Resource sharing between neighboring nodes in heterogeneous Wireless Sensor Networks

Alessandro Pagani
DEI, University of Bologna, Italy
Email: alessandropagani.90@gmail.com

Konstantin Mikhaylov
Center for Wireless Communications
University of Oulu, P.O. BOX 4500, Oulu, Finland
Email: konstantin.mikhaylov@ee.oulu.fi

***Abstract*—In this paper the Wireless Resource Sharing Method (WRSM) for wireless sensor and actuator networks is introduced. The proposed method aims to enable distributed and dynamic discovery, negotiation and sharing of tasks and resources (e.g., hardware accelerators, communication interfaces or memory) between the neighboring nodes in heterogeneous and multi-tasked IoT-like network environment. The paper introduces the WRSM, discusses challenges faced and technical solutions developed for implementing it in practice, and reports results of the first hardware tests. The presented results confirm feasibility of the proposed method, highlight some limitations of the current solution and point out directions of further work.**

I. Introduction

Wireless Sensor and Actuators Networks (WSANs) of the future are envisioned to become an integral part of the Internet of Things (IoT), which is seen as a global networking society, where people, machines, data, services and applications are tightly integrated together by the means of information and telecommunications technology. This transformation will introduce significant changes in the operation environment of the WSANs thus revolutionizing the design and dramatically changing the requirements for these systems. The first and the most fundamental change is the requirement for the nodes to cope in the heterogeneous and multi-tasked environment. Traditional sensor nodes are designed to perform a pre-defined set of tasks using a static pool of resources, and implement very limited support for cooperation between the nodes. The future WSANs are envisioned to be composed of myriads of nodes having different structures and hardware components, processing and storage capabilities, communication interfaces, software applications and available services. All those form a set of *resources* available to a particular node. In order to operate efficiently, the nodes need to be aware of each other's resources and tasks, and have the mechanisms to share those.

Therefore, in the current paper we focus on this problem and propose the novel Wireless Resource Sharing Method (WRSM) for future WSANs, discuss its design and report details of its implementation. The major objectives of the proposed solution are:

- enable decentralized discovering and requesting resources for neighboring nodes;
- enable decentralized reserving of resources and transferring the tasks between the neighboring nodes.

The paper is organized as follows. Section II overviews the previous works in the field. Section III describes the proposed method and the format of used frames and resource descriptors. Sections IV and V discuss in details the WRSM implementation and present the results of initial tests respectively. Finally, Section VI concludes the paper and summarizes the results.

II. Related work

One of the first attempts to target the problem of resource sharing in wireless networks was done by McKnight et al. in [1]. The authors analyzed challenges, requirements and procedures required for enabling resource sharing in wireless grids. Few of the researchers focused on the problem of resource discovery in wireless networks. e.g., categorization of resource discovery models was done in [2] and a Dynamic Resource Discovery protocol for clustered WSNs was proposed in [3]. Oteafy and Hassane studied the problem of optimal mapping the available resources in multi-application WSNs in [4].

In [5] the authors demonstrated how interoperable mobile agents might be used for integrating IoT and WSNs and transferring the tasks between the nodes and smartphones. The mechanisms for cooperative and reliable data storing in WSNs have been proposed in [6]. The possibility of forming a cluster and distributing the low-density parity-check (LDPC) image encoding task among the nodes was discussed in [7].

Note that in the majority of these works authors used analytical methods to approach the problem and simulations to confirm the feasibility of their approach. Meanwhile, in this work, we approach the problem from the practical point of view and investigate how the resource sharing might be implemented in practice, keeping in mind the specifics and natural limitations of WSAN systems.

III. Wireless Resource Sharing Method

A. Proposed Mechanism

Consider a scenario where a sensor node (e.g., node A) either comes in contact with other nodes of a Wireless Personal Area Network (WPAN) or is already part of a WPAN. In both cases, all nodes have information about their structure and host description of their resources. Namely, we assume that the nodes are aware about the available sensors (temperature, humidity...), actuators (switches, appliances...), functionality extenders (localization engines, security accelerators...), communication interfaces (IEEE 802.15.4, WiFi, Bluetooth ...) or power sources (power grid, battery, energy harvesting...) and data storage drives. Once node A runs out of its available resources or faces a task which cannot be accomplished efficiently using node's capabilities, it can check if the desired functionality is available within the heterogeneous environment around it. For this, node A (referred to as *Applicant*) initiates WRSM communication broadcasting a *Resource Request* to adjacent nodes (Figure 1a). The message specifies the type of required resource and the planned period of resource exploitation T (or the time within which a specific operation needs to be completed). The format of resource descriptors and WRSM packet format are discussed in Subsections III-C and III-B

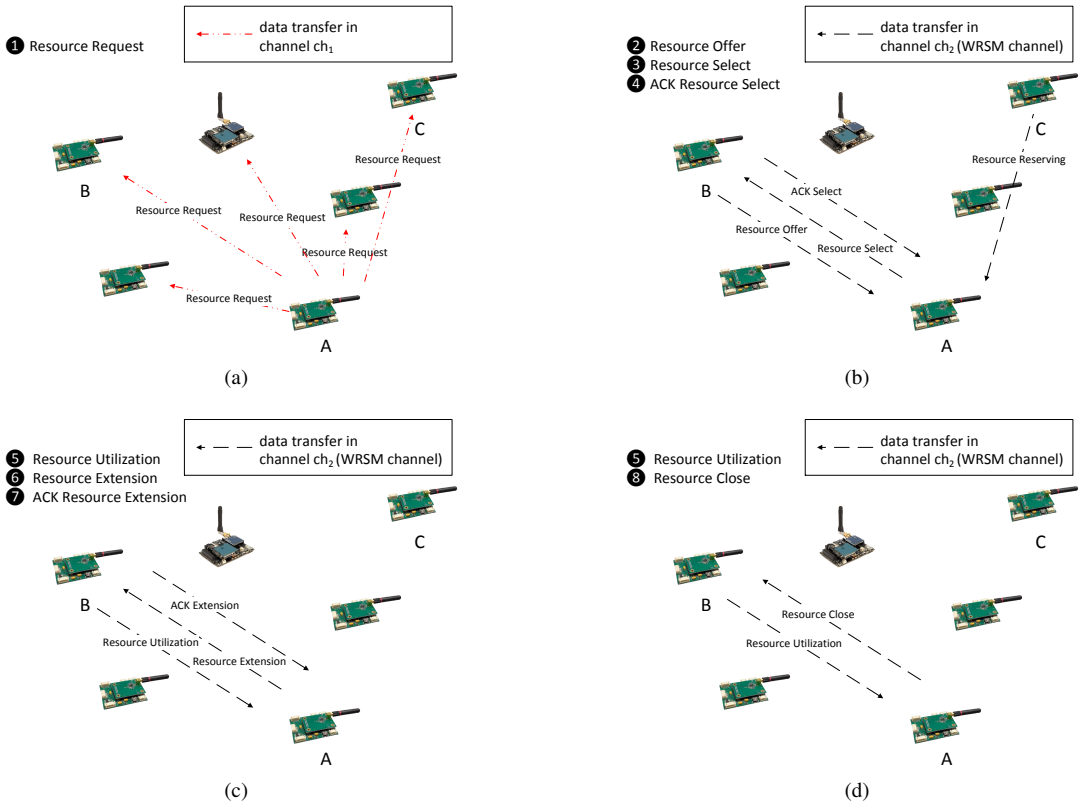


Fig. 1. WRSM procedure

respectively. Note that we assume that the used communication technology already has mechanisms enabling reliable packet transmission (e.g., automatic packet re-transmission).

The nodes which have received the *Resource Request* message check the availability of requested resources in their pool and decide if they can fulfill the request. While making the decision potential *Supplier* takes into account its active applications, power supply type and remaining energy. Nodes that have the requested resources and are capable to share these (e.g. nodes B and C) reply to the *Applicant* with a *Resource Offer* message (Figure 1b). Note, that in the case the used wireless communication technology and hardware support this, the WRSM data transfers might be executed in a frequency channel (e.g., *WRSMchannel*=ch_2) which differs from the one used by WPAN (i.e., ch_1) or even using another communication technology. This will reduce the communication overhead and help avoiding packet collisions. The frequency of ch_2 can be chosen based e.g., on spectrum sensing.

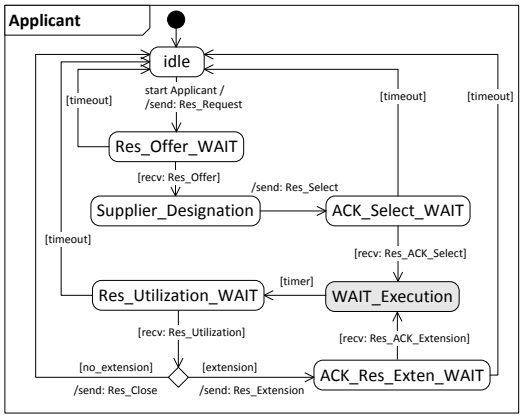
After sending the response, potential *Suppliers* switch to receive. The *Applicant* designates the selected *Supplier* by replying with a *Resource Select* message. In Figure 1b the *Applicant* chooses node B. The policy used to choose a Supplier can be one of the following:

- First-Received-First-Chosen: the *Applicant* chooses the first node, which replies to its Resource Request;
- based on the received power: the *Applicant* listens for incoming message for a certain period and selects the *Supplier* based on comparison of the received radio signals strength (e.g. the closest node);
- based on Supplier's available resources: the *Applicant* designates the *Supplier* based on the amount of resources it can share (e.g., memory, maximum transmission power) or on its remaining energy.

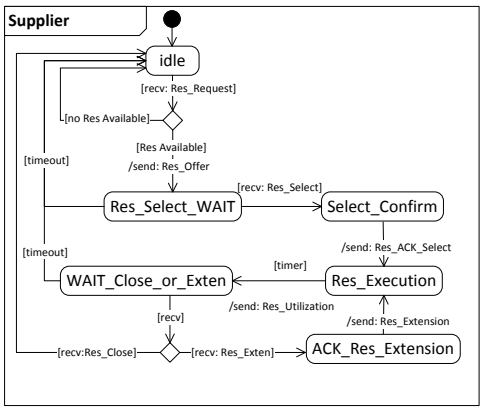
Note, that in the case if an *Applicant* does not receive any *Resource Offer* in a specified period of time (e.g., no nodes can satisfy the requirements) it stops WRSM execution. Also each potential *Supplier* arms a timeout which will enable node to switch off if not selected by *Applicant*.

It is obvious, that the efficiency of proposed WRSM strongly depends on the number of nodes in the area, their duty cycles, and the diversity of their resource pools and tasks. Therefore, in the current paper we assume that the network is dense enough or includes always-on nodes (e.g., mains-powered) which might act as potential *Suppliers*. This assumption sounds quite reasonable e.g., for such applications and environment as Smart City or WSANs in public places.

Depending on the requested resource the *Resource Select* message may contain data for the *Supplier* (e.g., data to be compressed or stored). An *ACK Resource Select* message acknowledges the designation. Then two nodes switch back to channel ch_1 and proceed with other tasks, whilst *Supplier* executes the requested WRSM task in background. At the end of the specified period of time T both nodes switch again to *WRSMchannel*. *Supplier* B sends to *Applicant* A the *Resource Utilization* message (Figure 1c). The message may contain a data block depending on the type of operation executed (e.g. monitoring results, elaborated data). In addition, the *Resource Utilization* message indicates the capability of the *Supplier* to repeat the previous WRSM operation. If the *Supplier* has specified that it can make the resource available again, the *Applicant* may reply with a *Resource Extension* message (Figure 1c)



(a)



(b)

Fig. 2. WRSM state machine diagrams

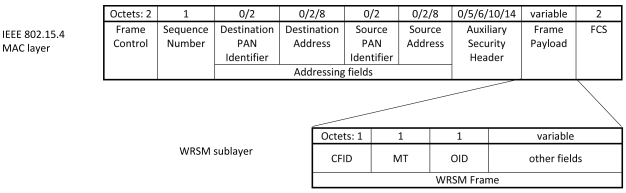


Fig. 3. MAC frame format

confirming successful *Resource Utilization* message reception and asking to repeat the operation. Just as the *Resource Select* case, this message may contain a data block. An *ACK Resource Extension* message acknowledges the extension (Figure 1c) and the described above procedure repeats. If the *Applicant* does not require or if the *Supplier* can not provide the resource any longer, the *Applicant* replies with *Resource Close* message confirming reception of the *Resource Utilization* message and closing the WRSM session (Figure 1d). Figures 2a and 2b depict the WRSM state machine diagrams for the *Applicant* and *Supplier*.

B. Frame Format

The WRSM frame is encapsulated in the frames of the used communication technology at link or network layers and composes a sequence of fields in specific order. This enables to use the available addressing, segmentation and security mechanisms provided by the communication technologies. The example WRSM frame format for operating on top of IEEE 802.15.4 MAC is illustrated in Figure 3.

The first three octets of the frame are common to each WRSM frame and are used for MAC Command Frame Identifier (CFID), the Message Type (MT) and the Operation ID (OID). Every WRSM message is identified by the first octet equal to 0x0A that is the first available value (reserved for future use) in the IEEE 802.15.4 MAC Command Frame Identifier field according to the standard [8]. The Message Type field defines the WRSM frame type (resource request, resource select, etc.). The Operation ID field identifies the resource request context. In combination with the *Applicant*'s MAC address, the Operation ID identifies every specific resource request session and is replicated in each following WRSM message regarding the same request.

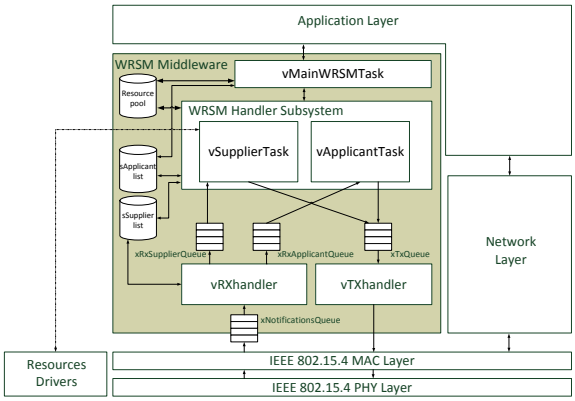


Fig. 4. Software Architecture

C. Resource Description

Collection of information regarding resources and negotiation of tasks inevitably requires a solution for common resource notation. Given the heterogeneous nature of WSANs, the resource specification must be standard and interchangeable in order to ensure interoperability among the nodes. There are two major approaches to this problem. The first option is to use a standard model for data representation based, e.g., on XML (e.g., Resource Description Framework (RDF) [9] or Resource Description Language (RDL) [10]). The major drawback of this approach is significant overheads and the need for effective compression algorithms. The other option is to use a non-standard flexible ad-hoc model. Despite the related interoperability issues, this approach makes the data packets more compact. For WRSM we have chosen this option, but due to the lack of space the details are not presented in this paper.

IV. WRSM IMPLEMENTATION

To test the WRSM we have used the prototype modular WSAN/IoT platform introduced in [11]. The platform is a perfect choice since it supports automatic discovery, identification and Plug-and-Play (P&P) connection of the modules featuring different supply source, processing units, wireless transceivers, sensors, actuators, etc. The core board is equipped with the STM32F207 ARM-based 32-bit microcontroller. Also

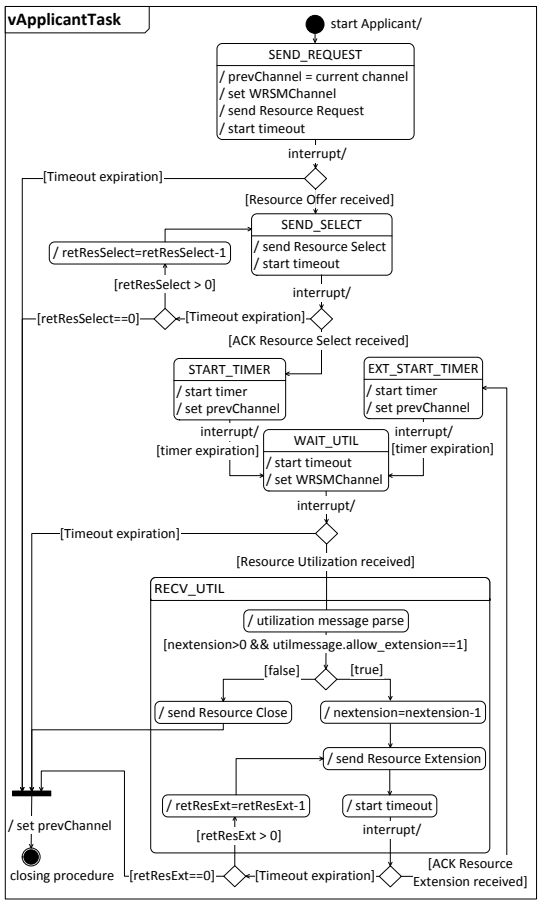


Fig. 5. Applicant State Machine [1]

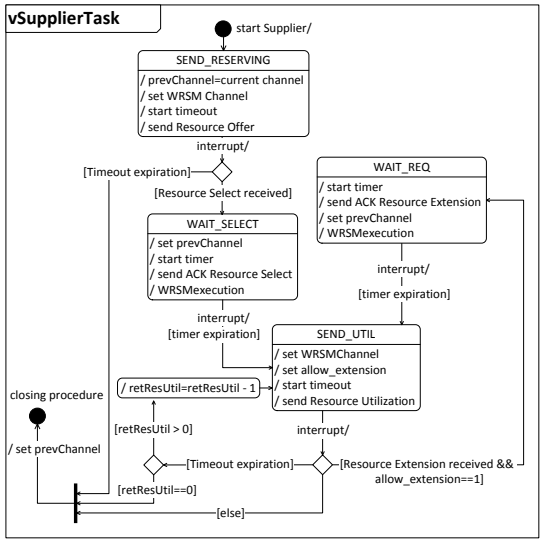


Fig. 6. Supplier State Machine [2]

for the tests were used peripheral modules encapsulating the IEEE 802.15.4 compatible Atmel AT86RF233 2.4GHz radio transceiver, battery and USB power modules.

The WRSM is implemented in C language for FreeRTOS environment and integrated in WSAN node middleware. The WRSM middleware is responsible to executing all the operations and functions needed to discover and share resources among the nodes. Structure of the middleware is depicted in Figure 4 and encompasses five software components with specific functionalities, each realized as FreeRTOS task. Arrows show data and command flows between tasks. The data transfers are implemented using OS queues in order to ensure proper synchronization between components of the system. sApplicant and sSupplier are linked lists used to store information about every active point-to-point WRSM connection, and include destination MAC address of the peer, OID, resource description format identifier, shared resources and time of next communication event, WRSM frequency channel and pointers to input and/or output data. Each WRSM middleware task can access sApplicant and sSupplier lists and retrieve the required parameters.

Two different software timer instances (not shown in the figure) are created for each active WRSM session. The former one, named *Communication Timeout*, is used to retransmit a WRSM message or even cancel a session in case of missing response in order to prevent endless waiting. The other one, named *Resource Timer* is used for determining the end of resource exploitation time interval.

The vMainWRSMTask is the task which handles the requests from application layer. A request for new resources first adds a new instance in sApplicant list and then initializes a transmission of Resource Request message. Also the task handles the request for available resources pool update.

vApplicantTask and vSupplierTask form the WRSM handler subsystem. The state machines describing those are detailed in Figures 5 and 6 respectively. The three queues are used to handle transmission and reception of the data from the communication technology and/or transceiver specific drivers, which are implemented in vTXhandler and vRXhandler tasks. Note, that the proposed architecture is quite flexible and enables each node to play a role of *Supplier* and *Applicant* or even combine those roles in different WRSM sessions. Although the proposed system can easily handle multiple simultaneous requests, in the current implementation a *Supplier* can not reserve a particular resource to more than one *Applicant* at a time.

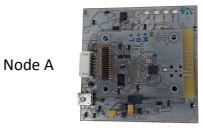
V. Evaluation

For our practical experiments we targeted the following scenario. A battery-powered node A measures the environment and periodically reports the results to remote sink. Mains-powered node B is located near node A and has compression accelerator implementing specific compression algorithm. To avoid memory overfill on node A, node A starts WRSM and requests node B to compress its data designating the results to be returned in 3 seconds. In this context, node A operates as *Applicant* and node B acts as *Supplier*. The process is repeated periodically.

In order to monitor the over-the-air traffic we used the combination of CC2531 USB dongle and packet sniffer software from Texas Instruments. An example of WRSM session captured during experiment is illustrated in Figure 8. Node

[1] *nextension* is the maximum number of extensions the *Applicant* may request for WRSM session, *retResSelect* and *retResExt* are the maximum number of *Resource Select* and *Resource Extension* retransmissions in case of failure

[2] *retResUtil* is the maximum number of *Resource Utilization* message retransmissions in case of failure



Applicant node

Radio settings

Short MAC	0x308A
PAN ID	0xBEEF
PAN Channel	11 (i.e., 0x0B hex)
WRSM Channel	15 (i.e., 0x0F hex)

Resource pool
- 2.4GHz IEEE 802.15.4 radio transceiver
- Sensor 1
- Battery powered

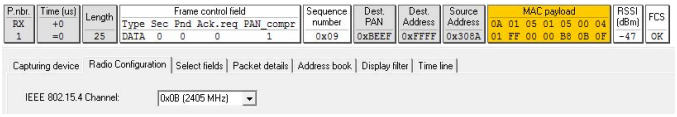
Supplier node

Radio settings

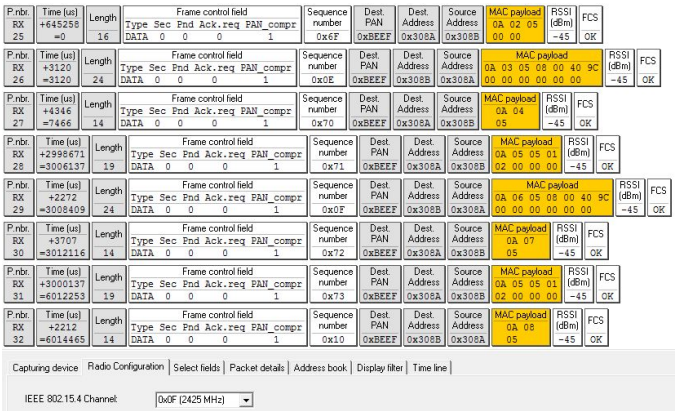
Short MAC	0x308B
PAN ID	0xBEEF
PAN Channel	11 (i.e., 0x0B hex)
WRSM Channel	15 (i.e., 0x0F hex)

Resource pool
- 2.4GHz IEEE 802.15.4 radio transceiver
- Sensor 2
- Compression accelerator
- Mains powered

Fig. 7. WRSM implementation scenario



(a)



(b)

Fig. 8. Example of over-the-air traffic during a WRSM session

A sends a *Resource Request* (packet 1, subfigure 8a) in PAN channel and receives an offer from node B (packet 1, subfigure 8b) in WRSM channel. The data to be compressed are transferred in *Resource Select* (packet 2, subfigure 8b) and acknowledged by node B (packet 3, 8b). The compressed data are returned to node A in *Resource Utilization* (packet 4, subfigure 8b) after 3 seconds. Then the session is extended (packets 5-7, subfigure 8b) for one more round and finally closed by *Resource Close* (packet 8, subfigure 8b).

As one can see, for the proposed protocol only 3 packets have to be transmitted for enabling node A access the requested resource, while e.g. the protocol proposed in [3] requires at least 5 packets [3] (namely: energy advertisements from both nodes, meta-data, query and reply, resource utilization message). Figure 8 reveals that for the discussed scenario (i.e., 8 bytes of resource-data) resource access was accomplished in less than 10 ms (compare, e.g. with around 800 ms resource access procedure reported in [5]).

[3]since the packet and data formats for [3] are not reported, it is impossible to assess the actual time

VI. Conclusion and Future Work

In this paper we have proposed a novel Wireless Resource Sharing Method (WRSM) for WSANs and reported the details of its practical implementation. The presented method enables the devices to dynamically access and use the resources, services and applications available on neighboring nodes in heterogeneous network environment. We have introduced few technical solutions (e.g., use of designated radio channel for WRSM) and discussed in details the software architecture required for implementing WRSM in practice. The presented results of practical test using the novel modular WSAN platform confirm the feasibility of the proposed mechanism.

To the best of authors' knowledge, the current paper is one of the first attempts of practical implementation of resource sharing between the nodes of a WSAN. Nonetheless, the reported work is intended as a proof of concept and has multiple limitations. First of all, in this work we have not investigated the networking aspects and the performance of WRSM in multinode environment. Although the restriction of WRSM operation to only single-hop case, limited traffic and capability of using different frequency channels make us expect that the protocol has potential for scaling up. Nonetheless, further studies are required to confirm this. Second, our implementation does not take into account the sleep cycles of wireless nodes. Third, the nodes can specify only one resource in each Resource Request. Fourth, the security and reliability aspects were not considered in our implementation. In future we will consider targeting these issues and relaxing some of made assumptions, as well as studying the networking aspects of resource sharing more deeply.

References

[1] L. W. McKnight *et al.*, "Guest editors' introduction: Wireless grids–distributed resource sharing by mobile, nomadic, and fixed devices," *IEEE Internet Comput.*, vol. 8, no. 4, pp. 24–31, Jul-Aug 2004.

[2] M. Hijab and D. Avula, "Resource discovery in wireless, mobile and ad hoc grids - issues and challenges," in *Proc. 13th Int. Conf. Advanced Commun. Technology (ICACT)*, Feb 2011, pp. 502–505.

[3] S. Tilak *et al.*, "Dynamic resource discovery for sensor networks," in *Proc. Embedded and Ubiquitous Computing–EUC 2005 Workshops*. Springer, Dec 2005, pp. 785–796.

[4] S. Oteafy and H. S. Hassanein, "Resource re-use in wireless sensor networks: Realizing a synergetic internet of things," *J. Commun.*, vol. 7, no. 7, pp. 484–493, Jul 2012.

[5] T. Leppänen *et al.*, "Mobile agents for integration of internet of things and wireless sensor networks." in *Proc. SMC*, 2013, pp. 14–21.

[6] K. Piotrowski *et al.*, "tinydsm: A highly reliable cooperative data storage for wireless sensor networks," in *Proc. Int. Symp. Collaborative Technologies and Syst. (CTS '09)*. IEEE, 2009, pp. 225–232.

[7] P. N. Huu *et al.*, "Distributed image encoding scheme using ldpc codes over gf (q) on wireless sensor networks," in *Proc. 4th Int. Conf. Intelligent Networking and Collaborative Syst. (INCoS)*. IEEE, 2012, pp. 198–205.

[8] *IEEE Standard for Local and metropolitan area networksPart 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*, IEEE Std. 802.15.4, 2011.

[9] D. Wood *et al.*, "RDF 1.1 concepts and abstract syntax," W3C, W3C Recommendation, Feb 2014, http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/.

[10] A. Santos *et al.*, "Resource description language: A unified description language for network embedded resources," *Int. J. Distributed Sensor Networks*, vol. 2012, 2012, Copyright ©2012 Andr C. Santos et al. doi: 10.1155/2012/860864.

[11] K. Mikhaylov and M. Huttonen, "Modular wireless sensor and actuator network nodes with plug-and-play module connection," in *Proc. IEEE SENSORS '14*, Nov 2014, pp. 470–473.