane and installs them on the requesting datapath and possibly also on others.

Forwarding rules can be installed either in a *proactive* or *reactive* manner [27, 28]. In *proactive* mode, sufficient rules are installed a priori such that tables misses cannot occur. Such rules are usually coarse-grained, i.e., their match fields describe large traffic aggregates. In *reactive* mode, no or only a few rules are provided a priori. Therefore, the datapaths are configured to inform the controller in case of a table-miss so that it can calculate and install appropriate flow entries. Such rules are usually fine-grained, i.e., they pertain only to the packets of a single flow, i.e., to packets with identical source/destination IP address/port combination.

With *proactive* mode, table misses cannot occur so that packets can be immediately handled. However, if fine-grained rules are needed, not all of them may be known in advance and their number may be too large for the flow tables. The *reactive* mode is more dynamic and flexible in the sense that flow tables hold only the flow entries currently needed. Some use cases like routing can be well supported with *proactive* mode. Others, like NAT or firewalling can be supported only with *reactive* mode. Beyond that, *proactive* and *reactive* mode can be combined, i.e., some rules may be installed for aggregate flows a priori and some other rules are installed only on demand.

4. SD-WSN

In this section, we give an overview of SD-WSN. We first describe the general architecture of SD-WSN and explain its differences to non-software-defined WSNs. Then, we compare of SD-WSN and wireline SDN and finally we give an overview of software tools of SD-WSNs.

4.1. Architecture of SD-WSNs

Fig. 4 shows the general architecture of SD-WSNs. The architecture consists of the following logical planes: i) data plane, ii) control plane, and iii) application plane. The data plane of SD-WSNs includes a set of sensor nodes which sense and forward data in the network. The control plane includes the controller which controls the whole network. The application plane of SD-WSNs includes diverse applications of WSNs such as routing.

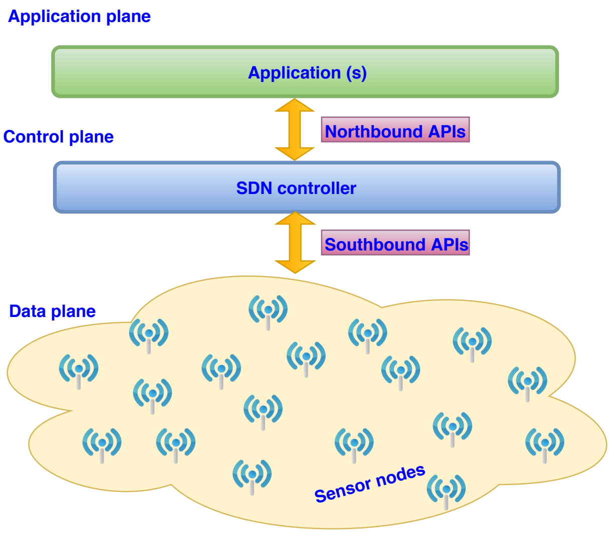


Figure 4: The General architecture of SD-WSNs.

Flow-Sensor [29] tries to leverage OpenFlow features in WSNs. It provides the separation of control plane and data plane in WSNs. In Flow-Sensor, the communication between the controller and BS are based on OpenFlow. *Flow-sensor* leverages TCP/IP for the communications of BS and sensor nodes in the data plane.

The first SDN-enabled architecture for WSNs was proposed in [30]. The authors introduced *Sensor OpenFlow* (SOF) as the communication protocol between data plane and control plane. In this architecture, each sensor node is a *flow-based* packet forwarding element and the controller is the intelligent part for decision making. Each node can communicate with the controller via SOF and the controller is programmable via APIs. SOF supports both IP-based and non-IP based communication between the controller and the nodes.

The main idea behind SOF is to program the data plane of WSNs through APIs. *SOF* makes the non-SDN based WSNs become: i) versatile by supporting more customizable applications for the deployed nodes, ii) flexible by providing a centralized controller which has a direct control on the entire network, and iii) manageable by using suitable open APIs without the need to hack existing code.

We can distinguish control approaches of SD-WSNs into two different categories:

i) **Directly connected controller.** In this category, the controller directly communicates with all sensor nodes. The controller requires a separate channel for control traffic. This is called out-of-band control in SDN [28].

ii) **Indirectly connected controller.** The controller communicates with sensor nodes over other sensor nodes i.e.,using multi-hop communications. The controller sends the control traffic like data traffic over the core network infrastructure, which is also called in-band control in SDN [28].

4.2. Difference to non-SDN based WSNs

In non-SDN based WSNs, to obtain the topology of the network, topology discovery mechanisms are required. They rely on broadcast messages which periodically are sent by each node within its transmission range to identify the neighbors. This operation adds a significant overhead to the network and it also consumes a lot of energy. After obtaining the network topology, several decisions can be made for the network, e.g., routing decisions to steer the network traffic. To perform these decisions each node needs to store routing tables within its limited memory and computes the path for other nodes.

In SD-WSN, many resource-hungry tasks are moved to the controller because it has a power supply and a global view of the network. We give examples. In SD-WSNs, the nodes do not need to send broadcast message periodically for topology discovery. The routing decisions are taken by the controller in SD-WSN. Therefore, the nodes do not require to store the routing information within their routing tables. Furthermore, the controller can also tune the transmission range of each node to reduce the communication interference among nodes. Performing these tasks by the controller in SD-WSNs can save the residual energy of the nodes.

4.3. Comparison of SD-WSN and Wireline SDN

Applying SDN to WSNs introduces a number of new research challenges which make them different from wireline networks. In this section, we give an overview of these new challenges.

Network management in WSNs is different from traditional networks. In traditional networks, the main goal is to minimize the response time while in WSNs the main goal is to minimize the energy consumption.

A WSN has a highly dynamic structure and failures are common. They can occur at any time, e.g., failures due to insufficient residual energy of nodes or communication failures due to environmental obstacles [31]. Therefore, SD-WSNs inherit the same features. For example, in the presence of a failure it may take some time to inform the controller by multi-hop communications. In contrast to SD-WSNs, the network structure is stable in wireline SDN networks.

Furthermore, wireless error-prone channels in WSNs can lead to frequent packet-transmission errors and link disconnections [32]. Therefore, any SDN solution should deal with controlling and monitoring the nodes' communication links to control the network.

4.4. Standardization Efforts

The standards of SD-WSNs should define the set of functions and protocols for sensors and controllers. The authors of [33] used the same standard of WSNs, i.e., IEEE 802.15.4, to build sensor nodes that can be leveraged in an SD-WSN. This standard is not confirmed by any standardization community. Indeed, there is no formal standard for SD-WSNs, yet [3]. The standardization organizations of SDN such as ONF [22] and WSN such as IEEE 802.15.4 [17] should co-operate to define the standards for SD-WSNs.

4.5. Software Tools

In this section, we give an overview of software tools for SD-WSNs. We concentrate on open-source tools which are freely available and can be exploited.

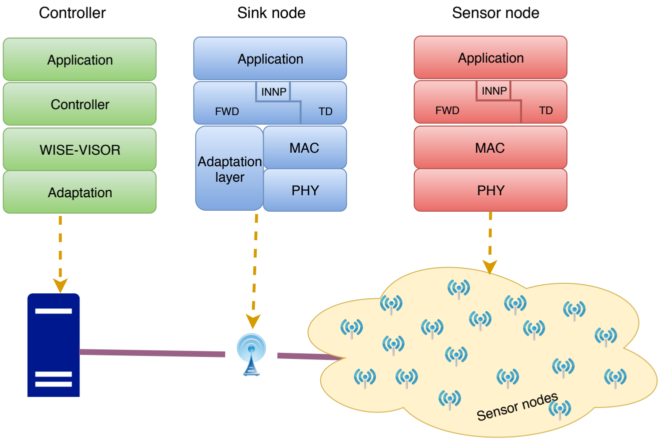


Figure 5: SDN-WISE architecture and protocol stack.

4.5.1. SDN-WISE

SDN-WISE [33] is developed at the University of Catania, Italy. It offers a software framework for SD-WSNs and a prototype hardware for SD-WSNs. SDN-WISE has two main objectives: i) reducing the the amount of exchanged information between nodes compared to non-SDN based WSNs and ii) making the sensor nodes programmable. We overview the architecture, flow table structure, the software simulation tool, and features of SDN-WISE.

4.5.1.1. Architecture. The SDN-WISE architecture has three different components: sensor node, sink node, and controller. Fig. 5 illustrates the general architecture of SDN-WISE and the protocol stack of each component. We describe each of them in the following.

Each sensor node has the following layers in its protocol stack: i) Application, ii) In-Network Packet Processing (INPP), Forwarding, and Topology Discovery (TD), iii) Media Access Control (MAC), and iv) Physical. INPP is responsible for data aggregation or in-network processing operations. TD can gather the local information of nodes in the network and controls their behavior.The Forwarding layer includes an IEEE 802.15.4 transceiver and a micro-control unit (MCU) which manages all incoming packets.

Sink nodes have a similar protocol stack as sensor nodes. The only difference between a sensor node and a sink node is the Adaptation layer. This layer formats the received messages from the sensor nodes in such a way that they can be handled by the controller. Other layers, such as topology discovery, forwarding, application, etc., of a sink node are exactly the same as a sensor node.

The controller has the following layers in its protocol stack. i) Application, ii) Controller, iii) WISE-Visor, and iv) Adaption. The Adaptation layer of controller has the similar functionality of same layer in sink node. The WISE-Visor contains a topology management (TM) layer which provides an abstraction for network resources. The controller layer defines the network policies which have to be implemented by sensor nodes.

4.5.1.2. Flow Table. Tab. 1 shows an example of the WISE flow table of SDN-WISE. A WISE flow table consists of *matching rules*, *actions*, and *statistics*. The matching rule includes the following fields: i) **Opt** determines the operation that should be performed on the *Value* field of the packet. ii) **Size** shows the size of the string in the packet. iii) **S** indicates the state of the packet. If S=0, the matching rule is not applied for this packet. iv) **Addr** determines the source address of the packet. v) **Value** shows the assigned value to the packet. The action consists of the following fields: i) **Type** specifies the type of action, e.g., forward, drop, etc. ii) **M** is a flag that determines whether the action is exclusive (M=0) or not (M=1). For M=0, after executing the corresponding action to that packet, the other actions of the WISE flow table are ignored for execution, even if the matching rules are satisfied. Otherwise, after the execution of the corresponding action, other actions in the WISE flow table will be executed if the matching rules are satisfied. iii) **S** indicates whether the action must be executed on the packet. iv) **Addr** determines the destination address of the packet. v) **Value** shows the assigned value to a packet.

The statistics section of the WISE flow table consists of *TTL* and *counter* fields. TTL determines the time to live for the flow and the counter shows the number of packets that have been matched for the corresponding matching rule.

4.5.1.3. Software Simulation Tool. SDN-WISE offers functionalities similar to Mininet [34]. Mininet is a widely used network simulator to perform campus-size network experiments. It uses Cooja [35], which is a network simulator for Contiki OS, which is the operating system for low-power wireless Internet of Things [36], to create the network. Fig. 6 shows a running example of an SD-WSN with 17 nodes in SDN-WISE, which is randomly deployed in a two-dimension network area. Node 1 is the sink node in this figure.

Table 1: WISE flow table in SDN-WISE [33].

Matching rule					Matching rule					Matching rule					Action					Statistics	
Opt	Size	S	Addr.	Value	Opt	Size	S	Addr.	Value	Opt	Size	S	Addr.	Value	Type	M	S	Addr.	Value	TTL	Counter
=	2	0	2	B	>	2	0	10	x_{THR}	=	1	1	0	0	Modify	1	1	0	1	122	23
=	2	0	2	B	≤	2	0	10	x_{THR}	=	1	1	0	1	Modify	1	1	0	1	122	120
=	2	0	2	B	-	0	-	-	-	-	0	-	-	-	Forward	0	0	0	D	122	143
=	2	0	2	A	=	1	1	0	0	-	0	-	-	-	Drop	0	0	-	-	100	42
=	2	0	2	A	=	1	1	0	1	-	0	-	-	-	Forward	0	0	0	D	100	43

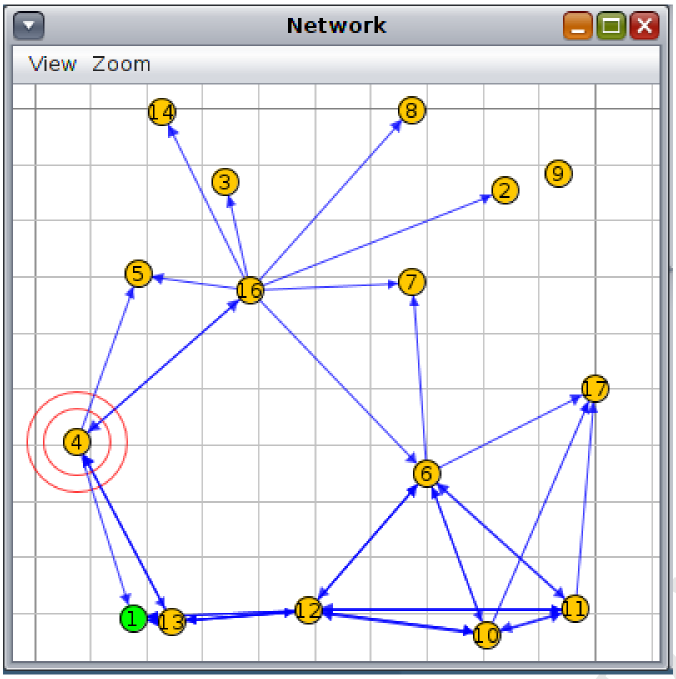


Figure 6: A sample network in SDN-WISE.

SDN-WISE defines an open-source controller which performs the routing decisions among the deployed nodes based on Dijkstra's algorithm. The nodes collaborate with the controller through sink node.

4.5.1.4. SDN-WISE Features. SDN-WISE supports duty cycle, i.e., the possibility of periodically turning off the radio interface of each node and its data aggregation. SDN-WISE handles the packets based on the information in its payload and its header section.

4.5.2. Tiny-SDN

Tiny-SDN is a TinyOS-based SDN framework for WSNs [37]. In this section, we give an overview on architecture, flow specifications, and features of the Tiny-SDN in more detail.

4.5.2.1. Architecture. The Tiny-SDN architecture has two types of components: *SND-enabled sensor node* which has the functionality of a sensor node as well as an SDN-switch and *SDN-controller* which is in charge of managing control plane operations such as routing decisions. Their structure is depicted in Fig. 7. We describe them in the following.

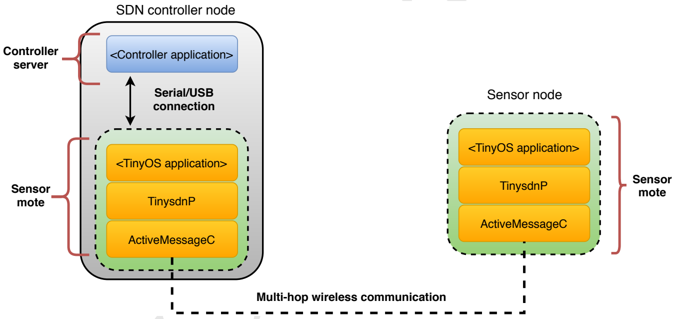


Figure 7: Layers of Tiny-SDN components.

Each SDN-enabled sensor node in the TinySDN architecture has three main components: i) **TinyOS Application.** A component that acts as an SDN device in the network and has the responsibility of generating packets and putting them on the network. ii) **TinysdnP.** A component which checks the flows based on the match fields and performs suitable actions to that match. iii) **ActiveMessageC.** A component that programs and manages the interfaces to handle radio module of the nodes. All tasks corresponding to the wireless communication channels are performed by this component.

Table 2: Data flow table of TinySDN.

FlowID	Action	Value	Counter
1	Drop	N/A	100
5	Forward	5	20
10	Forward	10	50

Table 3: Control flow table of TinySDN.

Destination Node ID	Action	Value	Counter
0	Forward	4	5
1	Forward	4	4
7	Forward	6	4

The SDN controller node performs traffic flow management. It has two main components: **Sensor mote module** which is responsible for communicating with other sensor motes using *ActiveMessageC*. Each sensor mote module was shown as an instance of a sensor node in Fig. 7, and **Controller server module** which hosts the controller application and manages the network flows and the topology information.

Table 4: Comparison of SDN-WISE and TinySDN.

Feature	SDN-WISE	TinySDN
Node types	Controller, Sink, Sensor node	Controller, Sensor Node
Wireless Channel	IEEE 802.15.4	IEEE 802.15.4
Node deployment	Manual, Random, Ellipse, Linear	Manual, Random, Ellipse, Linear
Mote type	EMB-Z2530PA	TelosB mote
Programming Language	Java	nesC
Software Simulator	Cooja	Cooja
Network Heterogeneity	Yes	No
Supported Actions	”Forward to”,”Drop”,”Modify”,”Send to INPP”,”Turn off radio”	”Drop”,”Forward”

4.5.2.2. Flows and Actions Specifications. SDN-enabled sensor nodes support two actions: *drop* and *forward*. Two types of flows are also supported by each end-device. First, *Data flows* which are used for applications data traffic. Tab. 2 shows a data flow example of TinySDN. Second, *control flows* which are used to control the traffic between the SDN-enabled sensor node and the SDN-controller. Tab. 3 illustrates an example of control flow table in TinySDN.

4.5.2.3. Tiny-SDN Features. Tiny-SDN enables the implementation of multiple controllers for a network. It focuses on in-band control traffic of WSNs. To decrease the latency of the network, Tiny-SDN supports using multiple controllers in the network.

4.5.3. Comparison of SDN-WISE and Tiny-SDN

We compare SDN-WISE and TinySDN in Tab. 4. SDN-WISE offers three types of nodes in the architecture while TinySDN has two types of nodes. Both software tools used the same wireless channels and the deployment scenarios. Cooja [35] is the common software simulator for both systems. Cooja provides the same node deployment mechanisms for both systems in a given network area. Each node in an SDN-WISE emulated network can communicate with a virtual network of OpenFlow switches which are controlled by ONOS [38]. This feature enables SDN-WISE to control heterogeneous networks, i.e., the network consists of sensor nodes and the network instances of Mininet. SDN-WISE supports more matching fields and actions than TinySDN. Examples for matching fields are addr, value, and S. Examples for actions are forward, drop, and modify. TinySDN provides the opportunity for deploying several controllers in the WSNs. TinySDN supports two actions in the flow tables. Examples are drop and forward.

5. Advances in WSN through SDN

In this section, we overview SDN-based approaches for WSNs and classify the research literature in several categories. Fig. 8 depicts the organization of the reviewed works in this section.

5.1. Energy Efficiency

Energy-efficiency is one the most critical aspects of WSNs and it is the objective of many WSNs research works. Sleep scheduling approaches can be leveraged to switch the nodes into idle state if their functionality is not required. These algorithms can be used to reach the networks’ goal.For instance, the authors of [39, 40, 41] used sleep scheduling approach to extend the network lifetime while keeping the connectivity of nodes and preserving the coverage requirements. Here, we classify the energy-efficient works into three areas: *lifetime*, *coverage control*, and *clustering*.

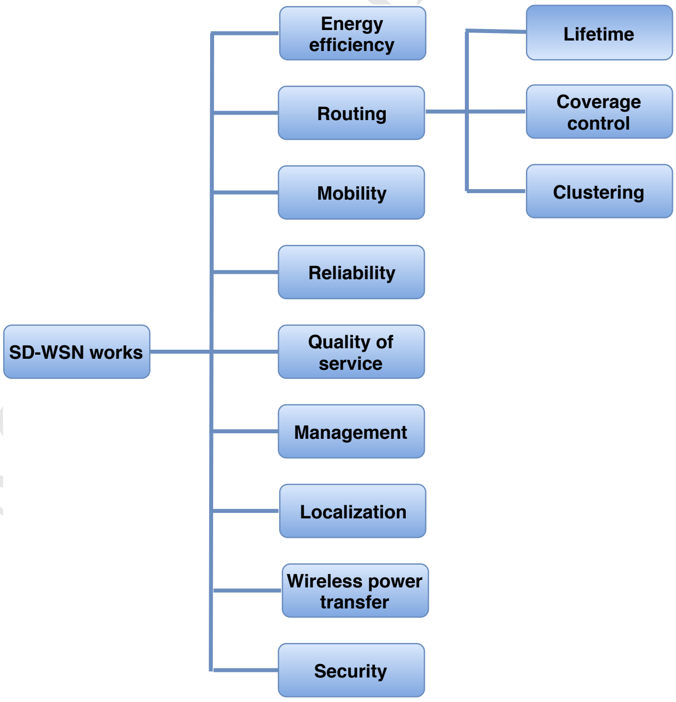


Figure 8: Categorization of SD-WSN works.

5.1.1. Lifetime

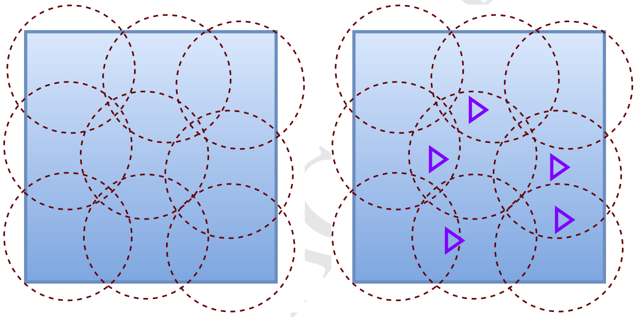
Prolonging the network lifetime gives the possibility to utilize the nodes functionalities for a longer period of time [42, 43]. For example, computational tasks like path selection and neighbor discovery consume most energy in WSNs. The energy consumption to send a single bit of data by a sensor in a WSN, e.g., composed of MICA motes [44], is at least 480 times higher than performing one additional 32-bit instruction by CPU [45]. The authors of [45] stated that data transmission consumes approximately 80% of nodes’ power.

Energy efficiency in SD-WSNs is investigated in [46, 47, 48]. An SDN-based method to utilize the energy of nodes in WSNs is proposed in [48]. It also maintains the connectivity of nodes [48]. In the proposed architecture for the sleep scheduling, all nodes are connected to a switch via suitable links and the switch is connected to the SDN controller. Consequently, each node in the network can have two types of connections: first, it can have a connection with other nodes. Second, it has a connection to interact with controller. In this case, the computation tasks are just moved from nodes to the controller. After making a decision by a controller for each deployed node, the rule can be installed on the nodes via a switch in the network.

A fuzzy logic based algorithm to improve the lifetime of SD-WSNs was proposed in [47]. It controls the network topology to prolong the network lifetime.

5.1.2. Coverage Control

Coverage [49, 50] is one of the widely used applications of WSNs in which a network area or a set of targets should be covered by the sensor nodes in the network [12, 51, 52, 53, 54]. Coverage control activates or deactivates the sensor nodes to cover a network region. Network coverage can be categorized into: *target*, *area*, and *barrier* coverage. The goal of *target* coverage is to cover a set of stationary or moving targets while in the *area* coverage the goal is to monitor the whole network area. Fig. 9 shows two different coverage problems in WSNs. Each dashed circle shows the sensing range of a sensor and each triangle indicates a fixed-position target in this figure. The network area is depicted as a rectangle in Fig. 9. For example, Fig. 9a illustrates a network that the deployed nodes were exploited to monitor the whole network area while Fig. 9b shows a network so that the sensor nodes should monitor a set of targets. One common approach to the area or target coverage is to use a subset of nodes to monitor the network coverage requirements. This technique is also known as cover-set approach [55].



(a) Area coverage. (b) Target coverage.

Figure 9: Coverage control examples

Furthermore, in some application scenarios covering the entire network area is not necessary and it is enough just to partially monitor the network area. This is known as *partial* coverage or *p-percent* coverage. Leveraging node deployment mechanisms can improve the energy efficiency of nodes in partial coverage [56, 41].

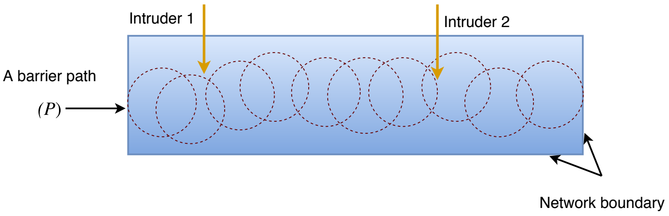


Figure 10: An example of barrier coverage with two intruders and a barrier path ($\mathcal{P}$) which can detect any penetration to the network.

Preserving the network barriers from intruders is the goal of *barrier* coverage [51]. The selected nodes in the barrier coverage should guarantee the network area from penetration. Border surveillance is the common application for barrier coverage of WSNs [51]. Fig. 10 illustrates a network with enough to guarantee the barrier coverage requirements. Two intruders aim at entering the network from the top (north) to bottom (south) of the network.

Several SDN-based works for the coverage problems of WSNs can be found in [52, 57, 53]. The target coverage in SD-WSN was studied in [57]. The authors studied three SDN-based solutions for scheduling sensor nodes to monitor the targets in such a way that the total energy consumptions of the nodes are minimized. In this work, the SDN controller is in charge of selecting active nodes to monitor the deployed targets. In this scenario, the authors assume that the targets are stationary. Tab. 5 classifies SD-WSN Works on coverage control.

Table 5: Coverage control mechanisms in SD-WSNs

Techniques	**Coverage type**		
	Target	Area	Barrier
[52]			✓
[58]			✓
[57]	✓		
[53]		✓	
[48]		✓	

5.1.3. Clustering

Clustering [59, 60, 61] is widely used in WSNs for controlling the energy consumption of nodes and for routing. Clustering puts the nodes into clusters and there is a head node for each. Cluster heads (CHs) are in charge of collecting data from the nodes in their clusters and sending them to the BS while non-CH nodes are responsible for gathering the network information and forwarding it to the CHs [62]. The idea is to select the most powerful node as a CH to transfer the network data to sink node. Therefore, selecting suitable CHs is a challenging issue which was considered by researchers. Fig. 11 shows an example of a clustered network with three clusters. Each cluster member is connected to the sink node through its CH.

Clustering in SD-WSN with the aim of reaching energy efficiency was studied in [63, 64]. In this work, the SDN controller collects information of the network topology via Link Layer Discovery Protocol (LLDP) and installs suitable rules to gather

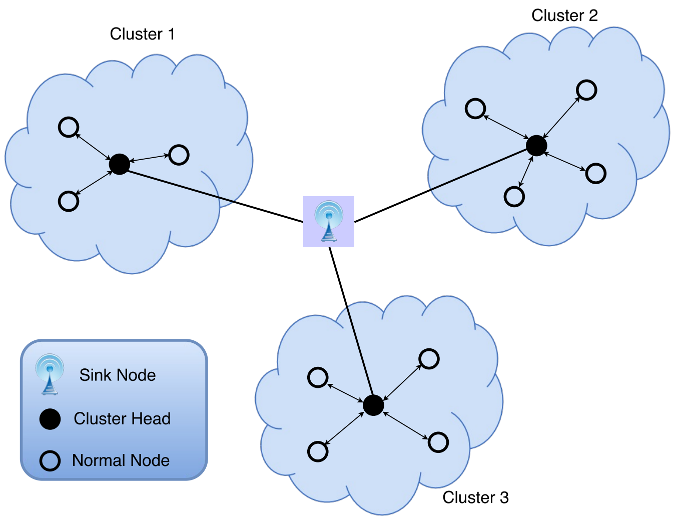


Figure 11: A clustered WSN with three clusters.

the statistics of the nodes. The SDN controller is co-located in CH in the proposed architecture to take the control of all nodes in the cluster. The SDN controller can install a suitable rule on each flow table of the nodes to gather the information and send them via the controller to sink node. There are more than one controller and they can interact with each other to meet the global goal of the network.

An SDN-based clustering approach to minimize the energy consumption of the nodes was proposed in [65]. The SDN controller divides the nodes into several clusters based on residual energy and the number of neighbor nodes. To balance the communication costs, it makes a routing tree among the clusters to steer the network traffic.

5.2. Routing

There are many routing protocols for WSNs. The works in [66, 67] provide a survey on routing challenges and design issues in WSNs. Transferring the network data efficiently is one of the main critical challenges in WSNs. Objectives pursued by routing protocols are: congestion control, delay minimization, throughput maximization, etc. The routing can be performed packet or flow-based [67].

In SD-WSN, routing requirements can be different from the non-SDN based WSNs because the nodes do not participate in path selection. The controller is in charge of that task, which alleviates the task of sensor nodes. Forwarding nodes may be chosen such that least energy is consumed and residual energy of all nodes is balanced. Several routing protocols for SD-WSNs are reported in [68, 58, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78]. Tab. 6 shows the routing protocols and their considered metrics in SD-WSNs.

5.3. Mobility

Mobility in WSNs can be classified into weak and strong mobility [79]. Weak mobility results from changes of the network topology. For example, nodes in WSNs are prone to failure such as hardware or battery problems. Therefore, they have to be replaced by new nodes. Strong mobility results in physically moving the place of nodes. This movement can be due to external forces, i.e., wind or water, or intrinsic characteristic of the nodes. Robomote [80] is an example of mobile sensor mote which is equipped with a wheel to move around. Consider a routing scenario, which nodes are responsible to transfer the data toward the sink node. The nodes close to the sink node deplete their energy for data communications [81]. The network becomes disconnected. Therefore, mobility can help the network to replace energy-drained nodes.

An SDN-based mobility solution was proposed in [52] for mobile nodes have mobility characteristic. In this work, the barrier coverage for a dynamic zone is considered and nodes can move within entire the network. A movement strategy controller controls the nodes' movement. To meet the barrier coverage requirement, the controller determines the new locations for the nodes such that activated nodes can detect any intrusion into the network.

5.4. Reliability

Reliability of WSNs includes the reliability of several components such as node and link. For example, the reliability of a node includes the reliability of battery, radio, hardware, middleware, operating system, and application [82]. In WSNs, the monitored data is transferred to the outside of the network via multi-hop connections. Any failure in the network causes energy consumption due to sending traffic through non-energy efficient paths [83, 82]. For instance, the objective of reliable routing algorithms is to maximize the packet delivery ratio.

The authors of [84] studied the reliability of nodes in SD-WSN using continuous-time Markov chains (CTMCs). In an SD-WSN either controller or node can fail. For example, if the network uses a unique controller, it becomes a single point of failure and the WSN is not reliable anymore. The proposed approach suggests using an extra controller to improve the reliability of the entire network. In this case, if a controller fails to act properly, a spare controller can be replaced to keep the desired reliability in the controller layer of SD-WSN. Each sensor node in this algorithm has a specific failure probability and the system fails if all sensors fail. They also suggested a lower bound on the number of failed sensors to detect the complete system failure.

The Reliability of industrial sensor nodes in SD-WSNs was studied in [85]. The proposed architecture takes several aspects such as heterogeneity, coverage, failure, and reliability into account to extend the overall energy efficiency of the network. The SDN controller balances the energy consumption by choosing the suitable nodes.

5.5. Quality of Service (QoS)

QoS provisioning deals with challenges that offer a guaranteed level of service delivery to a network [86]. QoS requirements can be specified into congestion, packet loss, bandwidth, and jitter. Providing QoS is different among applications because various requirements such as loss and delay could be

Table 6: Considered Metrics by different routing protocols in SD-WSNs

Techniques	**Metric**					
	Lifetime	Congestion	Delay	Reliability	Scalability	Throughput
[68]	✓	✓				
[58]			✓	✓		
[69]	✓					
[70]	✓				✓	
[71]	✓					
[72]	✓		✓			✓
[73]	✓					
[74]	✓					
[75]	✓		✓			✓
[76]	✓					✓
[77]	✓					
[78]	✓					

planned for a specific application. For example, real-time applications are sensitive to delay rather than loss, while for other applications like target tracking reliable and timely delivery data is important [87]. QoS provisioning can be performed per-packet or per-flow.

The QoS in carrying data traffic by the nodes of SD-WSNs was studied in [88]. It exploits per-packet *state* information, which is supported by SDN-WISE [33] to provide several levels of QoS. Each node stores the received packets in its buffer. There is a threshold on the size of the buffer in each node and by reaching the number of packets to the threshold value, the state of each node changes. Then, a priority is assigned to incoming packets to the buffer to classify them into different levels. The controller can provide a set of forwarding rules to each node based on traffic priority levels.

An SDN-based algorithm for QoS provisioning in SD-WSNs in the presence of congestion was studied in [89]. The authors used hop count and local traffic information in the network controller to distribute the traffic in the network. The authors claimed that by using SDN for congestion control in SD-WSNs, they reduce up to 46% packet loss. The core part of the devised algorithm relies on a traffic monitoring algorithm which notifies the occurrence of a congestion by sending an alarm packet to the controller. The controller creates flow rules for the congested node, the source node, and the appropriate forwarders to avoid further congestion.

The authors of [90] proposed an SDN-based solution to provide end-to-end QoS by considering packet loss and bandwidth over 6LoWPAN-based WSNs. It leverages IPv6 flow label for a QoS tag in 6LoWPAN. This label is kept unchanged in transforming 6LoWPAN to IPv6 format. Tab. 7 summarizes the QoS works in SD-WSNs.

Table 7: Quality of service works in SD-WSNs.

Techniques	**QoS Metrics**		
	Congestion	Packet loss	Bandwidth
[89]	✓	✓	
[33]	✓		
[90]		✓	✓

5.6. Management

Network management in WSNs is a challenging process including network configuration, provisioning, and maintenance [91]. Managing a network with different nodes from different vendors requires a complex management process. The management mechanisms allow the network administrators to manage vendor-specific nodes in WSNs.

Smart [91] is an SDN-based network management solution for WSNs, which offers a layered approach by co-locating the controller on the BS. Fig. 12 depicts the architecture of BS in smart. Smart has five layers in the protocol stack, namely: Physical, Medium Access Control (MAC), Network Operating System (NOS), Middleware, and Application layer. In this architecture, the Middleware, which co-located in the BS, is in charge of defining flow tables from the network applications, e.g., routing.

The Middleware layer has the following components: controller, flow table definition, mapping function, and mapping information. The mapping function creates a network map based on the received table from the neighbor sensors and can be directly invoked from the controller if needed. The network mapping information, e.g., energy consumption, response time, link quality, is stored in a database and can be invoked at any time.

The application layer can define specific functionality to each node, e.g., temperature monitoring, and contains a location component which is also denoted as Localization and Tracking Algorithms (LTA). The Application layer interacts with controller and mapping information components. LTA is in charge of providing a node's location information by processing mapping information. The controller can take more accurate information of the nodes' position through the application layer to manage the network. The authors claimed that Smart [91] can provide energy-efficiency, mobility management, and localization.

The authors of [92] proposed a distributed control system to

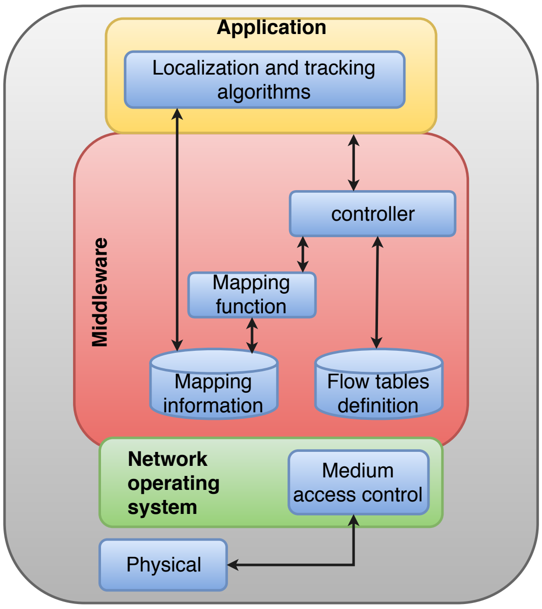


Figure 12: Base station architecture in Smart [91].

manage the nodes in SD-WNSs. To distribute the controllers in the network, a fragmentation mechanism is leveraged which aims at placing the controllers close to the network devices to improve the energy efficiency of the network. A radio resource allocation mechanism in SD-WSNs is proposed in [93]. The controller of SD-WNS dynamically assigns the suitable radio resource to each node. The authors modeled the problem as an optimization problem with QoS constraints to minimize the energy consumption of the nodes.

5.7. Localization

Location information of each node is necessary for many applications of WSNs. Typically, the nodes are randomly scattered in the network zone.Localization techniques aim at positioning each node [94]. Global Positioning System (GPS) is widely leveraged for this purpose, but it requires more energy to run and it is not easy to install this system on board of each node [95].

The authors in [96] modeled the localization problem in SD-WSNs based on 0-1 programming problem and proposed an SDN-based localization algorithm to select the nodes in localization mechanism. There are two types of nodes in this architecture which are called agent and anchor nodes. The agent nodes, with their exact location, were exploited to find the location of anchor nodes. The SDN controller interacts with agent nodes in the localization process.

An anchor-based scheduling algorithm for positioning the nodes in heterogeneous SD-WSNs was proposed in [97]. The SDN controller determines the position of each anchor node based on the network power constraints. Fig. 13 shows a sample architecture for localization in SD-WSNs. The SDN controller interacts with agent nodes through Sensor-OpenFlow [30] in localization process.

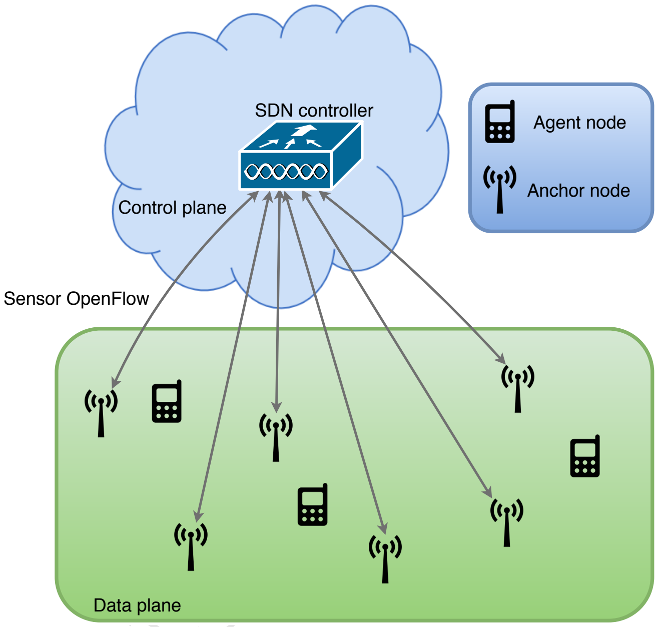


Figure 13: Node localization example in SD-WSNs

5.8. Security

Security is one of the critical challenges in WSNs. The authors of [98, 99] surveyed the security challenges of WSNs. The deployment of SDN in WSNs overcomes some security challenges [100, 101] because the sensor nodes perform only data forwarding toward the controller. Securing a network requires large number of computational operations. Performing intensive security operations with energy-constrained sensor nodes can deplete the residual energy of nodes. Therefore, these resource-hungry operations in SD-WSNs can be performed by the controller. Moreover, the controller in SD-WSNs has a global overview of devices status in the network which results in identifying the malicious user and their activities [101].

Applying SDN to WSNs introduces new security problems. For instance, errors in the network configuration can lead to security threats. The control plane provides an abstraction for the forwarding elements of the data plane, which is prone to denial-of-service (DoS) and distributed DOS (DDoS) attacks [102]. The nodes can be exploited by attackers as a gateway to enter the network [3]. Furthermore, the controller can be a single point of failure for the whole network, if the attacker compromises the controller.

Cryptography mechanisms can be leveraged for the security of SD-WNSs, but the main challenge is how to distribute the key in the network. The key distribution can lead to high communication overhead. The authors of [103] proposed a key distribution method based on physical unclonable functions (PUFs) for SD-WSNs to minimize the communication overhead.

The authors of [104, 102] classified the main threats on SDN-based networks as follows.

i) **Traffic flow attacks** can be performed on forwarding elements and controllers. The malicious user launches

DoS attacks to devastate the resource of network devices. This threat can be mitigated by authentication mechanisms [100].

ii) **Forwarding device attacks** could be used on each forwarding element to drop, slow down, or discard the network traffic. This attack can be also exploited to inject traffic to overload the controller.

iii) **Control plane communication attacks** can be performed as DoS attack for data theft in the network. Leveraging common secure communication protocols such as TLS or secure sockets layer (SSL) are not enough to avoid those attacks [105] because there are several man-in-the-middle attacks for the TLS/SSL model.

iv) **Controller attacks** compromises the controller to obtain the control of entire network. Using intrusion detection systems is not enough due to the difficulty in finding the exact combinations of events to construct this attack.

v) **Lack of trust between applications and the controller** is similar to control plane communication attacks because a trusted communication between network applications and the controller cannot easily be established. Certifying the forwarding devices is different than certifying of applications.

vi) **Administration stations attacks.** The devices in administration station are used to access the controller in SDN-based networks. Indeed, using the administration stations to control the network devices are also common in other networks. The difference is that each machine in the administration station can be exploited to program the network from a single point if the attacker compromises the controller.

vii) **Lack of trusted resources for forensics and remediation.** There are resources in a network that can be leveraged for troubleshooting. Such reliable information are necessary to investigate the facts of incidents in the network and without them, it is difficult to find a remedy for a problem. This is not specific to SDN networks.

The goal of this classification is to show that the threats in SDN-based networks are different than in other networks. Tab. 8 shows the security threats in SD-WSNs and their consequences. This table also clarifies whether a threat is specific to SD-WSNs.

5.9. Wireless Power Transfer

In a WSN, a sensor node can undertake several tasks that depletes the energy of a node. If nodes can recharge, wireless power transfer mechanisms may be exploited to replenish the nodes, i.e., a sensor node can transfer its energy to other nodes through an appropriate transmitter [106, 107].

The power transfer problem in SD-WSN was studied in [108] with aiming at real-time recharging of sensor nodes. In this work, the SDN controller is in charge of finding an optimal position for the energy transmitters. Also, it can determine the minimum number of energy transmitters over the course of primary process to prolong the charged energy by each node in the network. Additionally, the controller can fairly distribute the energy among all the nodes by having the workload information of each node. The authors proposed different methods for maximizing the charged energy and fairly distributing the energy among all nodes [108]. For this purpose, they formulated as an optimization problem with several constraints and proposed a solution. The controller is in charge of selecting energy transmitters to balance the energy consumption of the nodes.

5.10. Comparison of SDN-based and non-SDN based WSNs

In this section, we compare SDN-based and non-SDN based works in WSNs. One of the main advantages of exploiting SDN in WSNs is energy saving. As discussed in Sec. 5.1, sending broadcast messages is mandatory for topology discovery. While in the SDN-based WSNs, this process is performed by the controller, which save energy for each node. For instance, in the scenarios like localization and wireless power transfer, the SDN controller can easily locate the best places for the nodes. Tab. 9 summarizes the differences between SDN-WSNs and non SDN-based WSNs.

6. Challenges in SD-WSN

In this section, we discuss open challenges in SD-WSNs.

6.1. Network Operation

In this section, we discuss the network operation challenges that require further investigation in SD-WSNs.

6.1.1. Re-Clustering

In non-SDN based WSNs, cluster heads deplete their energy due to the high number of communications they have with other nodes within the cluster and with other cluster heads to transfer the network data. New cluster heads need to be selected to steer the network traffic. Cluster head nodes in SD-WSNs inherit the same characteristic of WSNs. Therefore, this challenge needs to be considered in SD-WSNs. SD-WSN may be able to achieve faster and better re-clustering which has not yet been studied.

6.1.2. Topology Control

Controlling the network topology can improve energy efficiency of the network. The primary objective of any topology management system is to maintain the network coverage while keeping the nodes connected [109]. Every topology control protocol tries to select a minimum number of nodes to maintain the network topology. Selecting a proper transmission range in a network with heterogeneous transmission range leads in reaching the efficiency goal of the network because by using a lower transmission range the nodes can consume less amount of energy. Nevertheless, none of the above works offers a complete control topology protocol for SD-WSNs.

Table 8: Security of SDN-based WSNs vs. non-SDN based WSNs

Threat number	Specific to SD-WSNs	Consequence in SD-WSNs
i	No	Can be a door for DoS attacks.
ii	No	The impact is potentially augmented.
iii	Yes	The communication with the controller could be explored.
iv	Yes	Having a control on the controller may lead to the control of entire network
v	Yes	Malicious applications can be developed and executed on the controller.
vi	No	The impact is potentially augmented.
vii	No	It is crucial to provide fast recovery and diagnosis on the time of happened faults.

Table 9: SDN-based WSNs vs. non-SDN based WSNs

Metrics	**SDN-based WSNs**	**Non SDN-based WSNs**
QoS	The controller takes care of QoS provisioning for the network	Each node is in charge of provisioning QoS
Routing	The controller decides for the nodes for routing	The nodes collaborate themselves for routing decisions
Energy-Efficiency	The nodes do need to send broadcast messages to the neighbors in order to find them. The controller does this energy consuming process. The controller also determines the active time of each node.	The nodes do need to send broadcast messages to the neighbors in order to find them. Collaboration among the nodes are required to determine the active time of each node.
Security	Introducing the controller opens new security threats for the network beside the common threats of WSNs.	The network has the common security threats.
Mobility	The controller determines the new place to move for each node	Interaction among the nodes are needed to determine the new place to move
Localization and power transfer	The controller determines the place for the nodes.	The nodes should interact with each other for this purpose.
Reliability	The controller and the nodes can fail in sending the traffic	The nodes can fail in steering traffic
Management	The controller manages the whole network	The nodes interact with each other in order to manage the network

6.1.3. Node Mobility

Sensor nodes may intentionally change their positions. That can improve the WSNs capabilities in many aspects such as automatic node deployment, rapid reaction to event changes, and flexible topology management [110, 111]. For instance, for coverage applications mobile node may improve coverage. Due to dynamic network changes and resource limitations such as bandwidth and power limitations, the mobility of the nodes should be carefully controlled by the controller. The mobility feature has not yet widely studied. It is difficult to use, but with SDN's intelligence multiple use cases may be achieved.

6.1.4. Improving Routing

Routing can be improved in SD-WSNs by leveraging the controller which has the global overview of the network and of the devices status. For example, a routing path may have several constraints like reliability. Moreover, other constraints such as bandwidth and delay can be considered. This issue can be modeled as Multi-Constraint Optimal Path (MOCP) problem [112]. Consider a network graph *G=(V,E)* where V indicates a set of sensor nodes and E indicates a set of edges between the sensor nodes. Each link in G , i.e., $(u, v) \in E$, is associated with a cost parameter $c(u, v)$ and n additive QoS parameter $w_k(u, v)$, for $k = 1, 2, \ldots, n$ [113]. Given n constraints, a MOCP problem can be defined as finding a path p from the source to the sink such that:

$$w_k(p) = \Sigma_{(u,v)\in p} \quad w_k(u, v) \leq c(u, v), \quad \text{for } k = 1, 2, \ldots, n \quad (1)$$

and $c(p) = \Sigma_{(u,v)\in p} \quad c(u, v)$ is minimized over all feasible paths satisfying Eq. (1). Thus, this concern should be considered in the future works of SD-WSNs.

6.1.5. Data Traffic Scheduling

Sensor nodes are exploited to gather environment data. After collecting the data from all or some nodes, they should be forwarded to a BS [114]. This can be performed by a collaboration among the nodes in a WSN. In contrast SD-WSNs, the controller performs such task. As discussed in Section 5, data transmission consumes around 80% of node's energy and leveraging a proper scheduling mechanism can save the energy from energy-constrained nodes. In one hand, the available nodes in the network should be scheduled in such a way that the network traffic transferred to the sink node efficiently. On the other hand, nodes with higher residual energies can be an alternative to schedule data traffic. Therefore, this needs investigation in the future SD-WSNs works.

6.1.6. Network Monitoring

Network monitoring checks the functionality of network devices through specialized management tools. It ensures the availability and the performance. WSNs are typically deployed in a complex and distant environment to monitor objects without human interactions [115]. Wireless links are not stable and prone to packet loss. Additionally, nodes can fail during the network operations. Thus, real-time monitoring tools are required to check the operations of the nodes in the network.

A high-level API-based method to monitor SDN-based networks through OpenFlow was proposed in [116]. It uses a statistic based algorithm to collect accurate status. OpenNetMon [116] is a tool that provides an end-to-end QoS monitoring for traffic engineering (TE) in SDN-based networks. Such network monitoring tools are also required for SD-WSNs to check the functionality of SD-WSNs' devices.

6.2. Challenges for Network Applications

Network applications can benefit from SDN in WSNs. We state the research challenges for WSN applications such as coverage and node mobility that require investigation in SD-WSNs.

6.2.1. Coverage

Some of coverage issues in SD-WSNs are currently studied in the literature. However, several aspects of coverage problem in SD-WSNs need further investigation. We overview them in the following.

6.2.1.1. Partial Coverage. The goal of area coverage is to cover the whole network area by the nodes. In partial coverage scenario, monitoring the whole network area is not required while it is enough just to monitor a special percentage of the network area. This problem needs also to be considered in the future works of SD-WSNs.

6.2.1.2. Coverage Holes. Coverage algorithms may lead to having coverage holes [117]. A coverage hole is the amount of the network area that is not covered either by the nodes or the chosen active nodes. Fig. 14 demonstrates a sample network in which the deployed nodes lead to a coverage hole. In this figure, the network area is divided into fixed-size cells, which is one of the common ways to compute the coverage contribution of each node. This is not easy to perform in non-SDN based WSNs because the network area information is required and it should be distributed among the nodes to check. This issue needs further investigations in SD-WSNs.

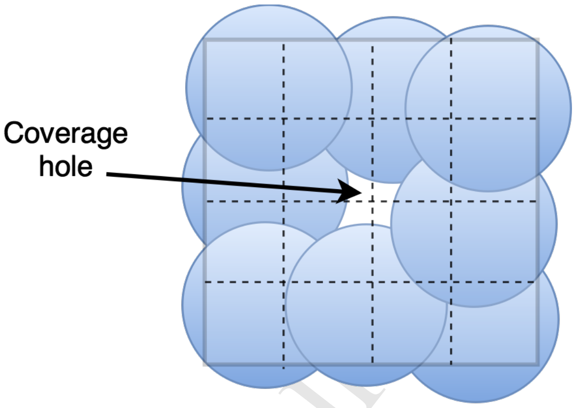


Figure 14: An example of coverage hole.

6.2.2. Leveraging Node Mobility

To improve the nodes' functionality in covering the network region, the nodes' mobility can be leveraged. For example, coverage holes can be covered by moving the nodes toward the coverage hole area. This problem needs investigation in future SD-WSNs works.

6.3. SDN-Specific Challenges

In this section, we describe the challenges that are specific to SDN networks and applying SDN to WSNs inherits the same issues.

6.3.1. Control Plane Resilience

In an SD-WSN, a single controller can be a single point of failure for the network. Multiple controllers can be leveraged to overcome the controllers' failure. The authors studied [84] the controller failure by adding an extra controller, but still, the inter-communication mechanism between controllers are not considered in this scenario. A complete solution is needed to handle controller failures in SD-WSNs.

6.3.2. Data Plane Resilience

In SDN network, the controller is in charge of detecting data plane failures and it the case link or node failures, packets can no longer be forwarded to affected next hops. The controller repairs the path by installing new flow entries in wireline SDN. Fast rerouting (FRR) [118] has been introduced for fast and local reaction without controller intervention. This may also be adopted for SD-WSNs.

6.3.3. Scalability

Scalability is one of the most challenging problems in SDN-based networks [119]. The robustness of the network was studied in [84], but it suffers from scalability issues, which has also to be considered in SD-WSNs. Utilizing several controllers in the network solves the problem but it opens the problem of optimal controller placement [120].

6.4. Security

Many WSNs have mission-critical responsibilities such as military applications. Therefore, security needs to be taken into account in designing the network for such applications [98]. Due to the nature of WSNs, security issues are more complicated than in other network types. The threats and vulnerabilities for SDN-based WSNs are identified in Sec. 5. There is a need for suitable solutions for each of those threats in the future works. Most of current SDN security solutions are adapted for switches and routers.

7. Lessons Learned

We summarize some insights gained during the preparation of this survey.

Sensor nodes have only a limited battery, which constrains their lifetime. Therefore, energy saving is an important goal in most WSNs. This is mostly achieved by adapting the communication range of sensor nodes. The communication range affects the resulting topology and impacts the management of the WSN. The sensing range impacts the coverage area of a node, which is important as most WSNs have been deployed for environmental monitoring. As the adaptation of communication range influences significantly the operation of a WSN, it is a difficult task. We believe that it can be better solved by a powerful server with a central view on the network than in a distributed way. Moreover, distributed control of WSNs by itself causes lots of communication overhead so that the communication of sensor nodes with an SDN controller may save energy. As offloading energy- and communication-hungry tasks to a powerful controller can significantly extend the lifetime of sensor nodes, WSNs may particularly benefit from SDN. However, there are some challenges to solve. So far, there is not yet a standardized architecture for SD-WSN and appropriate hardware is missing. There are some simulation tools for SD-WSN, but no testbeds such as mininet that allows running multiple real nodes on a single machine so that experimentation with SD-WSN requires more effort than in wireline SDN. Data plane and control plane resilience are partially unsolved problems in wireline SDN, which also holds for SD-WSN. When managing a WSN, topology, routing, and various applications need to be jointly optimized, and re-clustering actions may be needed to balance the battery of all nodes. These are demanding tasks even for a central control server and appropriate control strategies are needed. Finally, security in SDN is not fully understood, which is certainly an even bigger problem for SD-WSN as sensor nodes may be even more exposed to potential attackers. Below the line, we believe that the benefits of SDN outweigh potential drawbacks and see SD-WSN as a promising research area.

8. Conclusion

This survey gave a brief overview of WSNs and SDN and introduced the concept of software-defined WSNs (SD-WSNs) including their operations, e.g., topology discovery and routing decisions, that are different from WSNs. Coordination of distributed nodes and energy efficiency are the most important challenges in WSNs. In non-SDN based WNSs, they are mostly solved in a distributed manner. SD-WSNs favor central control. That may save energy because redundant communication can be avoided, energy-constraint nodes can be offloaded from energy-efficient task by moving them to the controller, and application-specific goals may be achieved with fewer active nodes through more intelligent operation. We reviewed advances for WSNs through SDN and challenges for SD-WSNs that should be solved in the future. Finally, we pointed out lessons learned during the preparation of this survey.

References

[1] N. Mckeown, How SDN will shape networking (Oct. 2011).
URL `http://www.youtube.com/watch?v=c9-K5O_qYgA.`

[2] The Open Networking Foundation, Software-defined networking (SDN) definition (retrieved: Jan 2017).
URL `https://www.opennetworking.org/sdn-resources/sdn-definition`

[3] H. I. Kobo, A. M. Abu-Mahfouz, G. P. Hancke, A survey on software-defined wireless sensor networks: Challenges and design requirements, IEEE Access 5 (2017) 1872–1899.

[4] M. Ndiaye, G. P. Hancke, A. M. Abu-Mahfouz, Software defined networking for improved wireless sensor network management: A survey, Sensors 17 (5) (2017) 1031.

[5] N. A. Jagadeesan, B. Krishnamachari, Software-defined networking paradigms in wireless networks: a survey, ACM Computing Surveys (CSUR) 47 (2) (2015) 27.

[6] I. T. Haque, N. Abu-Ghazaleh, Wireless software defined networking: A survey and taxonomy, IEEE Communications Surveys Tutorials 18 (4) (2016) 2713–2737. doi:10.1109/COMST.2016.2571118.

[7] H. M. Ammari, A. Shaout, F. Mustapha, Sensing coverage in three-dimensional space: A survey, Handbook of Research on Advanced Wireless Sensor Network Applications, Protocols, and Architectures (2016) 1.

[8] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: A survey, Comput. Netw. 38 (4) (2002) 393–422. doi:10.1016/S1389-1286(01)00302-4.

[9] L. Yu, N. Wang, X. Meng, Real-time forest fire detection with wireless sensor networks, in: Proceedings. 2005 International Conference on Wireless Communications, Networking and Mobile Computing, 2005., Vol. 2, IEEE, 2005, pp. 1214–1217.

[10] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, J. Anderson, Wireless sensor networks for habitat monitoring, in: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications, ACM, 2002, pp. 88–97.

[11] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, et al., A line in the sand: a wireless sensor network for target detection, classification, and tracking, Computer Networks 46 (5) (2004) 605–634.

[12] H. Mostafaei, M. Shojafar, A new meta-heuristic algorithm for maximizing lifetime of wireless sensor networks, Wireless Personal Communications 82 (2) (2015) 723–742.

[13] D. Ye, D. Gong, W. Wang, Application of wireless sensor networks in environmental monitoring, in: Power Electronics and Intelligent Transportation System (PEITS), 2009 2nd International Conference on, Vol. 1, IEEE, 2009, pp. 205–208.

[14] S. H. Lee, S. Lee, H. Song, H. S. Lee, Wireless sensor network design for tactical military applications: Remote large-scale environments, in: Military communications conference, 2009. MILCOM 2009. IEEE, IEEE, 2009, pp. 1–7.

[15] V. Jelicic, M. Magno, D. Brunelli, G. Paci, L. Benini, Context-adaptive multimodal wireless sensor network for energy-efficient gas monitoring, Vol. 13, IEEE, 2013, pp. 328–338.

[16] J. Yick, B. Mukherjee, D. Ghosal, Wireless sensor network survey, Comput. Netw. 52 (12) (2008) 2292–2330. doi:10.1016/j.comnet.2008.04.002.

[17] I. Howitt, J. A. Gutierrez, Ieee 802.15.4 low rate-wireless personal area network coexistence issues, in: Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE, Vol. 3, IEEE, 2003, pp. 1481–1486.

[18] P. Kinney, et al., Zigbee technology: Wireless control that simply works, in: Communications design conference, Vol. 2, 2003, pp. 1–7.

[19] Z. Shelby, C. Bormann, 6LoWPAN: The wireless embedded Internet, Vol. 43, John Wiley & Sons, 2011.

[20] Isa100 wireless compliance institute (Accessed Jun. 2018).
URL `https://www.isa.org/isa100/`

[21] H. Karl, A. Willig, Protocols and architectures for wireless sensor networks, John Wiley & Sons, 2007.

[22] The Open Networking Foundation, SDN architecture, Tech. rep. (Jun. 2014).
URL `https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR_SDN_ARCH_1.0_06062014.pdf`

[23] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, Openflow: enabling innovation in campus networks, ACM SIGCOMM Computer Communication Review 38 (2) (2008) 69–74.

[24] S. J. Vaughan-Nichols, Openflow: The next generation of the network?, Computer 44 (8) (2011) 13–15.

[25] K. Greene, Tr10: Software-defined networking - mit technology review, Tech. rep. (Feb. 2009).

[26] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, S. Uhlig, Software-defined networking: A comprehensive survey, Proceedings of the IEEE 103 (1) (2015) 14–76.

[27] M. P. Fernandez, Comparing openflow controller paradigms scalability: Reactive and proactive, in: Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on, IEEE, 2013, pp. 1009–1016.

[28] W. Braun, M. Menth, Software-defined networking using openflow: Protocols, applications and architectural design choices, Future Internet 6 (2) (2014) 302–336.

[29] A. Mahmud, R. Rahmani, Exploitation of openflow in wireless sensor networks, in: Proceedings of 2011 International Conference on Computer Science and Network Technology, Vol. 1, 2011, pp. 594–600. doi:10.1109/ICCSNT.2011.6182029.

[30] T. Luo, H.-P. Tan, T. Q. Quek, Sensor openflow: Enabling software-defined wireless sensor networks, IEEE Communications Letters 16 (11) (2012) 1896–1899.

[31] W. L. Lee, A. Datta, R. Cardell-Oliver, Network management in wireless sensor networks, Handbook of Mobile Ad Hoc and Pervasive Communications (2006) 1–20.

[32] M. Kobayashi, S. Seetharaman, G. Parulkar, G. Appenzeller, J. Little, J. Van Reijendam, P. Weissmann, N. McKeown, Maturing of openflow and software-defined networking through deployments, Computer Networks 61 (2014) 151–175.

[33] L. Galluccio, S. Milardo, G. Morabito, S. Palazzo, Sdn-wise: Design, prototyping and experimentation of a stateful sdn solution for wireless sensor networks, in: 2015 IEEE Conference on Computer Communications (INFOCOM), 2015, pp. 513–521. doi:10.1109/INFOCOM.2015.7218418.

[34] B. Lantz, B. Heller, N. McKeown, A network in a laptop: rapid prototyping for software-defined networks, in: Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, ACM, 2010, p. 19.

[35] The cooja simulator (Accessed Feb. 2018).
URL `https://github.com/contiki-os/contiki/wiki/An-Introduction-to-Cooja.`

[36] Contiki: The open source os for the internet of things (Accessed Jun. 2018).
URL `http://www.contiki-os.org/`

[37] B. T. de Oliveira, C. B. Margi, Distributed control plane architecture for software-defined wireless sensor networks, in: 2016 IEEE International Symposium on Consumer Electronics (ISCE), 2016, pp. 85–86. doi:10.1109/ISCE.2016.7797384.

[38] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, et al., Onos: towards an open, distributed sdn os, in: Proceedings of the third workshop on Hot topics in software defined networking, ACM, 2014, pp. 1–6.

[39] T. Rault, A. Bouabdallah, Y. Challal, Energy efficiency in wireless sensor networks: A top-down survey, Computer Networks 67 (2014) 104–122.

[40] C. Zhu, V. C. Leung, L. T. Yang, L. Shu, Collaborative location-based sleep scheduling for wireless sensor networks integratedwith mobile cloud computing, IEEE Transactions on Computers 64 (7) (2015) 1844–1856.

[41] H. Mostafaei, M. S. Obaidat, A greedy overlap-based algorithm for partial coverage of heterogeneous wsns, in: GLOBECOM 2017 - 2017 IEEE Global Communications Conference, 2017, pp. 1–6. doi:10.1109/GLOCOM.2017.8254431.

[42] I. Dietrich, F. Dressler, On the lifetime of wireless sensor networks, ACM Transactions on Sensor Networks (TOSN) 5 (1) (2009) 5.

[43] H. Mostafaei, M. R. Meybodi, Maximizing lifetime of target coverage in wireless sensor networks using learning automata, Wireless Personal Communications 71 (2) (2013) 1461–1477.

[44] Mica wireless measurement system (Feb. 2018).
URL `http://www.contiki-os.org/`

[45] N. Kimura, S. Latifi, A survey on data compression in wireless sensor networks, in: Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on, Vol. 2, IEEE, 2005, pp. 8–13.

[46] L. Wenxing, W. Muqing, W. Yuewei, Energy-efficient algorithm based on multi-dimensional energy space for software-defined wireless sensor networks, in: 2016 International Symposium on Wireless Communication Systems (ISWCS), 2016, pp. 309–314. doi:10.1109/ISWCS.2016.7600920.

[47] N. Abdolmaleki, M. Ahmadi, H. T. Malazi, S. Milardo, Fuzzy topology discovery protocol for sdn-based wireless sensor networks, Simulation Modelling Practice and Theory 79 (2017) 54 – 68. doi:https://doi.org/10.1016/j.simpat.2017.09.004.

[48] Y. Wang, H. Chen, X. Wu, L. Shu, An energy-efficient sdn based sleep scheduling algorithm for wsns, Journal of Network and Computer Applications 59 (2016) 39–45.

[49] B. Wang, Coverage control in sensor networks, Springer Science & Business Media, 2010.

[50] A. Sangwan, R. P. Singh, Survey on coverage problems in wireless sensor networks, Wirel. Pers. Commun. 80 (4) (2015) 1475–1500. doi:10.1007/s11277-014-2094-3.
URL `http://dx.doi.org/10.1007/s11277-014-2094-3`

[51] H. Mostafaei, M. U. Chowdhurry, M. S. Obaidat, Border surveillance with wsn systems in a distributed manner, IEEE Systems Journal-doi:10.1109/JSYST.2018.2794583.

[52] L. Kong, S. Lin, W. Xie, X. Qiao, X. Jin, P. Zeng, W. Ren, X. Y. Liu, Adaptive barrier coverage using software defined sensor networks, IEEE Sensors Journal 16 (20) (2016) 7364–7372. doi:10.1109/JSEN.2016.2566808.

[53] M. Tang, F. Yan, S. Deng, L. Shen, S. Kuang, S. Xing, Coverage optimization algorithms based on voronoi diagram in software-defined sensor networks, in: 2016 8th International Conference on Wireless Communications Signal Processing (WCSP), 2016, pp. 1–5. doi:10.1109/WCSP.2016.7752658.

[54] H. Mostafaei, Stochastic barrier coverage in wireless sensor networks based on distributed learning automata, Computer Communications 55 (2015) 51 – 61. doi:http://dx.doi.org/10.1016/j.comcom.2014.10.003.

[55] M. Cardei, M. T. Thai, Y. Li, W. Wu, Energy-efficient target coverage in wireless sensor networks, in: Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies., Vol. 3, 2005, pp. 1976–1984 vol. 3. doi:10.1109/INFCOM.2005.1498475.

[56] H. Mostafaei, A. Montieri, V. Persico, A. Pescapé, A sleep scheduling approach based on learning automata for wsn partial coverage, Journal of Network and Computer Applications 80 (2017) 67 – 78. doi:http://dx.doi.org/10.1016/j.jnca.2016.12.022.

[57] S. Tomovic, I. Radusinovic, Energy efficient target coverage in partially deployed software defined wireless sensor network, in: International Conference on Cognitive Radio Oriented Wireless Networks, Springer, 2016, pp. 729–740.

[58] W. Qi, Q. Song, X. Kong, L. Guo, A traffic-differentiated routing algorithm in flying ad hoc sensor networks with sdn cluster controllers, Journal of the Franklin Institute-

doi:https://doi.org/10.1016/j.jfranklin.2017.11.012.

[59] O. Younis, M. Krunz, S. Ramasubramanian, Node clustering in wireless sensor networks: recent developments and deployment challenges, IEEE network 20 (3) (2006) 20–25.

[60] G. Gupta, M. Younis, Load-balanced clustering of wireless sensor networks, in: Communications, 2003. ICC'03. IEEE International Conference on, Vol. 3, IEEE, 2003, pp. 1848–1852.

[61] P. G. V. Naranjo, M. Shojafar, H. Mostafaei, Z. Pooranian, E. Baccarelli, P-sep: a prolong stable election routing algorithm for energy-limited heterogeneous fog-supported wireless sensor networks, The Journal of Supercomputing 73 (2) (2017) 733–755. doi:10.1007/s11227-016-1785-9.

[62] B. Alsaify, H. Shen, Power conservation techniques in wireless sensor networks, Handbook of Research on Developments and Trends in Wireless Sensor Networks: From Principle to Practice: From Principle to Practice (2010) 108.

[63] F. Olivier, G. Carlos, N. Florent, Sdn based architecture for clustered wsn, in: Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2015 9th International Conference on, IEEE, 2015, pp. 342–347.

[64] O. Flauzac, C. Gonzalez, F. Nolot, Developing a distributed software defined networking testbed for iot, Procedia Computer Science 83 (2016) 680 – 684, the 7th International Conference on Ambient Systems, Networks and Technologies (ANT 2016) / The 6th International Conference on Sustainable Energy Information Technology (SEIT-2016) / Affiliated Workshops. doi:http://dx.doi.org/10.1016/j.procs.2016.04.151.

[65] H. Yu, Z. Jia, L. Ju, C. Liu, X. Ding, Energy efficient routing algorithm using software defining network for wsns via unequal clustering, in: H. Yuan, J. Geng, F. Bian (Eds.), Geo-Spatial Knowledge and Intelligence, Springer Singapore, Singapore, 2017, pp. 154–163.

[66] N. A. Pantazis, S. A. Nikolidakis, D. D. Vergados, Energy-efficient routing protocols in wireless sensor networks: A survey, IEEE Communications Surveys & Tutorials 15 (2) (2013) 551–591.

[67] J. N. Al-Karaki, A. E. Kamal, Routing techniques in wireless sensor networks: a survey, IEEE Wireless Communications 11 (6) (2004) 6–28. doi:10.1109/MWC.2004.1368893.

[68] W. Xiang, N. Wang, Y. Zhou, An energy-efficient routing algorithm for software-defined wireless sensor networks, IEEE Sensors Journal 16 (20) (2016) 7393–7400.

[69] L. F. d. S. Santos, F. F. d. Mendonça, K. L. Dias, μsdn: An sdn-based routing architecture for wireless sensor networks, in: 2017 VII Brazilian Symposium on Computing Systems Engineering (SBESC), 2017, pp. 63–70. doi:10.1109/SBESC.2017.15.

[70] M. Aslam, X. Hu, F. Wang, Sacfir: Sdn-based application-aware centralized adaptive flow iterative reconfiguring routing protocol for wsns, Sensors 17 (12). doi:10.3390/s17122893.

[71] W. Xiang, N. Wang, Y. Zhou, An energy-efficient routing algorithm for software-defined wireless sensor networks, IEEE Sensors Journal 16 (20) (2016) 7393–7400.

[72] S. Manisekaran, R. Venkatesan, An analysis of software-defined routing approach for wireless sensor networks, Computers & Electrical Engineering 56 (2016) 456 – 467.

[73] J. Wang, Y. Miao, P. Zhou, M. S. Hossain, S. M. M. Rahman, A software defined network routing in wireless multihop network, Journal of Network and Computer Applications 85 (2017) 76 – 83. doi:https://doi.org/10.1016/j.jnca.2016.12.007.

[74] J. Wang, P. Zhai, Y. Zhang, L. Shi, G. Wu, X. Shi, P. Zhou, Software defined network routing in wireless sensor network, in: J. Wan, K. Lin, D. Zeng, J. Li, Y. Xiang, X. Liao, J. Huang, Z. Liu (Eds.), Cloud Computing, Security, Privacy in New Computing Environments, Springer International Publishing, Cham, 2018, pp. 3–11.

[75] S. Misra, S. Bera, A. M. P., S. K. Pal, M. S. Obaidat, Situation-aware protocol switching in software-defined wireless sensor network systems, IEEE Systems Journal PP (99) (2017) 1–8. doi:10.1109/JSYST.2017.2774284.

[76] R. Wang, Z. Zhang, Z. Zhang, Z. Jia, Etmrm: An energy-efficient trust management and routing mechanism for sdwsns, Computer Networks 139 (2018) 119 – 135. doi:https://doi.org/10.1016/j.comnet.2018.04.009.

[77] G. Li, S. Guo, Y. Yang, Y. Yang, Y. Yang, Y. Yang, Y. Yang, Traffic load minimization in software defined wireless sensor networks, IEEE Internet of Things Journal (2018) 1–1doi:10.1109/JIOT.2018.2797906.

[78] L. Peizhe, W. Muqing, L. Wenxing, Z. Min, A game-theoretic and energy-efficient algorithm in an improved software-defined wireless sensor network, IEEE Access 5 (2017) 13430–13445.

[79] R. Silva, J. S. Silva, F. Boavida, Mobility in wireless sensor networks – survey and proposal, Computer Communications 52 (2014) 1 – 20. doi:https://doi.org/10.1016/j.comcom.2014.05.008.

[80] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, G. S. Sukhatme, Robomote: enabling mobility in sensor networks, in: IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005., 2005, pp. 404–409. doi:10.1109/IPSN.2005.1440957.

[81] S. Basagni, A. Carosi, C. Petrioli, Mobility in wireless sensor networks, Wiley Series on Parallel and Distributed Computing. John Wiley & Sons, Inc., Hoboken, NJ, 2008.

[82] A. Dâmaso, N. Rosa, P. Maciel, Reliability of wireless sensor networks, Sensors 14 (9) (2014) 15760–15785.

[83] M. A. Mahmood, W. K. Seah, I. Welch, Reliability in wireless sensor networks, Comput. Netw. 79 (C) (2015) 166–187. doi:10.1016/j.comnet.2014.12.016.

[84] N. Gong, X. Huang, Reliability analysis of software defined wireless sensor networks, in: Asian Simulation Conference, Springer, 2015, pp. 65–78.

[85] Y. Duan, W. Li, X. Fu, Y. Luo, L. Yang, A methodology for reliability of wsn based on software defined network in adaptive industrial environment, IEEE/CAA Journal of Automatica Sinica 5 (1) (2018) 74–82. doi:10.1109/JAS.2017.7510751.

[86] M. Karakus, A. Durresi, Quality of service (qos) in software defined networking (sdn): A survey, Journal of Network and Computer Applications 80 (2017) 200 – 218. doi:https://doi.org/10.1016/j.jnca.2016.12.019.

[87] M. A. Kafi, J. B. Othman, N. Badache, A survey on reliability protocols in wireless sensor networks, ACM Comput. Surv. 50 (2) (2017) 31:1–31:47. doi:10.1145/3064004.

[88] P. Di Dio, S. Faraci, L. Galluccio, S. Milardo, G. Morabito, S. Palazzo, P. Livreri, Exploiting state information to support qos in software-defined wsns, in: Ad Hoc Networking Workshop (Med-Hoc-Net), 2016 Mediterranean, IEEE, 2016, pp. 1–7.

[89] H. Fotouhi, M. Vahabi, A. Ray, M. Björkman, Sdn-tap: An sdn-based traffic aware protocol for wireless sensor networks, in: e-Health Networking, Applications and Services (Healthcom), 2016 IEEE 18th International Conference on, IEEE, 2016, pp. 1–6.

[90] T.-H. Lee, L.-H. Chang, W.-C. Cheng, Design and implementation of sdn-based 6lbr with qos mechanism over heterogeneous wsn and internet., KSII Transactions on Internet & Information Systems 11 (2).

[91] A. De Gante, M. Aslan, A. Matrawy, Smart wireless sensor network management based on software-defined networking, in: Communications (QBSC), 2014 27th Biennial Symposium on, IEEE, 2014, pp. 71–75.

[92] H. I. Kobo, A. M. Abu-Mahfouz, G. P. Hancke, Fragmentation-based distributed control system for software defined wireless sensor networks, IEEE Transactions on Industrial Informatics (2018) 1–1doi:10.1109/TII.2018.2821129.

[93] Y. Zhang, Y. Zhu, F. Yan, W. Xia, L. Shen, Energy-efficient radio resource allocation in software-defined wireless sensor networks, IET Communications 12 (3) (2018) 349–358. doi:10.1049/iet-com.2017.0937.

[94] L. Cheng, C. Wu, Y. Zhang, H. Wu, M. Li, C. Maple, A survey of localization in wireless sensor network, International Journal of Distributed Sensor Networks 8 (12) (2012) 962523. doi:10.1155/2012/962523.

[95] J. Kuriakose, S. Joshi, R. Vikram Raju, A. Kilaru, A review on localization in wireless sensor networks, in: S. M. Thampi, A. Gelbukh, J. Mukhopadhyay (Eds.), Advances in Signal Processing and Intelligent Recognition Systems, Springer International Publishing, Cham, 2014, pp. 599–610.

[96] Y. Zhu, F. Yan, Y. Zhang, R. Zhang, L. Shen, Sdn-based anchor scheduling scheme for localization in heterogeneous wsns, IEEE Communications Letters 21 (5) (2017) 1127–1130. doi:10.1109/LCOMM.2017.2657618.

[97] Y. Zhu, S. Xing, Y. Zhang, F. Yan, L. Shen, Localisation algorithm with node selection under power constraint in software-defined sensor networks, IET Communications 11 (13) (2017) 2035–2041.

[98] X. Chen, K. Makki, K. Yen, N. Pissinou, Sensor network security: a survey, IEEE Communications Surveys & Tutorials 11 (2).

[99] Y. Zhou, Y. Fang, Y. Zhang, Securing wireless sensor networks: a survey, IEEE Communications Surveys Tutorials 10 (3) (2008) 6–28. doi:10.1109/COMST.2008.4625802.

[100] T. Kgogo, B. Isong, A. M. Abu-Mahfouz, Software defined wireless sensor networks security challenges, in: 2017 IEEE AFRICON, 2017, pp. 1508–1513. doi:10.1109/AFRCON.2017.8095705.

[101] S. W. Pritchard, G. P. Hancke, A. M. Abu-Mahfouz, Security in software-defined wireless sensor networks: Threats, challenges and potential solutions, in: 2017 IEEE 15th International Conference on Industrial Informatics (INDIN), 2017, pp. 168–173. doi:10.1109/INDIN.2017.8104765.

[102] I. Ahmad, S. Namal, M. Ylianttila, A. Gurtov, Security in software defined networks: A survey, IEEE Communications Surveys Tutorials 17 (4) (2015) 2317–2346. doi:10.1109/COMST.2015.2474118.

[103] M. Huang, B. Yu, S. Li, Puf-assisted group key distribution scheme for software-defined wireless sensor networks, IEEE Communications Letters 22 (2) (2018) 404–407. doi:10.1109/LCOMM.2017.2778725.

[104] D. Kreutz, F. Ramos, P. Verissimo, Towards secure and dependable software-defined networks, in: Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, ACM, 2013, pp. 55–60.

[105] R. Holz, T. Riedmaier, N. Kammenhuber, G. Carle, X. 509 forensics: Detecting and localising the ssl/tls men-in-the-middle, Computer security–esorics 2012 (2012) 217–234.

[106] J. Huang, C.-C. Xing, C. Wang, Simultaneous wireless information and power transfer: technologies, applications, and research challenges, IEEE Communications Magazine 55 (11) (2017) 26–32.

[107] L. Xie, Y. Shi, Y. T. Hou, A. Lou, Wireless power transfer and applications to sensor networks, IEEE Wireless Communications 20 (4) (2013) 140–145.

[108] W. Ejaz, M. Naeem, M. Basharat, A. Anpalagan, S. Kandeepan, Efficient wireless power transfer in software-defined wireless sensor networks, IEEE Sensors Journal 16 (20) (2016) 7409–7420.

[109] M. Conti, C. Boldrini, S. S. Kanhere, E. Mingozzi, E. Pagani, P. M. Ruiz, M. Younis, From manet to people-centric networking: milestones and open research challenges, Computer Communications 71 (2015) 1–21.

[110] Y.-C. Wang, F.-J. Wu, Y.-C. Tseng, Mobility management algorithms and applications for mobile sensor networks, Wireless Communications and Mobile Computing 12 (1) (2012) 7–21.

[111] M. Bouaziz, A. Rachedi, A survey on mobility management protocols in wireless sensor networks based on 6lowpan technology, Computer Communications 74 (2016) 3 – 15, current and Future Architectures, Protocols, and Services for the Internet of Things. doi:http://doi.org/10.1016/j.comcom.2014.10.004.

[112] J. M. Jaffe, Algorithms for finding paths with multiple constraints, Networks 14 (1) (1984) 95–116.

[113] T. Korkmaz, M. Krunz, Multi-constrained optimal path selection, in: INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, Vol. 2, IEEE, 2001, pp. 834–843.

[114] F. Wang, J. Liu, Networked wireless sensor data collection: Issues, challenges, and approaches, IEEE Communications Surveys Tutorials 13 (4) (2011) 673–687. doi:10.1109/SURV.2011.060710.00066.

[115] Z. Zhao, W. Huangfu, L. Sun, Nssn: A network monitoring and packet sniffing tool for wireless sensor networks, in: 2012 8th International Wireless Communications and Mobile Computing Conference (IWCMC), 2012, pp. 537–542. doi:10.1109/IWCMC.2012.6314261.

[116] S. R. Chowdhury, M. F. Bari, R. Ahmed, R. Boutaba, Payless: A low cost network monitoring framework for software defined networks, in: 2014 IEEE Network Operations and Management Symposium (NOMS), 2014, pp. 1–9. doi:10.1109/NOMS.2014.6838227.

[117] T. Amgoth, P. K. Jana, Coverage hole detection and restoration algorithm for wireless sensor networks, Peer-to-Peer Networking and Applications 10 (1) (2017) 66–78. doi:10.1007/s12083-015-0407-2.

[118] D. Merling, W. Braun, M. Menth, Efficient data plane protection for sdn, in: 4th IEEE International Conference on Network Softwarization (NetSoft 2018), 2018, to appear.

[119] H. I. Kobo, G. P. Hancke, A. M. Abu-Mahfouz, Towards a distributed control system for software defined wireless sensor networks, in: IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society, 2017, pp. 6125–6130. doi:10.1109/IECON.2017.8217064.

[120] B. Heller, R. Sherwood, N. McKeown, The controller placement problem, in: Proceedings of the first workshop on Hot topics in software defined networks, ACM, 2012, pp. 7–12.

Habib Mostafaei is currently pursuing the PhD degree at the department of engineering of Roma Tre University, Italy. He received the M.S. degree in software engineering from the Islamic Azad University, Arak branch in 2009. Prior to the PhD training at Roma Tre University, he was a lecturer at the Computer Engineering Department of Islamic Azad University (2009-2015). He has served as a reviewer for a number of journal and conference papers and he is awarded as an outstanding reviewer for Elsevier Journal of Networks and Computer Applications (JNCA) in October 2016 and for Journal of Computational Science in April 2017. His research interests include Software Defined Networking (SDN), Interdomain routing, and wireless networks.

Michael Menth is professor at the Department of Computer Science at the University of Tuebingen/Germany since 2010 and chairholder of Communication Networks. He studied, worked, and obtained diploma (1998), PhD (2004), and habilitation (2010) degrees at the universities of Austin/Texas, Ulm/Germany, and Wuerzburg/Germany. His special interests are performance analysis and optimization of communication networks, resilience and routing issues, resource and congestion management, industrial networking and Internet of Things, software-defined networking and Internet protocols. Dr. Menth published more than 150 papers in the field of computer networking.