SOFTWARE-DEFINED WIRELESS SENSOR NETWORKS: OPPORTUNITIES AND CHALLENGES

Slavica Tomovic[*], *Milutin Radonjic*[**], *Milica Pejanovic-Djurisic*[***], *Igor Radusinovic*[****]

Keywords: *energy efficiency, SDN, WSN.*

Abstract: Resource limitations of sensor nodes are the main weakness of wireless sensor networks (WSNs). This problem is difficult to alleviate in traditional ad-hoc networks, where each node participates in decentralized routing process and independently determines the next-hop to send data towards the destination. In that regard, this paper presents WSN architecture based on principles of software defined networking (SDN), which enables automatic network reconfiguration and optimal management of energy constrained sensor nodes with limited communications abilities. We discuss implementation challenges and the key benefits of software-defined WSN design, with a special emphasis on advantages offered by centralized routing algorithms.

1. INTRODUCTION

Wireless sensor networks (WSNs) consist of small, low-powered sensor nodes, that measure specific parameters of the environment (e.g. temperature, pressure, motion, etc.) and convert the measurement results in electrical signals. Depending on applications, WSNs may include up to hundreds of sensor nodes, which communicate with the external gateway either directly, or by multi-hop data transmission. The last scenario is known as ad-hoc sensor network, which is self-organized and does not require pre-established network-infrastructure or centralized management. Instead, each node participates in routing and forwards data for other nodes in the network.

[*] S. Tomovic, Faculty of Electrical Engineering, University of Montenegro, Montenegro (e-mail: slavicat@ac.me).

[**] M. Radonjic, Faculty of Electrical Engineering, University of Montenegro, Montenegro (e-mail: mico@ac.me).

[***] M. Pejanovic-Djurisic, Faculty of Electrical Engineering, University of Montenegro, Montenegro (e-mail: milica@t-com.me).

[****] I. Radusinovic, Faculty of Electrical Engineering, University of Montenegro, Montenegro (e-mail: igorr@ac.me).

One of the main WSN performance parameters is reliability, which is hard to achieve due to limited resources and communication abilities of sensor nodes. In today's WSNs, sensor nodes usually forward data to the closest neighbour [1]. This entails that the largest amount of traffic is routed over nodes near the gateway. Since batteries of these nodes rapidly drain, the risk of network collapse increases. Various distributed routing algorithms have been proposed in literature to address this problem [2]. However, most of the solutions are too complex and optimized in a way that practical implementation is not sustainable.

In general, many inherent problems of WSNs are deeply rooted in the network architecture, because each node is implemented as autonomous system equipped with all functionalities from physical up to application layer. Thus, besides traffic forwarding, the nodes perform many control tasks, including routing. Functionalities implemented on the nodes are adjusted to the needs of sensing application. This imposes negative impact on network utilization, because multiple networks are often deployed in the same or overlapping areas for the purpose of different applications, while the same goals could be reached with one programmable network [3].

Having in mind the problems mentioned above, in this paper we consider application of Software Defined Networking (SDN) [4] concept in implementation of WSN infrastructure. SDN is relatively new model of network architecture, initially proposed for wired networks, that decouples the control logic from the traffic forwarding hardware. SDN network intelligence is centralized at programmable controller, which has global view of the network and capability to adjust the network behaviour in real time via automated programs. In this paper, we presented potential benefits of the SDN-based solutions for WSN architecture that could be implemented on open-source hardware platforms Arduino [5] and Raspberry Pi [6]. In particular, we have focused on energy efficiency issue, and demonstrated potential of SDN in that regard through MATLAB simulations.

The rest of the paper is organised as follows. In Section II we discuss the main challenges of introducing SDN concept in WSNs. Section III provides overview of several architectural proposals for software-defined WSNs (SD-WSNs). Then, a few use cases that could especially benefit from SDN paradigm are described in Section IV. Section V presents simulation results of comparison between software-defined and traditional WSNs in terms of energy efficiency. Finally, the conclusion remarks are given in Section VI.

2. SOFTWARE DEFINED WSN AND RELATED CHALLENGES

Three-layer SDN architecture is illustrated in Fig. 1. One of its determining features is that network devices only forward traffic, while forwarding decisions are performed by the external controller [4]. Communication among different layers of the architecture is achieved through the northbound and the southbound interfaces. The first interface connects applications to the controller while the second interface handles communication between controller and data-forwarding plane [7]. OpenFlow protocol is the most commonly used southbound interface [8]. It enables the flow-based traffic control. The data plane devices differentiate and process traffic flows according to instructions of the controller. To define a flow, controller can use any subset of L2-L4 packet header fields supported by OpenFlow [8], along with the identifier of the interface with incoming packet.

This highly-granular control allows network to dynamically adjust to the application and user requirements.

There is also an increasing interest for adoption of SDN in wireless domain, where decoupling of the control and the data plane promises additional benefits [7]. In wireless networks, there is almost total consensus about technical solutions that should be used at physical and link layer of protocol stack. However, at higher layers different protocols are used, depending on specific scenario. For example, energy constrained WSNs are usually based on IEEE 802.15.4 protocol, but the higher layers often use ZigBee or 6LOWPAN, which are not compatible [10]. Also, there is no consensus regarding routing protocols to be used and many management operations, which significantly limits possibility of sensor node migration from one network to another. In SDN networks these limitations are less pronounced, because all management operations are logically centralized and completely separated from the forwarding operations. What is most important, the network functions are software-defined and can be dynamically changed according to the needs.

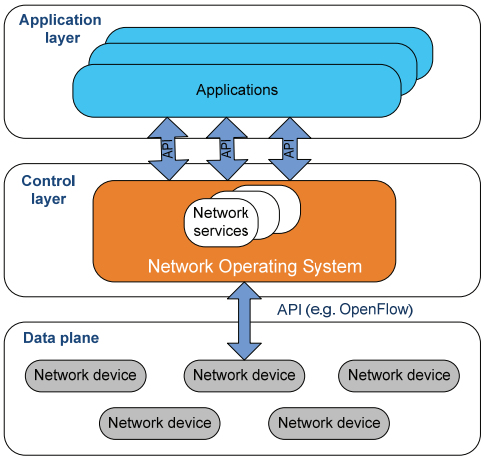


Fig. 1. SDN architecture

Application of SDN concept in WSN networks is very challenging due to limited energy, memory and computational resources of sensor nodes. One option to reduce energy consumption is to use duty cycle techniques, which imply only periodic activity of radio interfaces. Another option is to perform data aggregation on some nodes in the network. At the expense of larger delay, the aggregation increases energy efficiency because redundant data are eliminated within the network. The third option, that we mostly analysed in this paper, is to evenly distribute traffic load with centralized routing algorithm. However, implementation of these mechanisms requires southbound interface that will allow wider spectrum of flow definitions and processing actions. Unlike OpenFlow version 1.0, which differentiates traffic flows only based on TCP(UDP)/IP header fields, this new interface should take into account protocols typical for WSN networks. Also, acquiring data of interest (e.g. temperature > 40^{o}C) in some scenarios is more important than knowing the node which sent the data, so *attribute-based* routing should be considered as well [3].

3. THE SYSTEM STRUCTURE

Implementation of SD-WSN network requires development and integration of large number of hardware and software components. Fig. 2 illustrates the network model considered in this paper. The network consists of distributed sensor nodes and a gateway where the collected data are either processed locally or sent via broadband network to a remote location. Such a network model is typical in precision agriculture applications, where various environmental parameters are sensed over the large areas. Because large-area WSNs usually require battery power supply, architecture of sensor nodes is kept simple, while all decisions regarding the crop treatment are made remotely, usually in the Cloud. The sensor nodes could be designed as combination of sensors, microcontroller and XBee module [11]. The XBee module can be configured into three types of devices: coordinator, router, and end device. Coordinator is able to control the entire network. Router can relay messages in a tree or mesh network topologies. End device can only communicate with the coordinator or the router. Only one coordinator can exists in a network, but the number of router or end devices is unlimited. On the gateway, XBee module is configured as a coordinator, while on sensor nodes it acts as a router.

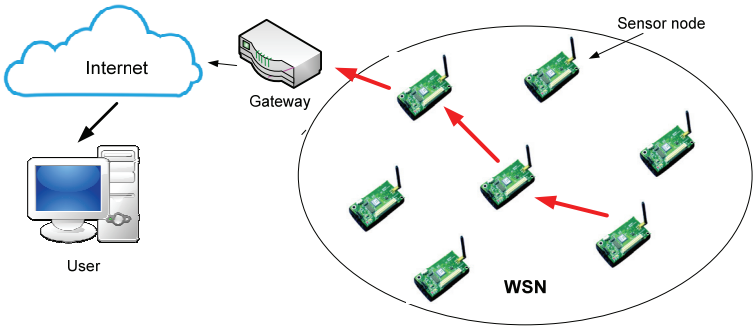
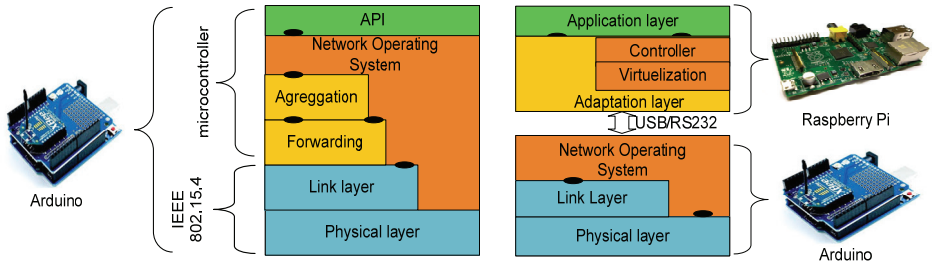


Fig. 2. The considered WSN model

We assumed that the role of SDN controller is assigned to the network gateway. Since SDN controller performs all control functions, it requires significantly higher computational and communication capabilities then sensor nodes. For this purpose, the small single-board computer that runs some of the Linux OS distributions, such as the Raspberry Pi device, could be used. Sensor nodes could be implemented on Arduino platform, which is characterized by significantly lower power consumption.

A. Architecture of SDN sensor node

Fig. 3(a) illustrates cross-layer architecture of SDN sensor node proposed in [10]. Beside the standard physical and link layer functionalities, the microcontroller performs additional functions that are organized in three layers. The Forwarding layer accepts packets from the Link layer and processes them in accordance with the controller's instructions stored in so-called flow table. Due to specificity of WSN environment, format of flow table is significantly modified compared to the OpenFlow [9]. Traffic classification



(a) (b)

Fig. 3. SDN protocol stack[10]

into flows is performed on the basis of specific bytes of the received packets. The table entries consist of classification windows, action field that defines the way of processing, and counters used for statistical purposes (Table 1). Each classification window consists of four fields that define: the number and location of bytes that should be analysed (*size* and *addr* fields), relational operator that will be used during the analysis (=,≠,<,>), and value which must be matched against the specified byte(s). Set of actions includes: forwarding, field modification, packet dropping, data aggregation and deactivation of radio interface. Depending on the type of action, the '*value*' field may indicate: the next hop on the route, byte to be modified and its new value, packet rejection probability, the next hop for the aggregated data, or interval during which the radio interface must be turned off.

Table I

FORMAT OF FLOW TABLE [10]

Window 1				...	Action		Stats
Size	**Op.**	**Addr.**	**Value**		**Type**	**Value**	**Counters**
2	=	2	172.24	...	Forward	170.21	17
2	=	2	170.16	...	Drop	1	3
2	≠	2	170.25	...	Forward	170.22	3

The Aggregation layer runs on top of the Forwarding layer. It performs actions required to aggregate information flowing through the network. According to the controller's instructions, this layer may combine several accepted packets in one containing information about average, maximum or minimum value of the specific parameter measured by sensors. Network Operating System (NOS) layer has access to all the other protocol layers on the device, and controls their behaviour.

Software-defined WSNs could be much sooner brought from idea to reality if the controller's southbound interface is not developed from the scratch. Thus, probably the simplest form of implementation is to extend the OpenFlow protocol which is used in wired networks. A new versions of OpenFlow protocol (1.2 and newer versions) offer high flexibility in flow definition. Instead of static, fixed length match structure, a new extensible match structure has been introduced (OXM match [12]) which allows controller to specify any "set-field" action or match fields in general. This means that OpenFlow could provide support for the special addressing schemes used in WSNs. Although that support is not straightforward, with appropriately defined OXM fields in flow tables traffic

could be classified based on ZigBee network address, or even based on information carried in the packet payload, as proposed in [3].

B. Architecture of the gateway node

Fig. 3(b) shows architecture of the network gateway. With regard to the first two layers of the protocol stack, there is no difference compared to ordinary sensor nodes. However, functions of the higher protocol layers require more computing and communication capabilities. Thus, we envisioned them to run on Raspberry Pi platform. Communication between Raspberry Pi board and Arduino device can be achieved via USB or some other communication interface. The principle of operation is as follows. Link layer forwards received packets to the Adaptation layer, which shapes the messages in the format understandable to WPAN (Wireless Personal Area Network) devices. Virtualization layer uses the local information collected by sensor nodes to build a detailed network graph, including: current battery capacity and neighbour list of all nodes, quality of the links, etc. In addition, the Virtualization layer enables the coexistence of multiple logical networks that are independently controlled by separate controllers. It transparently filters packets from different logical networks and directs them to corresponding controllers. In this way, network provides plug-and-play support for various applications. The Controller accepts packets which sensor nodes were not able to classify in one of the existing flows, and based on current network status determines how the packets should be treated. Application layer includes different network services and management policies. Sensor nodes use beacon mechanism to learn a path towards the controller [10]. The controller periodically broadcasts beacon packet in the network. At each hop this packet is updated to carry information about the current distance from the gateway (in a number of hops) and a measurement of the local battery level. Based on these information and measured *received-signal-strength indicators* (RSSI), each sensor node identifies the most suitable next hop towards the gateway/controller. In pre-defined time intervals the nodes report list of neighbours and all the measured RSSI values to the controller, which builds detailed network representation.

It should be noted that OpenFlow protocol is primarily designed for IP networks. It assumes that both SDN controller and OpenFlow switches are identified by IP addresses. In order to ensure reliable and in-order message delivery, OpenFlow relies on TCP connectivity, which is generally unavailable in WSN networks [3]. Therefore, if non-IP sensor nodes are used, features of OpenFlow protocol could be exploited only if transport protocol is overlaid directly over WSN.

4. BENEFITS OF SDN APPROACH

A. Energy efficiency

One of the key requirements in WSNs is energy efficiency. Sensor nodes are usually battery-powered, so energy consumption directly impacts the network lifetime. This imposes strict limitations on the design and implementation of routing algorithms in the network. For example, when the gateway is too distant from sensor node, direct transmission has to be avoided because the energy loss can be quite extensive. However, conventional multi-hop communication such as Minimum Transmission routing (MTE)

often have equally undesirable effect because the nodes closest to the gateway rapidly drain their batteries [1]. By centralizing the network control plane, SDN allows implementation of more sophisticated routing algorithms. The controller could be programmed to periodically collect information about current battery level of sensor nodes, and use them to make energy-optimized routing decisions. Such routing decisions should lead to increased network lifetime through balanced energy consumption among the sensor nodes. On the other hand, since all control functions are run by the controller, SDN sensor nodes do not consume energy to run routing algorithms and exchange huge amounts of control traffic. Complex tasks such as traffic engineering, virtualization and QoS (Quality of Service) provisioning could be performed without impact on the network lifetime, since the control software runs on machine having much more resources than ordinary network node.

B. Mobility support

In some scenarios sensor nodes are characterized by high level of mobility (e.g. VANET networks). Since that entails frequent topology changes, convergence time of distributed routing protocol may be significant and cause high packet loss rate. On the other side, SDN provides better visibility of the mobility events and allows dynamic adjusting of the flow table rules. For example, the sensor node may be configured to inform the controller about their mobility, or protocol for mobility management could be implemented on the controller [13]. In this way, the controller could track location of sensor nodes and timely update the flow table rules in the network.

C. Network management

In traditional networks there is no unique method for configuring the network devices. In fact, configuration mechanisms are vendor-specific. Therefore, the use of multi-vendor network components significantly complicates the management process. In WSNs management complexity is even more pronounced. Deployment of a new sensing application, or a new network protocol, requires re-programming of microcontrollers. Sometimes this requires physical access to the sensor node, which may not be feasible [13]. SDN promises to significantly simplify network configuration and management of network resources. Centralized control plane abstracts the forwarding hardware, thereby eliminating the need for separate versions of the control application for different types of sensor nodes. The importance of the network programmability will only increase with the growing interest in Internet of Things (IoT). Clearly it will be hard to manually enforce security and other policies to the myriad of devices with different capabilities. The IoT deployments are expected to be fundamentally heterogeneous in terms of used communication technologies (ZigBee, Wi-Fi, NFC, etc.) and resource availability. While this makes opportunities for a wide range of new applications, many new challenges arise, such as: sharing of network and sensor resources among applications, traffic differentiation, QoS provisioning and connectivity management. In this context, SDN capabilities of network virtualization, centralized resource allocation and dynamic load management will be particularly useful.

5. COMPARISSON WITH MTE NETWORKS

In this Section we compare the performance of SD-WSN and traditional WSN that uses MTE routing protocol. MTE is conventional routing protocol that aims to minimize energy dissipated on data transmission. The operation principle is simple: each sensor node communicates with the closest neighbour on the way to the gateway. For SD-WSN we assumed the architecture proposed in [10], which is briefly described in Section IV. In order to verify the efficiency of SDN-based network model, we performed set of MATLAB simulation and measured performance in terms of WSN lifetime. The behaviour of SD-WSN was imitated by centralized routing logic that uses the following link cost:

$$\cos t(l) = \frac{EC(l)^{\alpha}}{R(source\_node)^{\beta}} \tag{1}$$

where R denotes a residual energy of the source node and EC energy needed to transmit and receive packet on the link. Parameters α and β are configurable. They can be chosen to find the minimum energy path or the path with nodes having the most energy, or a combination of the above. In our simulation we set them to values 1 and 4, respectively. Once the link costs are determined, Dijkstra algorithm is used to find the least cost path towards the gateway.

We assumed the same radio models discussed in [1],[14]. Energy consumed for the transfer of a k bit message between two sensor nodes separated by a distance of r meters has been calculated as follows:

$$E_T = E_{Tx}k + E_{amp}k = \begin{cases} E_{Tx}k + \varepsilon_{FS} r^2 k, & r \le r_o \\ E_{Tx}k + \varepsilon_{TW} r^4 k, & r \ge r_o \end{cases} \tag{2}$$

$$E_R = E_{Rx}k \tag{3}$$

where E_T denotes the total energy dissipated in the transmitter and E_R is the total energy dissipated in the receiver electronics. Parameters E_{Tx} and E_{Rx} are per-bit energy dissipations for transmission and reception, respectively. During transmission, additional energy is dissipated to amplify the signal (E_{amp}). Parameters ε_{FS} and ε_{TW} denote amplifier parameters corresponding to the free-space and two-ray propagation models, respectively, while r_o is threshold distance given by the expression:

$$r_o = \sqrt{\varepsilon_{FS} / \varepsilon_{TW}} \tag{4}$$

All the radio model parameters were configured to values used in [1],[14] ($E_{Tx} = E_{Rx} = 50nJ/bit$, $\varepsilon_{FS} = 10pJ/bit/m^2$, and $\varepsilon_{TW} = 0.0013pJ/b/m^4$).

Throughout the simulations we considered a 100-node network, with nodes randomly distributed in a 200mx200m area. We assumed that sensor nodes are within communication range of each other and the gateway. The gateway was positioned so that is at least 20m from the nearest node (x=100, y=220). The initial energy of each node was set to 1J. Furthermore, we assumed that sensor nodes send data at fixed rate. Namely, at pre-defined time-steps or "round" of the simulation, each node generated a 2000-bit data packet to the gateway.

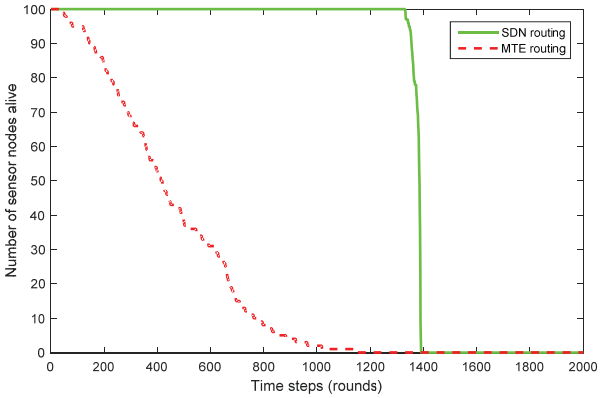


Fig. 4. Lifetime of SD-WSN and MTE WSN in the simulated scenario

Fig. 4 shows results in terms of number of sensor nodes alive as a function of the number of sensing rounds. The results clearly show that SDN significantly prolongs the network lifetime. In MTE scenario the first node "died" after only 48 rounds, and after 1137 rounds all the nodes were dead. The nodes closest to the gateway suffered the highest energy loss (Fig. 5) due to large amount of data they forwarded for the other nodes in the network. Their batteries drained very fast, causing the increase in energy required by other nodes to reach the gateway. In this way the cascading effect is created which reduces the network lifetime. In SDN scenario number of sensor nodes alive reduces much slower, because routing decisions are made by considering the remaining energy of each sensor node. If we define the system lifetime as the number of rounds for which 75% of the nodes remain alive, in the simulated scenario the proposed SDN solution exceeds the MTE lifetime more than 8 times.

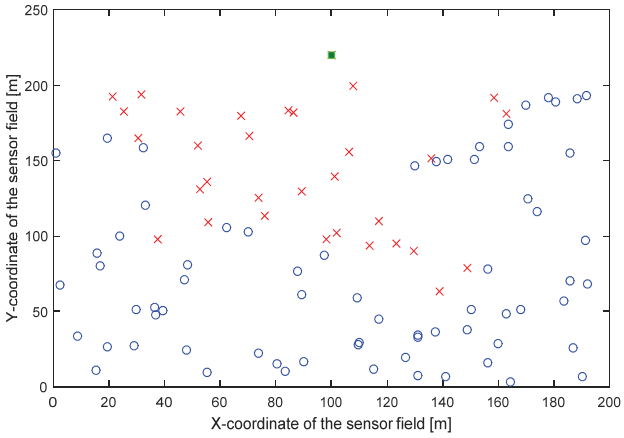


Fig. 5. Sensor nodes that remained alive (circles) and those that are dead ('x') after 300 rounds in MTE network. The square shape denotes the gateway.

6. CONCLUSION

In this paper we explored possibility of applying SDN concept in WSN networks. We identified major weaknesses of WSN networks in general, and presented new SDN-based WSN architecture. In particular, three possible benefits of software-defined network design have been identified: energy efficiency, mobility support and management simplicity. Through MATLAB simulations it has been shown that significant energy savings could be achieved without duty cycle and data aggregation mechanisms, just using a simple centralized routing algorithm. An important step in validating the results is to implement a prototype version of SDN architecture. In the future, we aim to make such a prototype on open-source hardware platforms Arduino and Raspberry Pi, and demonstrate its flexibility by implementing different routing logic for different types of applications, such as data collection applications and event-driven application.

ACKNOWLEDGEMENT

This work is supported by the Montenegrin Ministry of Science under grant 01-451/2012 (FIRMONT) and the BIO-ICT Centre of Excellence (Contract No. 01-1001) founded by Ministry of Science of Montenegro and the HERIC project.

REFERENCES

[1] S. D. Muruganathan, D. C. F. Ma, R. I. Bhasin and A. Fapojuwo, "A centralized energy-efficient routing protocol for wireless sensor networks," *IEEE Comm. Mag*., vol.43, pp. 8-13, 2005.
[2] R. Soua and P. Minet, "A survey on energy efficient techniques in wireless sensor networks," WMNC, 2011 pp. 1-9, Oct. 2011.
[3] T. Luo, H.-P. Tan, and T. Quek, "Sensor OpenFlow: Enabling Software-Defined Wireless Sensor Networks," *IEEE Communications letters*, vol. 16, issue 11, 2012.
[4] Open Networking Foundation, "Software Defined Networking: the new norm for networks," Web. White Paper, Retrieved Sept. 2015.
[5] Arduno platform: www.arduino.cc/.
[6] Raspberry Pi platform: www.raspberrypi.org/
[7] S. Tomovic, M. Pejanovic-Djurisic, I. Radusinovic, "SDN-based mobile networks: concepts and benefits," *Wireless Personal Communications,* vol. 78, issue 3, pp. 1629-1644, Oct. 2014.
[8] N. McKeown, T. Anderson, H. Balakrishnan, J. Rexford, "OpenFlow: Enabling innovations in campus networks," ACM SIGCOMM, vol. 38, issue 2, pp. 69–74, 2008.
[9] S. Tomovic, N. Prasad, I. Radusinovic," Performance comparison of QoS routing algorithms applicable to large-scale SDN networks," IEEE Eurocon 2015, pp. 172-177, Salamanka, Spain September 2015.
[10] S. Costanzo, L. Galluccio, G. Morabito, and S. Palazzo, "Software defined wireless networks: Unbridling sdns," EWSDN, pp. 1–6, 2012.
[11] ZigBee Alliance: available at http://www.zigbee.org/
[12] OpenFlow switch specification v1.3.4, Retrieved: Sept. 2015, Available at: https://www.opennetworking.org/
[13] A. Gante, M. Aslan, and A. Matravy, "Smart Wireless Sensor Network Management Based on Software-Defined Networking," 27th *QBSC*, pp. 71-74, 2014.
[14] W. R. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," 33rd Hawaii Int'l. Conf. Sys. Sci., Jan. 2000.