The Presidium of Wireless Sensor Networks - A Software Defined Wireless Sensor Network Architecture

Donna O'Shea(✉), Victor Cionca, and Dirk Pesch

Nimbus Centre for Embedded Systems Research, Cork Institute of Technology, Cork, Ireland
{donna.oshea,victor.cionca,dirk.pesch}@cit.ie
http://www.nimbus.cit.ie

Abstract. Software Defined Networking (SDN) is emerging as a key technology to deal with the ever increasing network management burden created by our increasingly interconnected world. Wireless sensor network (WSN) are part of this interconnection, enabling to connect the physical world to the cyber world of the Internet and its networks. This connection of physical items, "Things", to the Internet in the form of an Internet of Things is creating many new challenges for the management of the Internet networks. SDN moves away from a distributed management approach that has been at the core of wireless sensor networks since their inception and introduces a centralised view and control of a network. We believe that the SDN concept as well as the general compute virtualisation enabled through infrastructure as a service can offer the required flexible management and control of the network of Things. While the application of SDN to WSN has already been proposed, a comprehensive architecture for Software Defined Wireless Sensor Networks (SD-WSN) is currently missing. This paper provides a survey of related work considering both SDN and centralised non-SDN approaches to network management and control, examines the challenges and opportunities for SD-WSNs, and provides an architectural proposal for SD-WSN.

Keywords: Wireless sensor networks · Software defined networking

1 Introduction

Wireless sensor networks (WSN) have emerged over the past 15 years as a key technology to interface computing systems with the physical world and being a central part of cyber-physical-systems and machine to machine communication systems. The community envisages WSN deployments throughout our environments, producing data for specific use cases such as environmental monitoring in buildings, traffic monitoring in cities, flood monitoring in river estuaries, and many more. The remote nature of WSN deployments requires unattended operation and a high degree of autonomy and self-configuration. From early on,

D. O'Shea and V. Cionca—The first two authors should be regarded as joint First Authors.

R. Agüero et al. (Eds.): MONAMI 2015, LNICST 158, pp. 281–292, 2015.
DOI: 10.1007/978-3-319-26925-2_21

the community traded centralised protocols for distributed, self-organising, self-healing, behaviour, sometimes with emergent properties inspired from nature [20]. In reality though, such deployments rarely functioned correctly and most deployments today are heavily monitored and controlled as nodes fail often and need manual intervention [4, 15].

Another application of sensor networks is in critical or real-time applications where there are strict timing and reliability requirements that have been addressed by implementing complex, application-driven protocols [17]. This may be a viable option for private, isolated and application-specific deployments, however it is less so if the WSN infrastructure is to be shared by multiple stake-holders, which is the case in virtualised WSN environments. In a shared WSN infrastucture, different occupants will have differing QoS requirements, which can't be addressed in an application-specific way but require a wide and detailed picture of the network state to make informed decisions about optimal traffic routing.

What the WSN community needs right now is increased control over the network to understand radio behaviour, diagnose and debug faults and ensure QoS. A relatively recent technology that provides increased control in networking is Software Defined Networking, or SDN. Software Defined Networking technology is under development as a mechanism to deal with the ever increasing configuration and management burden that large scale enterprise networks, data centre networks, and next generation mobile/wireless networks pose on network administrators. SDN simplifies network management by reducing the complexity of network elements (typically switches) to a rule table comprising match elements and actions, fully and directly controlled by a central manager that has complete control over the network [2]. The benefits of SDN are:

– a more programmable network where new protocols can easily be deployed;
– deterministic control that increases predictability of network behaviour over distributed protocols;
– optimal control enabled by a complete model of the network state.

While initial deployments in SDN were developed for wired networks, more recently it has excited the curiosity of researchers in wireless networks [12, 31] and WSNs (discussed further in Sect. 4.1). SDN has been a paradigm shift for network management, and it would probably provide an even more valuable shift for WSN, where resource constraints are as important as they are. Applying SDN to WSN however, represents a very different environment from what SDN was originally designed for in enterprise networks: low data rates, low performance and unreliable links. This paper joins an increasingly longer list of works that try to integrate SDN concepts into the WSN domain. However, the existing literature seems to be disconnected from previous WSN research and does not highlight previous forays into centralised WSN control, which, ultimately, is the core concept of SDN. This leads to an incomplete understanding of the challenges faced by an implementation of SDN in WSN and solutions that cannot work due to issues of scale and inconsistency. This paper proposes a possible architecture for Software Defined Wireless Sensor Networks (SD-WSN) and attempts to expose the challenges and opportunities that arise in this application of SDN, by analysing recent SD-WSN approaches together with older attempts at centralised WSN control.

2 Architecture for Software Defined Wireless Sensor Networks

The following section presents an architecture for Software Defined WSNs, called SURF (Service-centric networking for URban-scale Feedback Systems).

Typical SDN architectures consist of simple,"dumb", switches that are directly controlled by a logically centralised network manager. The manager has complete knowledge of the network topology and its state and therefore can run centralised algorithms for routing, load-balancing, etc. The controller talks to switches through a south-bound interface, the most common being OpenFlow [2]. The controller also exposes a north-bound interface for developers of network applications, such as routing algorithms, firewalls, proxies, etc., however, this one hasn't been standardised yet.

The SURF architecture acknowledges that sensor networks differ in many ways from the description above, the main difference being that nodes are not only switches (or routers, to be precise), but they also have one or more application components. In WSN we are not interested that much in network applications such as firewalls, but more in sensing applications, such as assigning a subset of the nodes to a single application (*e.g.* get the temperature in the city centre). Finally, the WSN case that is being explored is that of a large infrastructure shared by multiple stakeholders with different requirements.

The focus in a WSN SDN is on optimal resource sharing, from the point of view of sensing and communication. The SURF controller, presented below in Fig. 1 has the following capabilities:

- set up and manage data flows through the network that maintain a required level of QoS;
- find the optimal subset of nodes that can service an external sensing request, in terms of quality of sensing and communication;
- dynamically adjust allocations of data flows and sensing applications, by migrating flows or applications, in order to respond to external changes (*e.g.* interference) or reallocation requests (*e.g.* resources required by a higher priority application).

The SURF architecture can be described according to the following layers:

- **Network Applications.** This layer comprises of the business and network applications that monitor and control a set of resources managed by one of more SDN controllers.
- **Controller.** Through its northbound API and access to its Network Information Base (NIB) [1], the main responsibility of the controller is to faithfully execute the requests of the applications defined though this layer. It also is responsible for: resource monitoring and (re)optimisation; responding to and

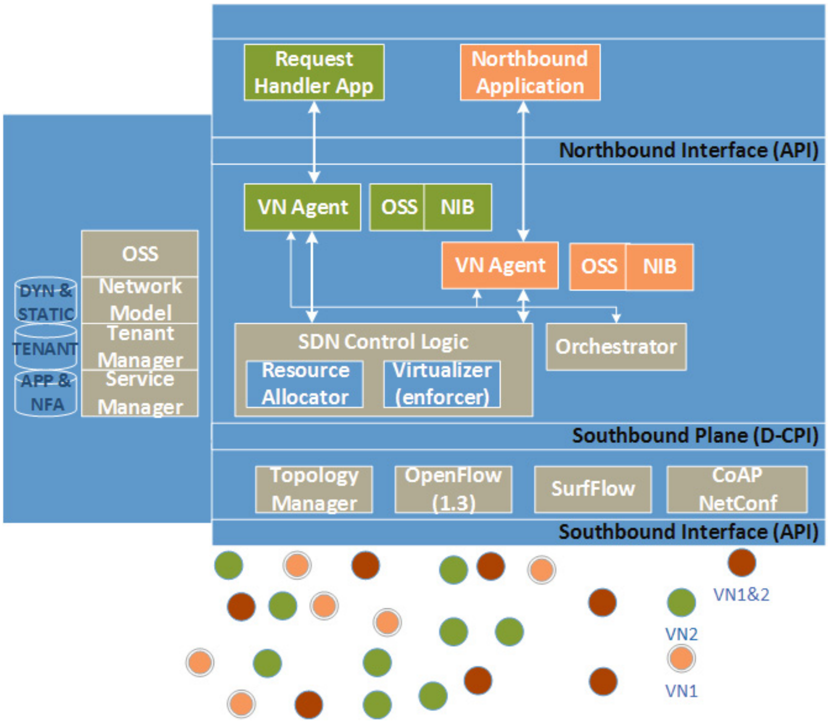

Fig. 1. Architecture of a software-defined wireless sensor network controller

generating events as a result of changes in the underlying network; and computing a collection of packet forwarding rules. These forwarding rules are then installed into the WSN nodes via the southbound API.

– **Physical and Virtual WSN.** This consists of the physical or virtualised network elements which can implement the decisions made in the controller layer issued via the southbound interface.

These layers are further described in detail below.

Northbound Layer. Within the northbound plane of the controller, programmatic interfaces also referred to as northbound Application Programming Interfaces (APIs) exist to create network services. Northbound APIs insulate the: applications from details of the network that are not needed; and the Network Operating System (NOS) (described further in Sect. 3.1) from applications allowing it to focus on its primary concern of dealing with application requests. Northbound applications can invoke external services potentially located in the cloud and may orchestrate applications from other SDN controllers to achieve its objectives. Applications at this layer receive events and notifications about the state of Virtual Networks (VNs) and can alter the state of VNs with varying levels of QoS and bandwidth. In the architecture described below it is important to recognise that we envisage that multiple physical SD-WSN deployments can be combined into a single virtual sensor network. In the case where each individual SD-WSN is controlled by a controller, we need to be able to have an over-arching control function that can deal with multiple controllers, perhaps with multiple owners. This control can be implemented as a client SDN controller or northbound application that invoke external services, applications or other SDN controllers to achieve its functional and business objectives.

Controller Layer. One of the key motivations behind our proposal for SD-WSN, is that the network can be virtualised to enable multiple users and use cases through multi-tenancy, that is multiple virtual overlays over the same physical network to allow multiple uses of the same physical infrastructure. This is also shown in Fig. 1 with two virtual networks overlaying the same physical infrastructure. To support this the SURF SDN control logic consists of a resource allocator, virtualizer and orchestrator the functions of which are further described below:

– **Resource Allocator** is the entity that is responsible for determining if a Virtual Network (VN) or Network Function Virtualisation (NFV) request can be accommodated by the network. In the event that the service request can be supported the resource allocator interacts with the virtualiser to allocate the physical resources that will form part of the VN.
– **Virtualiser** is responsible for creating an VN agent that represents the resources through a subset view of the NIB and actions available to the application. While this agent runs technically in the controller it should be noted that it is a trusted application to the client or application.
– **Orchestrator** To support NFV, service function chains which are used to compose network services, need to be flexibly compose network functions such as firewalls and data aggregation as independent services following Service-Oriented principles [5]. The orchestrator within the Surf architecture is used to provide this functionality. The orchestrator is also responsible for resolving conflicts between different applications and to ensure optimal performance in terms of resource utilisation, overhead, sleep schedules and routing is achieved.
– **Management** The controller also includes a management plane, which consists of a service manager, a tenant manager, a physical network model that keeps track of the physical infrastructure, and operation support services (OSS). The network model maintains a database of the network dynamics, the tenant manager has a database of tenant functions and the service manager maintains a database of virtual network applications and functions.

Physical and Virtual WSN. Communication between the controller and the physical/virtualised WSN nodes is achieved through the southbound interface, which is implemented through suitable protocols. Openflow is used in enterprise networks and a modified version such as proposed in [18] could be adopted here. However, we think that an extension of CoAP could also be a suitable alternative as it is already well established within the WSN field. We are also investigating alternative protocols that are tailored to the specifics of WSN. The southbound plane of the controller is expected to support multiple protocols that are designed bearing in mind the capabilities of the underlying infrastructure. In addition, this plane of the controller has a topology manager which updates the NIB for the SDN control logic. This layer also considers the modifications necessary on the sensor node protocol stack required to support communication with the controller via the southbound API, which is shown in Fig. 2. Through the separation of control and data plane, network nodes can have multiple data plane protocol stacks. In Fig. 2, two different data plane protocol stacks are shown. The control plane is implemented through a suitable control protocol as indicated above. The control functions configure the access to the physical communication medium via the MAC and physical layers. Depending on the services that are supported by the data plane application layers, the control plane needs to configure sleep schedules, medium access control, routing, and perhaps even data aggregation.

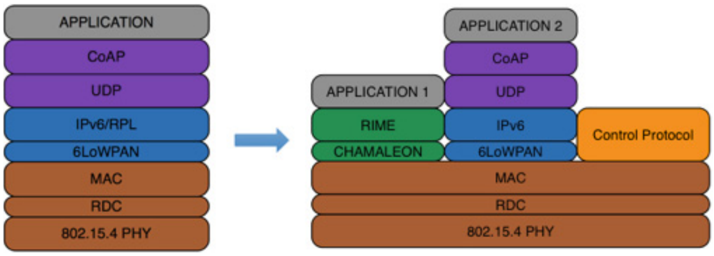

Fig. 2. Software defined wireless sensor node

3 Challenges and Opportunities

In the following we discuss the opportunities and challenges in migrating from the distributed status quo to a centralized management architecture as proposed for our SD-WSN concept.

3.1 Opportunities

This paper makes the case that there is a valid reason for adapting SDN concepts to WSN based on the main opportunities outlined below.

Network Operating System (NOS). In SDN, the NOS allows the decomposition of network operation into three distinct layers: the data plane, the management layer that is responsible for building and maintaining an abstract view of the network state, and the control logic that performs operations on network devices depending of its view of network state. These abstraction levels: "extract simplicity" from the network, reducing the need to "master complexity", make it possible to create network aware applications with the ability for expressive algorithms to reconfigure the underlying network in response to changing user applications requirements [25,28]. In WSN this abstract view of network state (represented as a network model) could be used to provide optimal resolution over a number of WSN constraints such that in general, a maximization of resource utilization through optimal task scheduling could be achieved (with the aim of minimising packet loss and latency, redcing energy consumption and maximising network lifetime).

Network Virtualization. Applying virtualization to Wireless Sensor Networks (WSNs) has been proposed recently [14,16]. WSN virtualization allows the evolution of sensor networks in their current manifestation, as isolated and application specific deployments. In this evolved view, the sensing abilities of WSN nodes can be shared among various federations and composed to form new applications and services beyond its original purpose or design. Separating the network infrastructure and its ownership also means that the traditional benefits associated with virtualization can also be leveraged in WSNs. These benefits include economies of scale, reduced cost of ownership and reduced cost to customers. While SDN is considered an enabler for network virtualization, it is important to recognise that the features offered by SDN were not directly designed to facilitate the creation of virtual networks, although SDN can be leveraged to facilitate network virtualisation in WSNs.

Tussle Networking. Tussle in networking is used to describe the contention among parties with conflicting interests. Within the context of WSN one of the tussles is the conflict between the irreconcilable network stacks that sit on top of the main standardized access technology IEEE802.15.4. These include: Zigbee, WirelessHART and RPL. As a result of these differences, situations may emerge that nodes cannot participate in the WSN even though all the devices use the same underlying access technologies. SDN provides mechanisms to deal with this tussle across heterogeneous WSN deployment by: defining routing functions through software that can be changed dynamically through a high level programming abstraction and facilitating layer 2 network virtualization.

3.2 Challenges

It is important to recognise that SDN is not a panacea in all situations and challenges exist in collecting data to provide a *consistent* and up-to-date network model to leverage the proposed opportunities. These challenges are further described below.

Out of Band Signalling. In enterprise networks there is either an out-of-band channel between the controller and each switch, otherwise secure tunnels are used that take advantage of the high data rates. The topology is static and the links are stable. Sensor networks have unreliable links, a dynamic topology and generally low data rates. An out-of-band channel could be obtained using dedicated higher power radios, but that would increase costs.

Dynamic Changes. Maintaining a consistent central model of the network when the network topology is dynamic requires periodic updates from each node. WSN attributes, such as the quality of links, can change quickly [3]. It is obviously impractical (if not impossible) to update the central model at each unit of change in a node's attribute set. The question is, then, *what is the smallest change in the sensor network that requires an update of the central model*? Obviously this depends on higher level policies, such as the range of QoS levels demanded from the network.

Low Data Rates. Another factor limiting the quantity of SDN control traffic is the generally low data rate of WSN. Wired networks have maximum transmission unit values in the order of kilobytes, and bandwidths of Gigabytes, whereas in WSN the MTU is usually around 100 bytes and maximum data rates around 250 kbps to perhaps 1 Mbps. The amount and rate of information sent by nodes to the central controller in a WSN should perhaps hold the same proportion as in wired networks. However, this may reduce the functionality and consistency of the central model and novel approaches for maintaining the network state will be required.

Distributed State Management. Running multiple SDN controllers physically distributed is necessary to ensure a responsive, scalable and reliable system. The challenge is ensuring that these physically distributed controllers act together to form a logically centralized one, to allow network refactoring through the control plane based on a global network view. The Ethane/Sane project [6], which is a predecessor to OpenFlow, proposed full replication across multiple controllers with a weakly synchronized state. ONIX [21], which was built on the work of Ethane, provides a general framework for dealing with distributed OpenFlow controllers and introduces the idea of a network information state shared by controller replicas. FlowVisor [26] supports multiple controllers by slicing the network resources and assigning control to one controller over the slice. HyperFlow [29] on the other hand, proposes a publish-subscribe mechanism where each controller selectively publishes events that are related to network state changes, and replica controllers use these events to construct the overall network state. More recently, Open Network Operating System (ONOS) [1] offers an open-source controller that also uses a publish-subscribe model to enable controllers to behave as a single logical entity. Other controllers that provide a distributed SDN architecture include Kandoo [11], DISCO [23] and ElastiCon [9]. The challenge of creating a logically centralized controller that integrates with the control of the core network for a future Internet of Things will introduce new scalability issues in terms of the number of devices connected and the need to be able to efficiently respond to the rate of change in network state from nodes (which are unreliable and could also be mobile). These are non-trivial problems that need to be addressed.

4 Related Work

In the following we present key recent work on both the application of software defined networking to wireless sensor networks and non-SDN based centralised management approaches in WSN.

4.1 Recent Work in Adapting SDN to WSN

To date rather limited research in the area of Software Defined Wireless Sensor Networks (SD-WSN) has been published in the literature. As indicated in the introduction, the majority of these proposals is disconnected from previous WSN research and focuses on issues of SDN that are specific to the core and enterprise network domains, without consideration for impact on WSN resources. With few exceptions, the papers do not provide results and overlook what is to these authors the most important problem in porting SDN to WSN: that the unreliable links, combined with the low data rate and energy constraints can lead to an incomplete or *inconsistent* network model.

Sensor OpenFlow [18] and SDWN [7] are two early adopters of SDN in WSN. Both papers propose architectures, highlighting potential issues such as what fields to use for matching rules and how to include additional functionality such as in-network processing. TinySDN [30] presents a similar architecture, where the CTP routing protocol is used to build the control path. Some results for overhead and latency as compared to basic CTP are presented; however, CTP only provides upward routing (nodes to sink) so it is unclear how the solution sends forwarding rules to individual nodes. The authors of SDWN have returned recently with SDN-WISE [10], a state-full WSN SDN controller that further develops the in-network processing capability. This paper also provides evaluation of latency and reliability, however the results are not related to control decisions but only to data packets, therefore the consistency of the central model is not addressed. Other approaches are by Qin *et al.* [24] that focuses on flow scheduling using a genetic algorithm and network calculus, and Jacobsson and Orfanidis [13], who propose an architecture where each node has a local controller.

4.2 Non-SDN Centralised Management Approaches

One of the key aspects of SDN is centralised network control. While Sect. 4.1 presented the related work in the field of applying SDN to WSNs, there are also significant contributions in the area of centralized routing and control for WSN, which none of the work on SDN for WSN considered. The following section highlights some of the main works in this area.

Estrin et al. in [22] were the first to point out the short-comings of distributed control and proposed instead the return to logically centralised control. Central control can be exerted over all the nodes at once, by disseminating commands, such as in SORA [19], where the network is seen as a price-driven marketplace. More fined grained approaches that target individual nodes also exist. For example the CentRoute protocol [27] exploits a multi-tiered network with devices of heterogeneous performance, where low-capability devices are "herded" into "flocks" by higher-performance devices, the "shepherds". Another example is the Flexible Control Protocol (FCP) which is used in the Koala [21] protocol suite. Both CentRoute and FCP, (re)compute the network model through periodic, bulk, downloading of data from the entire network and in order to save power the network lies dormant (from the point of view of the radio) for long periods of time and is then woken up by the base station and queried for new data. Both protocols assume a static topology and the centralised model built is used to install network routes and in the case of FCP also assigns radio channels and schedules the wake-up times of nodes along paths. It is important to note that Koala was found as the closest solution to SDN in WSN, even mentioning the separation of control and data planes. The paper also provides some results of the scalability of the central model. However there are several important differences - the central model is static (built on demand for each download session) and it is not used to enforce the optimisation of network parameters. A different, more generic, approach is taken with the Hydro protocol [8]. Hydro was the initial routing protocol for the TinyOS 6LoWPAN stack, only to be replaced later by RPL. Hydro and RPL both build upward and downward routes using beacons propagated from the root, and maintain a partial model of the network in the collection tree root. Hydro builds this global topology database by piggybacking statistics on data packets and addresses the need for efficient setup of peer-to-peer routes within the network, which differentiates it from RPL. To that extent, Hydro could be compared to OpenFlow in SDN where routes can be installed into nodes' forwarding tables which contain a traffic matching rule and a next hop. Hydro achieves good performance and relatively low control overhead even under heavy network failures.

5 Conclusions

In this paper we presented a proposal for applying the concept of software defined networking (SDN) to wireless sensor networks (WSN) and an architecture for software defined wireless sensor networks (SD-WSN). Our proposal is motivated by the need for improved management of large WSN deployments in use cases such as smart buildings and smart cities. WSNs are also a key part of the Internet of Things, which will lead to billions of wireless sensor nodes connected to the Internet over the next decade. As part of the Internet of Things, the concepts of multi-user or multi-tenant WSN is emerging, which will lead to virtualised WSNs similar to what is happening in enterprise or data centre networks. Our architecture is based on a central controller for the sensor network and a proposal for the split of data and control planes in sensor nodes. We also suggested how multiple SD-WSN domains, each controlled by a single logical controller, can be combined into multi-domain SD-WSNs. Implementing SDN for WSN also facilitates network virtualisation and multi-tenancy in a natural manner. In order to enable truly software defined WSN, a number of challenges need to be overcome in order to enable the many opportunities that this concept will facilitate. We highlighted some of these challenges and opportunities. Finally, we provided some related work on SD-WSN but also showed that the concept of centralised control in WSN is not new and that some of the prior work can be leveraged to solve some of the challenges ahead. We are currently working on a prototype implementation of our architecture with specific focus on the control protocols and challenges underlying network virtualisation.

Acknowledgements. This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number 13/IA/1885.

References

1. Introducing onos - a sdn network operating system for service providers. Technical report, ON.LAB
2. Openflowswitch specification. Technical report ONF TS-006, Open Networking Foundation, June 2012
3. Baccour, N., Koubâa, A., Mottola, L., Zuniga, M.A., Youssef, H., Boano, C.A., Alves, M.: Radio link quality estimation in wireless sensor networks: a survey. ACM Trans. Sen. Netw. **8**(4), 34:1–34:33 (2012). http://doi.acm.org/10.1145/2240116.2240123
4. Barrenetxea, G., Ingelrest, F., Schaefer, G., Vetterli, M.: The hitchhiker's guide to successful wireless sensor network deployments. In: Proceeding SENSYS 2008, pp. 43–56. ACM (2008)
5. Blendin, J., Ruckert, J., Leymann, N., Schyguda, G., Hausheer, D.: Position paper: software-defined network service chaining. In: 2014 Third European Workshop on Software Defined Networks (EWSDN), pp. 109–114, September 2014
6. Casado, M., Freedman, M., Pettit, J., Luo, J., McKeown, N., Shenker, S.: Ethane: taking control of the enterprise. SIGCOMM Comput. Commun. Rev. **37**(4), 1–12 (2007)
7. Costanzo, S., Galluccio, L., Morabito, G., Palazzo, S.: Software defined wireless networks: unbridling sdns. In: 2012 European Workshop on Software Defined Networking (EWSDN), pp. 1–6, October 2012
8. Dawson-Haggerty, S., Tavakoli, A., Culler, D.: Hydro: a hybrid routing protocol for low-power and lossy networks. In: 2010 First IEEE International Conference on Proceedings of the Smart Grid Communications (SmartGridComm) (2010)
9. Dixit, A., Hao, F., Mukherjee, S., Lakshman, T.V., Kompella, R.: Towards an elastic distributed sdn controller. In: Proceedings of HotSDN 2013, pp. 7–12. ACM, New York (2013)
10. Galluccio, L., Milardo, S., Morabito, G., Palazzo, S.: Sdn-wise: design, prototyping and experimentation of a stateful sdn solution for wireless sensor networks. In: Proceedings of the 34th IEEE INFOCOM Conference (2015)
11. Yeganeh, S.H., Ganjali, Y.: Kandoo: A framework for efficient and scalable offloading of control applications. In: Proceedings of HotSDN 2012, pp. 19–24. ACM, New York (2012)
12. Heming, W., Tiwary, P.K., Le-Ngoc, T.: Current trends and perspectives in wireless virtualization. In: Proceedings of MoWNet 2013, pp. 62–67 (2013)
13. Jacobsson, M., Orfanidis, C.: Using software-defined networking principles for wireless sensor networks. In: Proceedings of the 11th Swedish National Computer Networking Workshop (2014)
14. Jayasumana, A.P., Han, Q., Illangasekare, T.H.: Virtual sensor networks - a resource efficient approach for concurrent applications. In: Fourth International Conference on Information Technology, ITNG 2007, pp. 111–115, April 2007
15. Langendoen, K., Baggio, A., Visser, O.: Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. In: Proceedings 20th IEEE International Parallel & Distributed Processing Symposium, pp. 155. IEEE (2006)

16. Leontiadis, I., Efstratiou, C., Mascolo, C., Crowcroft, J.: SenShare: transforming sensor networks into multi-application sensing infrastructures. In: Picco, G.P., Heinzelman, W. (eds.) EWSN 2012. LNCS, vol. 7158, pp. 65–81. Springer, Heidelberg (2012)
17. Li, Y., Chen, C., Song, Y., Wang, Z.: Real-time qos support in wireless sensor networks: a survey. In: 7th IFAC International Conference on Fieldbuses & Networks in Industrial & Embedded Systems-FeT 2007 (2007)
18. Luo, T., Tan, H.-P., Quek, T.Q.S.: Sensor openflow: enabling software-defined wireless sensor networks. IEEE Commun. Lett. **16**(11), 1896–1899 (2012)
19. Mainland, G., Parkes, D., Welsh, M.: Decentralized, adaptive resource allocation for sensor networks. In: Proceedings of the 2Nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2, NSDI 2005, pp. 315–328. USENIX Association, Berkeley (2005)
20. Markham, A., Trigoni, N.: Discrete gene regulatory networks (dgrns): a novel approach to configuring sensor networks. In: 2010 Proceedings IEEE INFOCOM, pp. 1–9. IEEE (2010)
21. Musaloiu, R., Liang, C.J.M., Terzis, A.: Koala: ultra-low power data retrieval in wireless sensor networks. In: International Conference on Information Processing in Sensor Networks, IPSN 2008, pp. 421–432 (2008)
22. Paek, J., Greenstein, B., Gnawali, O., Jang, K., Joki, A., Vieira, M., Hicks, J., Estrin, D., Govindan, R., Kohler, E.: The tenet architecture for tiered sensor networks. ACM Trans. Sen. Netw. **6**(4), 34:1–34:44 (2010)
23. Phemius, K., Bouet, M., Leguay, J.: Disco: distributed multi-domain sdn controllers. In: 2014 IEEE Network Operations and Management Symposium (NOMS), pp. 1–4, May 2014
24. Qin, Z., Denker, G., Giannelli, C., Bellavista, P., Venkatasubramanian, N.: A software defined networking architecture for the internet-of-things. In: Proceedings of the IEEE Network Operations and Management Symposium (NOMS) (2014)
25. Rubio-Loyola, J., Galis, A., Astorga, A., Serrat, J., Lefevre, L., Fischer, A., Paler, A., Meer, H.: Scalable service deployment on software-defined networks. IEEE Commun. Mag. **49**(12), 84–93 (2011)
26. Sherwood, R., Gibb, G., Yap, K., Appenzeller, G., Casado, M., McKeown, N., Parulkar, G.: Flowvisor: a network virtualization layer. OpenFlow Switch Consortium. Technical report (2009)
27. Stathopoulos, T., Girod, L., Heidemann, J., Estrin, D.: Mote herding for tiered wireless sensor networks. Center for Embedded Network Sensing (2005)
28. Stuckmann, P., Zimmermann, R.: European research on future internet design. IEEE Wirel. Commun. **16**(5), 14–22 (2009)
29. Tootoonchian, A., Ganjali, Y.: Hyperflow: a distributed control plane for openflow. In: Proceedings of the 2010 internet network management conference on Research on enterprise networking, USENIX Association, p. 3 (2010)
30. Trevizan de Oliveira, B., Borges Margi, C., Batista Gabriel, L.: Tinysdn: enabling multiple controllers for software-defined wireless sensor networks. In: IEEE Communications (LATINCOM), pp. 1–6, November 2014
31. Wang, X., Krishnamurthy, P., Tipper, D.: Wireless network virtualization. In: International Conference on Computing, Networking and Communications (ICNC), pp. 818–822, January 2013