A Framework for Ontology Provisioning in Virtualized Wireless Sensor Networks

Rifat Jafrin*, Imran Khan†*, Jagruti Sahoo* and Roch Glitho*
*Dept. CIISE, Concordia University, H3G 2W1, Montreal, Canada
Email: {r_jafri, jsahoo} @encs.concordia.ca, glitho@ciise.concordia.ca
†Schneider Electric Industries SAS, 38TEC, 38050, Grenoble Cedex 9, France
Email: imran@ieee.org

***Abstract*—Virtualization in Wireless Sensor Networks (WSN) allows an efficient resource usage through the sharing of the same WSN physical infrastructure by multiple applications. Semantic applications are gaining more and more momentum. However, provisioning them in Virtualized WSNs (vWSNs) remains a big challenge; the data collected by the virtual sensors needs to be annotated in-network and for this annotation, an ontology needs to be provisioned, i.e. developed, deployed and managed. This paper proposes a framework for ontology provisioning in vWSNs. The framework comprises of an ontology provisioning center, an ontology enabled vWSN and an ontology provisioning protocol that enables the interactions between the provisioning center and the ontology enabled vWSN. To the best of our knowledge this is the first effort to provide such support in vWSNs. We have built a prototype to evaluate the performance of the framework and also present simulation results of the ontology provisioning protocol.**

***Index Terms*—Wireless Sensor Networks; WSN; Semantic web; WSN Virtualization; Ontology; Overlays**

I. Introduction

Virtualization aims at the efficient usage of Wireless Sensor Networks (WSN) deployments, by enabling the co-existence of multiple applications on the same Virtualized WSN (vWSN) [1]. Semantic applications are now ubiquitous and used in diverse application domains [2][3]. Integrating semantic applications in vWSN enriches sensor data and enables more rich and interactive applications. The same set of annotated data from a vWSN can be used by multiple applications (e.g. a weather application and a pollution monitoring application can use same data to infer additional implicit knowledge).

However, several challenges still need to be tackled to make semantic applications a reality in the vWSN environment: The data collected by the sensors needs to be annotated in-network at the (Infrastructure-as-a-Service) IaaS level to ease the application provisioning process. Furthermore, IaaS specific ontology needs to be provisioned, i.e. developed, deployed and managed in order to enable the in-network data annotation. We have already tackled the first challenge by proposing an in-network data annotation architecture [4]. We developed an ontology called *base ontology* to store the concepts related to the deployed sensors. However, it was assumed that the base ontology has already been provisioned in the vWSN.

This paper tackles the ontology provisioning challenge in vWSNs, which to the best of our knowledge has not been addressed so far. It proposes a framework for the ontology provisioning, which consists of an *Ontology Provisioning Center* (OPC), *an ontology enabled vWSN*, and an *Ontology Provisioning Protocol* (OPP).

The OPC allows vWSN providers to easily develop and manage the base ontology. This is achieved using a web-based GUI application for intuitive ontology creation and management even for a novice user.

The ontology enabled vWSN is realized by proposing a new architecture that facilitates the storage and utilization of the base ontology in the vWSN. The concept of overlays is used where several dedicated functional entities perform different roles to allow annotation of the sensor data in real-time. Because vWSNs are inherently resource-constrained, only capable nodes are selected to perform these roles. However, the proposed architecture is independent of any protocol or algorithm to make such selection.

The OPP provides a lightweight mechanism to deploy the base ontology in resource-constrained WSN. It enables interactions between the ontology provisioning center and the ontology enabled vWSN. The protocols uses control flooding to identify the potential candidates among all the nodes in the network that fulfil the requirements in terms of memory and battery power to store the ontology.

The proposed architecture uses a centralized approach to select and deploy the base ontology. The architecture is flexible enough to use either proactive or reactive approach to deploy base ontology on the selected nodes in the vWSN.

The rest of the paper is organized as follows: In Section II, we present a motivating scenario along with a set of requirements. Section III is devoted to the OPC. The ontology enabled vWSN is discussed is section IV. Section V describes the OPP. The prototype implementation and simulation results of OPP are presented in Section VI. Section VII provides a critical review of the related work whereas Section VIII concludes the paper and outlines future work.

II. Motivating Scenario and Requirements

Let us consider that, in initial stages, a WSN IaaS provider deploys heterogeneous sensors in a given geographic area to detect different physical phenomena. The IaaS provider uses sensors of brands kits for the initial deployment. Some of these sensors may have multiple sensing capabilities; e.g. a *Java SunSpot* sensor can sense temperature, humidity and light

using its three on-board sensors. The data from the sensors is annotated using the base ontology. Let us now assume when the WSN IaaS is in the operation stage, new sensors with new capabilities (e.g. wind speed) are introduced in the infrastructure. In such a situation, the base ontology should be updated by adding the corresponding concepts of the newly deployed sensors and be re-deployed.

We have derived few requirements from the above mentioned scenario. *First*, there is a necessity for a new framework through which the WSN IaaS provider can develop, deploy and manage the base ontology. We named this framework ontology provisioning center. A base ontology needs to be provisioned at the initial stage to enable applications provisioning. *Second*, the base ontology will be extended when new sensors are added to the infrastructure. *Third*, the new framework should be user-friendly and allow the WSN IaaS provider or any other novice user to interact with it without much technical knowledge or protocol details. *Fourth*, it should also allow distributed ontology provisioning, due to the very nature of vWSN. *Fifth*, it should be platform independent to be implemented by any standard technologies. *Sixth* requirement is, it should have the proper mechanism to interact with vWSN. *Finally*, the proposed solution should also be scalable.

III. Ontology Provisioning Center

Figure 1 shows the components of proposed Ontology Provisioning Center (OPC).

The WSN IaaS provider interacts with the OPC using the GUI application to create new concepts. The application interface is shown in Figure 2. It is assumed that the new concepts are created after the deployment of the sensors since these concepts will be directly related to the capabilities of these sensors. All concepts are stored in a data repository to allow easy storage, modification or removal. The ontology creation module is responsible for generating the base ontology by using the concepts stored in the data repository. We develop a set of mapping rules and use the ontology development language to create the base ontology. Finally, the base ontology is stored in an ontology repository.

The base ontology creation process is illustrated in Figure 3 and is described as follows: In *first* step, the IaaS provider defines the concepts according to the deployed sensors. For each concept, information such as sensor type, output type, output unit and observed property is provided. In *second* step, the application, checks for duplicate concepts to ensure that there is no ambiguity or duplication of concepts. In *third* step, the new concepts are stored in the data repository. These new concepts are incorporated to a default ontology. In this work, we use concepts from the standard SSN ontology [5] as the default ontology and extend it with the new concepts specific to the type and capabilities of the sensors we used in this work, like

$$Base\,Ontology = concept_{SSN} + concept_{temperature} + concept_{humidity} + ... + concept_k$$

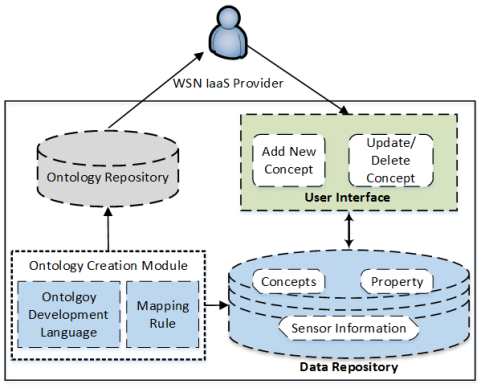

Fig. 1. Ontology Provisioning Center

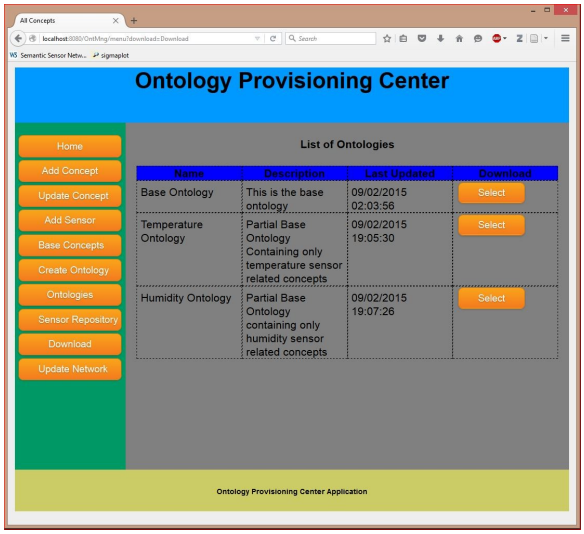

Fig. 2. Application to Develop and Manage Base Ontology

In *fourth* step, Ontology Creation Module (OCM) requests for the concepts in the data repository and uses them in the *fifth* step to create Base Ontology. In *sixth* step, ontology creation module checks locally for the default ontology. If not found, a request message is sent to the Ontology Repository in the *seventh* step and receives the default ontology in *eighth* step. OCM incorporates the new concepts to the default ontology by creating a parent-child relationship between the concepts from SSN ontology and the new concepts (child concepts) using a set of mapping rules. The value of property and domain range for each concept is updated to reflect the new additions or modifications of the concept.

Finally, the Base Ontology is sent to the ontology repository from where it is ready to be deployed. All steps except the first are performed automatically, i.e. without any user input.

IV. Ontology Enabled vWSN

Figure 4 presents the proposed architecture for the ontology enabled vWSN which is an extension of the architecture presented in [4].

The WSN IaaS Manager is a new entity added to the Physical Layer in our previous architecture. It is a capable

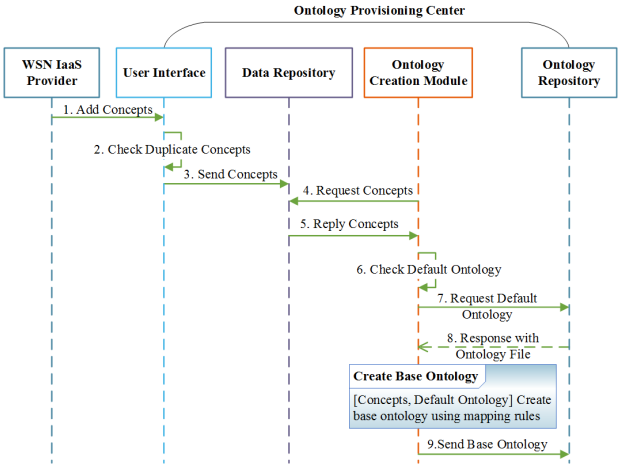

Fig. 3. Base Ontology Creation

node managing the deployed WSN IaaS. The Virtual Sensor Access Layer has two new functional entities residing in the ontology overlay: The WSN Infrastructure Manager (WIM) and Ontology Manager (OM). The WIM is the logical entity of the centralized WSN IaaS Manager present at Physical Layer.

WIM performs the following two tasks: *(a)* Selecting the capable nodes to act as either OMs or OAs, using a certain mechanism (e.g. by executing an algorithm [6]) and *(b)* deploying the ontology concepts to the selected nodes by using our proposed ontology deployment protocol presented in Section V-A. OM is another new entity added to the ontology overlay, to hold the complete base ontology and send it to the OAs upon request. To tackle the node failure problem, we duplicate the base ontology among multiple OMs. Capable nodes (e.g. GTO nodes and Type B sensors) are typically selected by WIM to act as OMs.

V. Ontology Provisioning Protocol

Our Ontology Provisioning Protocol (OPP) tackles the Base Ontology deployment phase. It is described in the following subsection. Afterwards an illustrative scenario along with a sequence diagram in Figure 5 illustrates all the steps involved in the deployment:

A. Ontology Provisioning Protocol

This section discusses our proposed OPP for deploying the base ontology in overlay network. OPP is used to exchange messages *(i)* from the ontology provisioning center to the WIM in the ontology overlay (Deploy Ontology and Ontology Deploy Request) and *(ii)* from WIM to other nodes in the ontology overlay. When the base ontology is developed, it is sent to WIM by the ontology provisioning center using the Deploy Ontology message. WIM can also periodically send a request to OPC for the updated base ontology. As the ontology overlay is responsible for holding the base ontology, the rest of the messages are exchanged between the nodes in the ontology overlay. Table I summarizes the messages exchanged for the deployment of base ontology whereas Table II summarizes the content of the discovery request message.

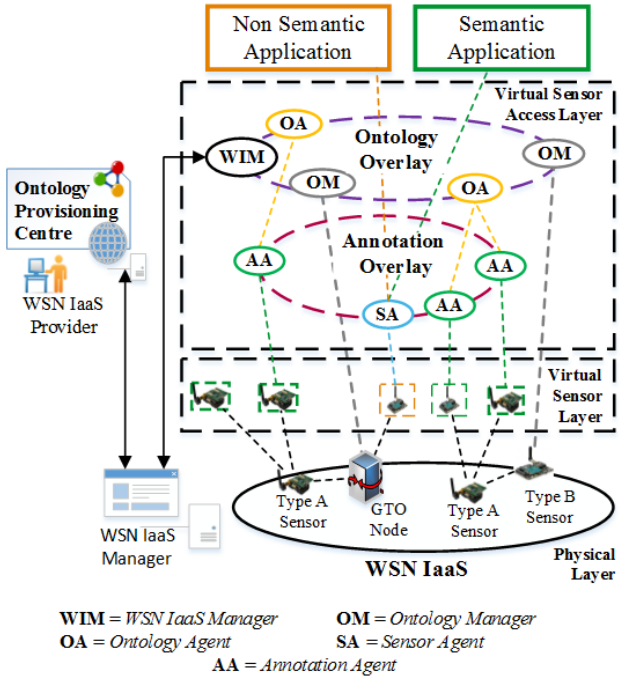

Fig. 4. Ontology Enabled vWSN Architecture

TABLE I
Messages Exchanged for Ontology Deployment

#	Message Name	Message Description	Message Type
1	*Deploy Ontology*	Sent by *OPC* to *WIM* to deploy full Base Ontology	Unicast
2	*Ontology Deploy Request*	Sent from *WIM* to the *OPC* to get the recent version of the Base Ontology	Unicast
3	*Discover Request*	Sent by the *WIM* to nodes in the Ontology Overlay.	Broadcast
4	*Discover Response*	Sent by the selected nodes to *WIM* with their ID	Unicast
5	*Notification*	Sent by WIM to notify selected *OAs & OMs*	Multicast
6	*ACK*	Sent by the selected nodes to *WIM*	Unicast
7	*Ontology Request*	Sent from *OM* to *WIM* to get the ontology	Unicast
8	*Send Full Base Ontology*	Sent by *WIM* to *OM* to send complete Base Ontology	Unicast
9	*Send Partial Base Ontology*	Sent by *WIM* to *OA* to send partial Base Ontology	Unicast

We have introduced several procedures that relay on our proposed OPP: *1)* Discover potential candidates as capable nodes; *2)* Select a subset of capable nodes, *3)* Notify the capable nodes, and *4)* Deploy ontology to the selected nodes.

1) Discover Potential Capable Nodes: Due to inherit limitation of WSN, OPP selects only capable nodes that fulfil the storage and battery power requirements. The mechanism

TABLE II
Discovery Request MESSAGE CONTENT

broadcast_id
source_address
node_storage_capacity
node_battery_life

for candidate discovery uses message 3 and message 4 (in Table I). Initially, WIM broadcasts the discovery request message to its neighbors. The field broadcast_id defines the broadcast message ID and is set by WIM. The source_address holds the ID of discovery request message sender. The final fields (node_storage_capacity & node_battery_life) are used to discover the potential candidates.

Each overlay node maintains a small cache consisting of <source_address, broadcast_id> pair to identify and reject the duplicate messages upon receipt. When a new discovery request message is received, it updates the cache and disseminates it to its neighbors. As a result, each node may receive multiple discovery request messages but it will broadcast the request only once to avoid broadcast storming problem. Each node also compares its storage capacity and energy level against the specified values in the request message. If it satisfies them, a discover response message is sent with its ID to the originator of the request message. The immediate source nodes add their own ID to this message if they fulfil the requirements otherwise they simply forwarded it.

This process continues until the response message reaches back to WIM. WIM can receive multiple response messages from the same neighbor nodes. However, it will take the message with the highest number of node IDs into account. As a result, WIM will get the aggregated response messages with the node IDs of potential candidates. The message aggregation will also help in dealing with ACK explosion problem since it reduces the number of messages in the network.

2) Select the Subset of Capable Nodes: At the end of potential capable nodes discovery phase, WIM will have a pool of ideal capable nodes that meet the storage and battery power requirements. However, it may be useful to further refine and select only a subset of capable nodes. For this WIM can use any appropriate criteria. One example is that it executes a genetic algorithm [6] that takes the node IDs of potential candidates as input and produces a subset of the capable node IDs that can act as OA and OM. WIM can also make selection based on distances or optimal placement of OAs and OMs to ensure complete coverage of the network.

3) Notify the Capable Nodes: After selecting capable nodes, WIM sends a Notification message to them. The message indicates that the targeted node is selected to act as OA and/or OM. The receiving nodes send back ACK message to announce that they are ready to act as OA and/or OM.

4) Provision Base Ontology to the Selected Nodes: In the final step, WIM sends the complete base ontology to the selected OMs. Since OAs are resource-constrained devices and may not need all concepts, WIM divides the base ontology into multiple parts in a way that each part contains all the concepts related to one sensing capability. Then, each parts is randomly sent to the selected OAs. When a different concept is required, an OAs can request it from an OM in the ontology overlay.

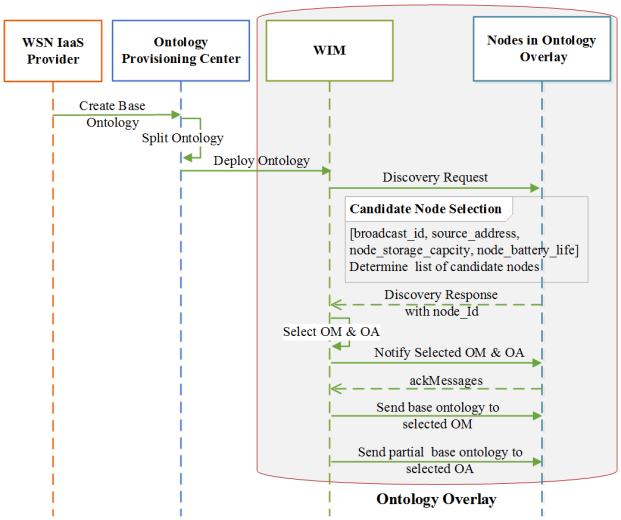

Fig. 5. Ontology Provisioning Steps

B. Illustrative Scenario

The sequence diagram in Figure 5 illustrates the steps involved in base ontology provisioning over vWSN.

After the base ontology is created, OPC splits it as mentioned before. The base ontology is then sent to WIM. A discovery request message is sent from WIM to the neighbouring nodes in the ontology overlay. After receiving response from potential capable nodes and executing genetic algorithm, WIM notifies the selected OM and OA nodes and receives ACK message. Finally, WIM sends the complete and partial base ontology to OMs and OAs respectively.

When the ontology is deployed it needs to be used by AAs for sensor data annotation in real-time. This can be done either using proactive or reactive approach. In the proactive approach, OMs send parts of the base ontology to the chosen OAs that, in turn, sends the received parts to the AAs. In the reactive approach, the virtual sensors send data to AAs for annotation. If AA does not have the required concept, it will make a request to the OAs. If available, the OA will send the partial ontology to the requesting AA; otherwise, the request will be sent to OM to provide the required ontology.

VI. PROTOTYPE IMPLEMENTATION & SIMULATION DETAILS AND RESULTS

In this section, we first present the implemented prototype and the results. Then, we compare our proposed protocol with simple flooding protocol for finding the capable nodes (OA, OM) and present the simulation results.

A. Prototype Implementation and Results

The implementation architecture is shown in Figure 6.

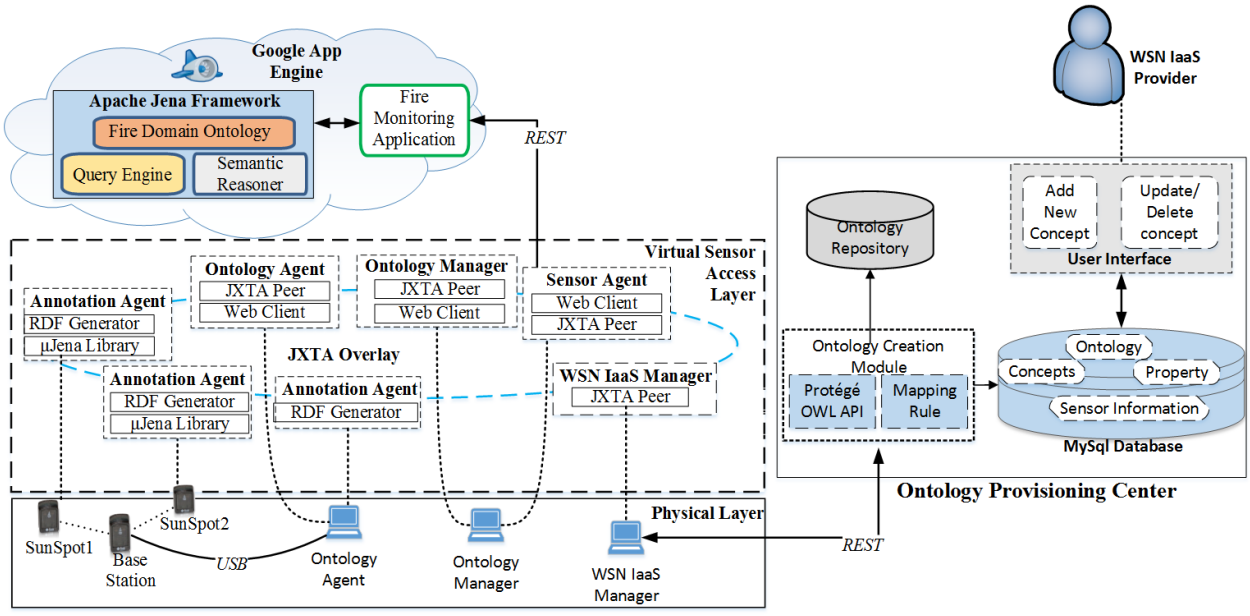

Fig. 6. Implementation Architecture

1) Prototype Implementation: The ontology provisioning center application is developed in Java, JAX-WS web services API and MySQL Database to store the base ontology concepts. Since it is a web-based application, it is hosted by the Apache Tomcat web server. In order to generate the base ontology using the stored concepts in the database, we use the Protg 3.8 API, an open-source Java-based library for OWL and RDFS. We use the Java-based JXSE implementation of JXTA protocol for the creation of annotation and ontology overlay. The WSN IaaS Manager, OM and OA implement JXTA Rendezvous Peer functionality to store the base ontology and its parts respectively. We use the JXTA Content Management System (CMS) to send the base ontology from WIM to OM, then from OM to OA and finally from OA to AA.

To evaluate the performance of the prototype, we use the following metrics: Overlay Creation Delay (OCD), Ontology Distribution Time (ODisT) and Ontology Download Time (ODT). OCD is the time to generate the JXTA overlay from a non-existent state to a ready state, when it is ready to accept the join requests. ODisT is the summation of the following delays: *i)* Delay from the ontology provisioning center to WIM; *ii)* delay from WIM to OM; and *iii)* delay from OM to OA. For ODT, we measured the delay when AA requests and receives the missing part of base ontology from OA.

2) Prototype Results: Figure 7 shows the OCD of 50 experiments and its average value of 1906ms. It is important to note that the OCD depends on the configurations of the machines acting as JXTA peers and is unavoidable. However, OCD occurs only during the overlay initiation phase therefore it does not necessarily make impact on the annotation process.

Figure 8 shows the ODisT. Both the ontology provisioning center and WIM are implemented on the same laptop so the delay between these two entities was minor, therefore it is not considered in the results. The average delay from WIM to OM

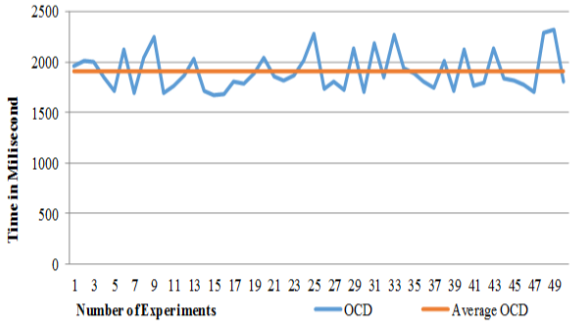

Fig. 7. Overlay Creation Delay

is 56ms. The average delay from OM to OA is 54ms. In total, the average ODisT from 50 experiments is 110ms.

Figure 9 shows ODT when a particular AA requests for the required part of the base ontology and receives the corresponding OWL file. The average ODT from 50 experiments is 105ms, typically found in private LAN settings using JXTA. The reason for higher ODT as compared to ODisT is that ODT includes both the request & reply delays while AA first sends a request for the ontology file and later receives it.

Overall OCD, ODisT and ODT delays are in acceptable range. Typically OCD and ODisT will have no impact on application performance since they will only occur during the setup phase. ODT will have impact if the required ontology concept is not available with AA but only on first few sensor messages that require the new concept for annotation.

B. Protocol for Finding Candidate Nodes to Act as OM & OA

1) Simulation Setup: We have developed a discrete event simulator [7] to implement the protocols to find OA and OM nodes that create the Ontology Overlay.

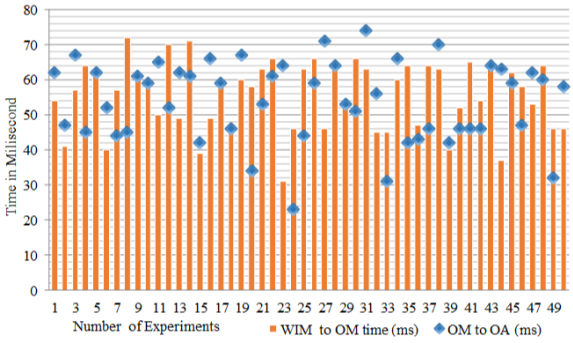

Fig. 8. Ontology Distribute Time

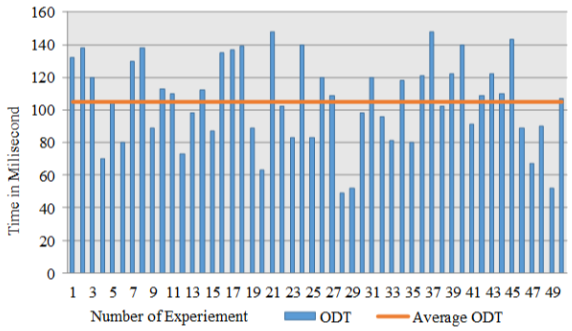

Fig. 9. Ontology Download Time

We consider two protocols to discover the OA and OM nodes: *(a)* No Delay Simple Flooding (NDSF); *(b)* Delayed Aggregation Flooding (DAF) in two network topologies i)*Grid* and ii)*Random* [8]. In both protocols, upon receiving a message for the first time, a node updates its cache and sends it to its neighbors. Duplicate messages are discarded. In the NDSF approach, the capable node sends a reply message immediately upon receiving a Discovery Request Message (DReqM). In DAF each node waits for an interval before sending a response message in order to aggregate responses from neighbor nodes. In the Grid topology, we assume the nodes are at a fixed distance in a rectangular area to represent a planned deployment. In Random topology, the positions of the nodes are determined with the Gaussian distribution [9] to represent an unplanned deployment of the sensors.

We vary the network size (i.e. number of nodes) from 1000 to 5000 and a parameter R (node degree) as the number of directly connected neighbors and the parameter C as the percentage of capable nodes. We evaluate the protocols by considering two different values of R i.e. (18, 25), and two different values of C i.e. (40%, 60%). The Mean and standard deviation for the Gaussian distribution was

$\frac{\sqrt{network\_size}}{2}$ and $\frac{\sqrt{network\_size}}{4}$ respectively.

We evaluate the performance of the protocols using the following metrics: a) Convergence Time (CT): the number of discrete events required to discover all the potential candidates and b) number of Discover Response Messages (DRM).

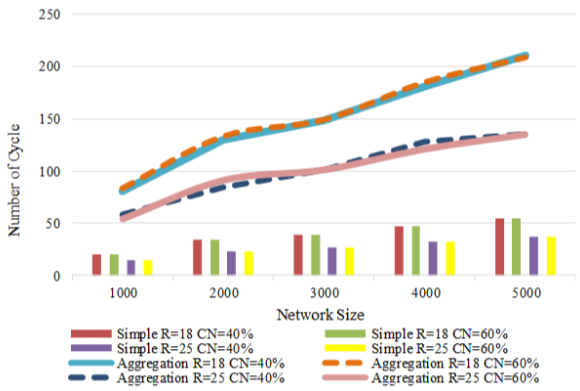

Fig. 10. Total Convergence Time of Grid Topology

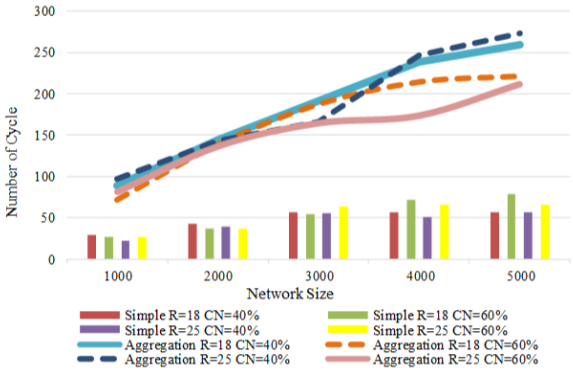

Fig. 11. Total Convergence Time of Gaussian Topology

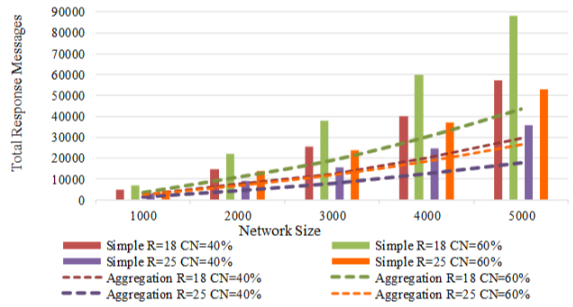

Fig. 12. Total Discovery Response Messages in Grid Topology

2) Simulation Results: Figure 10 and Figure 11 shows the CT of the Grid and Gaussian topologies using NDSF and DAF approaches respectively. For both topologies, the network convergence time is higher in the DAF approach, because each node waits for a certain time period before sending a response message. Since this discovering process will run during the network setup phase, we can ignore the fact that DAF approach needs more time to converge compared to NDFS. The DAF approach also helps to reduce the number of DRMs in the network thereby resolving the ACK explosion problem.

Figure 12 and Figure 13 show the total DRMs in the Grid and Gaussian topology respectively. The DAF approach performs better due to less number of DRMs in both topologies. The reason is that each node gets the opportunity to aggregate DRMs from large number of neighbors.

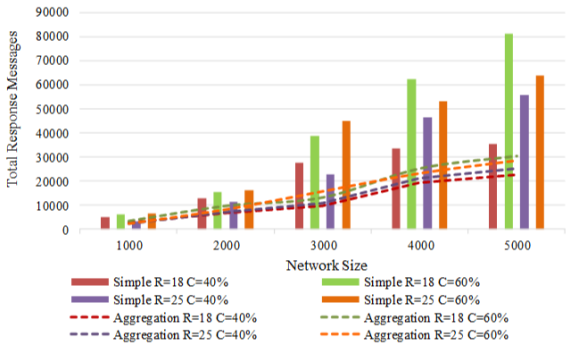

Fig. 13. Total Discovery Response Messages Gaussian Topology

VII. Related Work

Several knowledge management systems have been proposed for the development and management of ontologies. However none of the existing works fulfils our identified requirements. Authors in [10] propose ONKI framework for collaborative development and management of ontologies but ONKI uses centralized approach and is not useful for vWSNs. An ontology management framework proposed in [11] allows developers to create and manage ontologies but there is no mechanism to provision the developed ontology.

The works such as [3], [12] and [13] do not meet our requirement of technology independence. Furthermore, they do not include the procedures of ontology provisioning. More recent efforts include [14], [15] and [16] but they do address the ontology management problem. A few protocols exist for managing the network infrastructures such as Simple Network Management Protocol (SNMP) [17] and Ad Hoc Network Management Protocol (ANMP) [18]. However, they do not tackle ontology development, management and distribution.

VIII. Conclusion

In this paper, we proposed a framework to allow the provisioning of base ontology over vWSN. Our proposed solution allows the IaaS provider to easily develop, manage and use ontologies related to the deployed infrastructure. A proof-of-concept prototype is developed to show the feasibility of the proposed architecture as well a simulation study to discover OA and OM nodes that form the Ontology Overlay.

Several future work items also been identified. *First*, is to investigate on the integration of our framework with Platform-as-a-Service (PaaS) for rapid provisioning of applications that can be offered as SaaS. Issues like IaaS ontology discovery and publication at PaaS level need to be addressed. *Second* item is the optimization of the proposed algorithm in order to select capable OM and OA nodes to store the base ontology and deal with issues like OM and OA failures.

Third item is to work on a hierarchical architecture with several WIMs since this approach appears more relevant in the context of Smart City and IoT paradigms. However, it remains to be investigated if a signaling mechanism between these WIMs is required for collaboration. *Fourth and Final* item is to use lightweight formats like JSON-LD to generate annotated data instead of RDF since JSON is developer friendly and introduces little overhead while achieving the same results.

Acknowledgment

This work was supported in part by the Natural Science and Engineering Council of Canada (NSERC) Canada Research Chair in End-User Service Engineering for Communications Networks and by an NSERC Discovery Grant.

References

[1] I. Khan, F. Belqasmi, R. Glitho, N. Crespi, M. Morrow and P. Polakos, "Wireless Sensor Network Virtualization: A Survey," in *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 553-576, 2015.

[2] P. Desai, A. Sheth and P. Anantharam, "Semantic Gateway as a Service Architecture for IoT Interoperability," *Mobile Services (MS), 2015 IEEE International Conference on*, New York, NY, 2015, pp. 313-319.

[3] A. Sheth, C. Henson and S. S. Sahoo, "Semantic Sensor Web," in *IEEE Internet Computing*, vol. 12, no. 4, pp. 78-83, July-Aug. 2008.

[4] I. Khan, R. Jafrin, F. Z. Errounda, R. Glitho, N. Crespi, M. Morrow, P. Polakos, "A Data Annotation Architecture for Semantic Applications in Virtualized Wireless Sensor Networks", *in proceedings of Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, May 11-15, Ottawa, ON, 2015, pp. 27-35.

[5] M. Compton, et al., "The SSN ontology of the W3C semantic sensor network incubator group", *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 17, pp. 25-32, ISSN 1570-8268, Dec 2012.

[6] I. Khan, et al., A genetic algorithm-based solution for efficient in-network sensor data annotation in virtualized wireless sensor networks, in *proceedings of 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, Las Vegas, USA, 2016.

[7] G. Fishman, "Discrete-event simulation: modeling, programming, and analysis". *Springer Science & Business Media*, 2013.

[8] M. Younis, I. Senturk, K. Akkaya, S. Lee, F. Senel, "Topology management techniques for tolerating node failures in wireless sensor networks: A survey", *Computer Networks*, Volume 58, pp. 254-283, Jan 2014.

[9] Y. Wang, W. Fu and D. P. Agrawal, "Gaussian versus Uniform Distribution for Intrusion Detection in Wireless Sensor Networks," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 2, pp. 342-355, Feb. 2013.

[10] V. Komulainen, A. Valo, and E. Hyvnen, "A collaborative ontology development and service framework ONKI", *2nd European Semantic Web Conference (ESWC 2005)*, Helsinki University of Technology, Laboratory for Media Technology, 2005.

[11] N. F. Noy and M. A. Musen, "Ontology versioning in an ontology management framework," in *IEEE Intelligent Systems*, vol. 19, no. 4, pp. 6-13, Jul-Aug 2004.

[12] S. Zhou, H. Ling, M. Han, and H. Zhang, "Ontology Generator from Relational Database Based on Jena", *Computer and Information Science*, vol. 3, no. 2, Apr. 2010.

[13] A. Alishevskikh and G. Subbiah, "Simple Ontology Framework API", http://sofa.projects.semwebcentral.org [Accessed - 26-02-2016]

[14] A. Gyrard, C. Bonnet, and K. Boudaoud, "A machine-to-machine architecture to merge semantic sensor measurements," in *WWW 2013, 22nd International World Wide Web Conference, Doctoral Consortium*, May 13-17, 2013, Rio de Janeiro, Brazil, 2013.

[15] A. Gyrard, C. Bonnet, and K. Boudaoud, Enrich machine-to-machine data with semantic web technologies for cross-domain applications," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, March 6-8, 2014, Seoul, South Korea, pp. 559-564. 2014

[16] X. Su, H. Zhang, J. Riekki, A. Keränen, J. K. Nurminen, and L. Du, "Connecting IoT Sensors to Knowledge-based Systems by Transforming SenML to RDF," *Procedia Comput. Sci.*, vol. 32, pp. 215-222, 2014.

[17] A. Affandi, et al., "Design and implementation fast response system monitoring server using Simple Network Management Protocol (SNMP)," *Intelligent Technology and Its Applications (ISITIA), 2015 International Seminar on*, Surabaya, 2015, pp. 385-390.

[18] W. Chen, N. Jain and S. Singh, "ANMP: ad hoc network management protocol," in *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1506-1531, Aug 1999.