Integrating Wireless Sensor Networks in the NATO Network Enabled Capability using Web Services

Joakim Flathagen*‡¶ and Frank T. Johnsen*
*Norwegian Defence Research Establishment (FFI), ‡Q2S NTNU, ¶UNIK
Email: joakim.flathagen@ffi.no, frank-trethan.johnsen@ffi.no

***Abstract*—Wireless Sensor Networks (WSNs) are expected to provide greatly enhanced situational awareness for warfighters in the battlefield. Sensors widespread in the battlefield are however, of very limited value unless the sensors are reliable during the entire operation and the information produced is accessed in a timely manner. In this paper we focus on these issues by enabling WSNs as a capability in the NATO Network Enabled Capability (NNEC) using Web services. We demonstrate that Web services is an enabling technology for information-sharing, facilitating presentation of sensed data and alarms to a battlefield management system. In addition, we show the feasibility of using a Web services approach as a query processing tool enabling multi-sensor fusion and data aggregation in the WSN domain. The networking protocols can in this way inherently adjust data-aggregation and -processing criteria according to the requirements posed by external subscriber systems. In this way, energy efficiency, which is paramount in WSNs, is optimized without sacrificing the flexibility of Web services. Our proposed methods are tested using practical experiments with TelosB sensing nodes.**

***Index Terms*—Wireless Sensor Networks, Web services, Collection Tree Protocol**

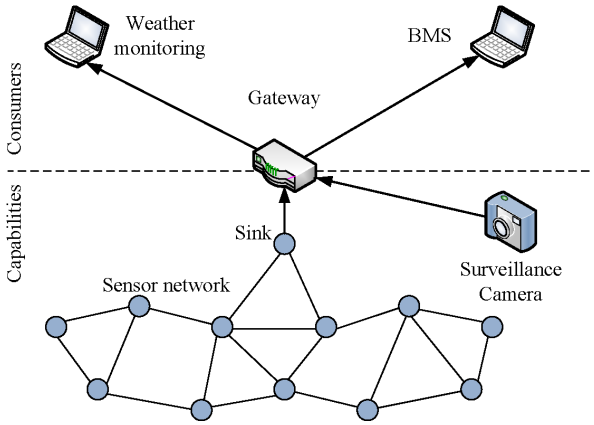

Fig. 1. Sensor network enabled as a service providing capabilities to different consumers. The gateway may invoke additional services to provide a composite service.

I. Introduction

Recent advances in integrated circuit design, micro electromechanical sensors and wireless network technology have enabled the development of low cost wireless sensors that can be deployed in large quantities. Wireless Sensor Networks (WSNs) can sense and gather information about the environment automatically and unattended. In the tactical domain, great benefit can be achieved by using covert miniaturized sensors, as they are difficult to avoid by a possible intruder and less subject to vandalism or theft compared to traditional sensor systems. Further, the network protocol redundancy and the vast number of sensing nodes improve reliability and minimize the false alarm probability compared to previous sensor systems.

Sensors widespread in the battlefield are, however, of very limited value unless the information is accessed and shared in a timely manner [1]. One of the main goals of the NATO Network Enabled Capability (NNEC) is to address this issue by facilitating seamless linking of sensors, decision makers and weapon systems. The NNEC feasibility study has identified *Web services* as the key enabling technology for NNEC [2]. Web services technology is based on a number of standards, which help ensure that different implementations from different vendors are interoperable. In this paper we explore enabling wireless sensor networks as a capability in NNEC using Web services. Since WSNs have scarce resources in terms of available bandwidth, battery, and computational power, it does not make sense to attempt to service-enable each and every sensing node. Instead, we use a wrapping approach, thus allowing existing mechanisms to be used within the WSN, while nodes external to the WSN may configure and receive information from the network using Web services. External consumer systems are for example Battlefield Management Systems (BMS) or Weather Monitoring Stations, see Fig. 1.

We do not, however, consider Web services only as an information-sharing and interoperability entity. In our architecture, we also suggest the use of a Web services gateway as a query processing system publishing relevant sensing and alarm-criteria to the WSN domain. The networking protocols can in this way inherently adjust data aggregation and processing criteria according to the requirements posed by the external subscriber systems. In this way, energy efficiency, which is paramount in WSNs, is optimized without sacrificing the flexibility of Web services.

The paper presents our Web services based WSN architecture and a real case-study to demonstrate our ideas applied to a tactical scenario. Before presenting our own setup and results in detail, it is worth reviewing some of the previous and related research.

II. Related work and background

Directed diffusion [3] was one of the first initiatives to create a combined routing and query system for WSNs. In DD, the queries are formatted as *interest* messages which are disseminated to all sensing nodes. Gradients from each sensing node back to the base station are set up during the interest dissemination. Since the interest messages are not reliably transmitted throughout the network, the base station must periodically retransmit the interest message. Directed diffusion supports in-network data processing and aggregation, and the interest message formation allows publish-and-subscribe to occur at a very fine-grained level. However, the protocol is based on a query-driven on demand data model, and is not efficient for event-initiated alarm scenarios, such as e.g., tactical surveillance. The interest message formation in Directed Diffusion is using a proprietary format and is therefore not appropriate when used in a multi-consumer WSN such as the one in Fig. 1. Query processing systems such as TinyDB [4] aim to provide a flexible and simple query API by enabling queries written in a SQL-like language inspired from Data base systems. Hence, queries can be formulated remotely by multiple consumers using different physical entities. As opposed to DB systems, the queries here operate on real-time streams of data passing through memory rather than performing queries to a disk. TinyDB queries are input to the base station node, which sends an optimized version of the query to the sensor network. In the network, the sensing nodes that have data satisfying the query predicates, formulate an answer. These answers are returned to the base station (or sink). Data can be transformed, combined, and summarized according to the query.

If WSN-interaction is necessary in a multi-consumer setting, *Web services* provide higher flexibility and increased interoperability compared to extending querying protocols to each consumer. Notice that there are many definitions of "Web services". The core idea is the same (i.e., using XML-formatted data for information exchange), but some of the finer details may vary. For example, the REST approach ignores most of the Web services standards and specifications, meaning that REST is too restrictive if one wants to implement a pervasive SOA for military networks. We need the flexibility of a broader spectrum of the Web services specifications for NNEC. Thus, when we discuss Web services in this paper we use the definition by the W3C [5]: "A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards."

A Web service based WSN can be realized either by service-enabling each and every sensing node or by providing a Web service gateway that hides the inner WSN protocols. The work by Delicato et al. [6] was an early architecture

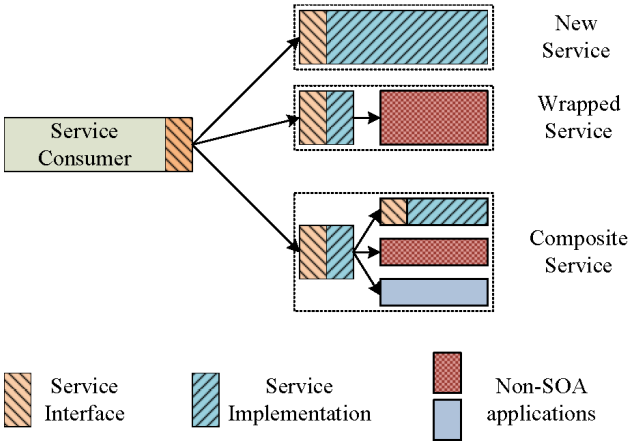

Fig. 2. Creating services (adapted from [12])

work belonging to the first category proposing to integrate full SOAP support in the WSN sensing nodes. The full SOAP will however, often lead to tremendous overhead due to the verbose XML format. Although compression can reduce the overhead of XML significantly, and binary coding such as Efficient XML can enable XML to be used at the tactical edge [7], our previous research [8] has shown that the overhead associated with compression libraries make them unsuitable for use on severely limited devices. Thus, in contrast to other WSN implementations, such as [9], we do not attempt to employ XML compression in our WSN in this paper.

An alternative method to reduce the overhead is to convert the XML messages to a more optimized format at a gateway before relaying them to the WSN devices. The authors of [10] for example, propose WSN-SOA to reduce XML formats to a size applicable for 802.15.4 devices, while Bressan et al. [11] rely on the Constrained RESTful Environments (CoRE) based on REST. We argue that there is no point in extending Web services to every sensing node. In contrast, the WSN should (from the Web services perspective) be seen as one single sensing unit, providing filtered and aggregated sensed data to one or more consumers. Therefore, a gateway should be responsible for interacting with the WSN nodes on the back-end side, and the consumers on the front-end side. This approach lets the WSN designers focus on energy-efficient protocols inside the WSN, thus limiting the need for implementing computationally intensive standards to the gateway which provides an interface to the outside world.

III. SOA for Wireless Sensor Networks

Web services technology is based on a number of standards, which help ensure that different implementations from different vendors are interoperable. In this paper we explore enabling WSNs as a capability in NNEC using Web services. There are several ways of realizing a capability as a service. For example, a service may be created from scratch, it may function as a front-end to a legacy system, or it may be a combination of existing services, as illustrated in Figure 2.

Since WSNs have scarce resources in terms of available bandwidth, battery and computational power, it does not make sense to attempt to service-enable each and every sensing node. Instead, we use the wrapping approach, thus allowing existing mechanisms to be used within the WSN, while nodes external to the WSN may configure and receive information from the network using Web services. Even if the SOAP messages themselves do not have to be transmitted to every sensing node, it is crucial that available query information inside the XML payload is utilized to optimize the overall system performance.

The first contribution of our proposed architecture is therefore to provide a Web services wrapper that enables external consumers to interoperate with the sensor network using XML and Web services. The interaction operates in both directions. The second contribution of the architecture is query dissemination and collection formation that is adaptive and based on the requests posed by the Web service consumers. The architecture is shown in Fig. 3 and is described subsequently.

A. Our gateway: A Web service wrapper

The gateway contains the Web services wrapper and provides an interface (a front-end) to the WSN using established Web services standards. A WSDL file defines the interface, data types and message flow, whereas SOAP is employed for message transmission. This part of the wrapper is accessible to other systems using COTS Web services technology. The Web service interface allows external clients to configure queries for the WSN, and register a service endpoint (EP) for pushed information. In other words, our wrapper supports the publish/subscribe pattern, in that clients register a query (step 1, subscription providing recipient EP) and results of this query (be it periodic reports or spontaneous alarms) are sent (i.e., published directly to the consumers in steps 6 and 7) to the registered service endpoint. A client connecting to the gateway is typically a BMS, requesting alarm reports when a subset of the sensing nodes detects an intruder which is trespassing the area monitored. Such an example query is shown in Listing 1.

Listing 1. XML Query requesting alarm reports when at least four IR detectors are trigged

```
<GetIntruder>
    <MinPIRDetections>4</MinPIRDetections>
    <LightMaxThreshold>1lux</LightMaxThreshold>
    <Duration>30d</Duration>
    <IncidentReport>http://10.0.0.2/</IncidentReport>
</GetIntruder>
```

When the WSN reports to the gateway (step 3) about a detected target, the gateway sends a request to a separate Web service enabled camera (step 4) to take a picture covering the area monitored. The target information (from step 3) and the picture provided (by step 5) are combined to a report sent to the BMS endpoint (e.g., step 6 and/or 7). COTS Web services technology is used to implement step 1 as well as steps 4 through 7, limiting proprietary solutions only to the functionality implemented in the back-end system, i.e., steps 2 and 3. Thus, our prototype follows the guidelines of the NNEC FS, using Web services technology to loosely couple services and clients.

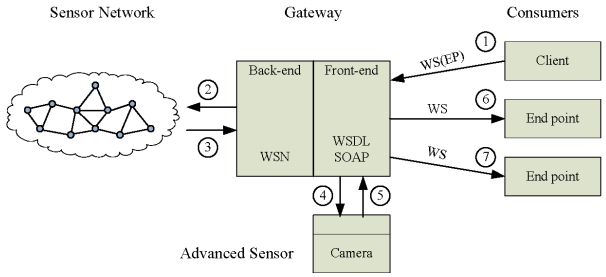

Fig. 3. Architecture

Another Web services client could be a weather monitoring station, requesting periodic temperature or humidity reports. A typical temperature report query, requesting individual temperature readings from each sensing node each 30 minute, is formatted in XML as shown in Listing 2.

Listing 2. XML Query requesting temperature reports from all sensing nodes

```
<GetTemperature>
    <CollectionStyle>Individual</CollectionStyle>
    <Duration>30d</Duration>
    <Interval>30m</Interval>
    <IncidentReport>http://10.0.0.1/</IncidentReport>
</GetTemperature>
```

In addition to supporting third party consumer applications, the architecture can also provide special case Web services for example to provide network developers with real-time information about the network at any given time, either during the initial deployment, create mid-life status reports, or to assist redeployment of energy exhausted nodes. These reports can be forwarded to a dedicated monitoring endpoint.

At the back-end, the gateway communicates with the WSN using two different traffic patterns: Dissemination (step 2) and Collection (step 3).

B. Dissemination of queries

The Web services wrapper shown in Fig. 3 interfaces with a back end, where the incoming XML configuration requests are transformed to a much more resource efficient, proprietary format used in our WSN. The format uses a compact and simple representation of sensor queries. Keys and attributes are represented as small integer values instead of text strings. Typical keys here are sensor type identificators and attributes are threshold values and timer values. A typical XML formatted Web services query of 200-300 bytes is translated to a small 10-15 byte message.

To disseminate the compact query through the WSN a dissemination protocol is required. Since messages can be lost due to e.g., collisions, channel noise or even buffer overflow, the dissemination protocol needs to be reliable. In addition, message synchronization could be necessary after a node reboot, e.g., if an application failure causes the watchdog timer to elapse. This means that simple flooding of the queries is not sufficient. In our implementation we have used Drip

dissemination [13] to account for the above circumstances. The constrained power budgets in WSNs often lead to slow converging dissemination protocols. Drip copes with this issue by building a reliable transport layer on top of the Trickle algorithm.

C. Collecting sensed data

In most sensor networks, the majority of the network traffic is destined to the sink. For such networks, a collection tree traffic pattern is preferred in rather than other ad hoc network protocols. Instead of implementing our own collection tree, we use the Collection Tree Protocol (CTP) [14] which is the de-facto collection protocol in TinyOS and is used successfully in many real WSN deployments. CTP consists of two parts; (i) data path validation, to quickly discover and fix routing inconsistencies by taking advantage of the data traffic; and (ii), adaptive beaconing using the Trickle algorithm, which optimizes the standard trade-off between low routing beaconing traffic overhead and low route repair latency. The anycast pattern employed by CTP also enables the possibility to extend our setup to a multiple sink architecture for increased reliability and reduced overall power consumption.

D. Aggregation

In our architecture, we focus on balancing the trade-off between the limited resources of the WSN and the required system performance necessary to fulfill the Web services query predicates. A query may for example ask for detailed reports requiring that every sensed value should be collected from the WSN and presented in a combined form in a Web services report. Alternatively, the query could indicate that a small report of filtered or aggregated measurements is preferred. Data aggregation in the Web services architecture can be employed either at the Web services gateway, or inside the WSN. From an energy-efficiency point of view, the latter alternative is preferred. To accomplish this, we have implemented a flexible data aggregation scheme running on the WSN nodes. Although the standard CTP does not include aggregation, the forwarding engine in CTP allows a routing extension to intercept the packets relayed by an intermediate node. Different aggregate functions can therefore alter the data upon interception as the sensed data traverses the collection tree.

Most data queries requests for periodically transmitted reports (e.g., each minute, each hour or each day). However, as the period timers are not fully synchronized among the nodes, there is an unknown time gap t between the first and the last node producing data in each period. Each node in the aggregation tree will therefore observe a gap $g \leq t$ between the arrival times of the sensing messages it receives from its child nodes. This time gap represents a challenge in WSN designs. If data freshness is paramount, each node should send its own measurements immediately when its period timer elapses, and retransmit all upstream messages immediately upon reception (i.e., no data aggregation). On the other hand, if the optimization objective is energy efficiency, each node should wait for a time $\geq t$ to account for all messages delivered from its child nodes before aggregation and transmission. The optimum balance between data freshness and energy efficiency can be found by optimizing the aggregation timeout of each node. One solution is to take advantage of the node position in the routing tree, as shown by Solis and Obraczka [15]. Our data aggregation algorithm on the contrary, minimizes the aggregation delay on each node without any routing protocol information. Rather, the node can learn g (the expected time difference between the messages received from its child nodes) by observing the inter arrival time of the packets received. The child node that triggers the end time of the period g is used as a synchronizer node trigging sensing, aggregation and transmission of the final data packet. Each node chooses the child node that constitutes the start of the maximum inter-arrival time in one periodic cycle as its synchronizer node (see Algorithm 1).

Algorithm 1 Data aggregation

Intercept $(Message\ m, Node\ n, f(m) = NO|MIN|MAX|AVG)$
- $if\,(f(m) == NO)\,send\,(m)\,, exit$
- $\mathcal{M} = aggregate\,(\mathcal{M}, m, f(m))$
- $\mathcal{T}\,(l) = t_{last} - t_{now}$
- $s = \arg\max_i \mathcal{T}\,(i)$
- $if\,(n == s)$
  - $\mathcal{M} = aggregate\,(\mathcal{M}, M_{this}, f(m))$
  - $send\,(\mathcal{M})$
- $l = n$
- $t_{last} = t_{now}$

On_Synchronizer_timeout (s)
- $\mathcal{M} = aggregate\,(\mathcal{M}, M_{this}, f(m))$
- $send\,(\mathcal{M})$
- $\mathcal{T} = 0$
- $s = 0$

If the synchronizer node times out (e.g., the CTP routing tree has changed), the node immediately transmit the aggregate of its temporarily stored data and sensed data and chooses a new synchronizer node on the next period. If no synchronizer is found, the node is a leaf node, and transmits sensed data immediately after its period timer has elapsed. Our aggregation scheme supports the following aggregation functions: *Average*, *Minimum*, *Maximum* and *No aggregation*, and adapts according to the queries transmitted from the gateway back end. *No aggregation* means that all measurements are delivered to the Web services gateway. Here, the measurements are combined to a joined report before reporting to the EP. The join-process could also include aggregation, but in-network aggregation is preferred.

IV. Evaluation

A. Experimental setup

The experiment was set up as shown in Fig. 3. The WSN consisted of 20 wireless tmote sensing nodes [16] running TinyOS 2.1.1. The nodes were equipped with the following sensors: sound, light, temperature, humidity, ultrasound, and passive IR (PIR) (see Fig. 4). The gateway with the Web

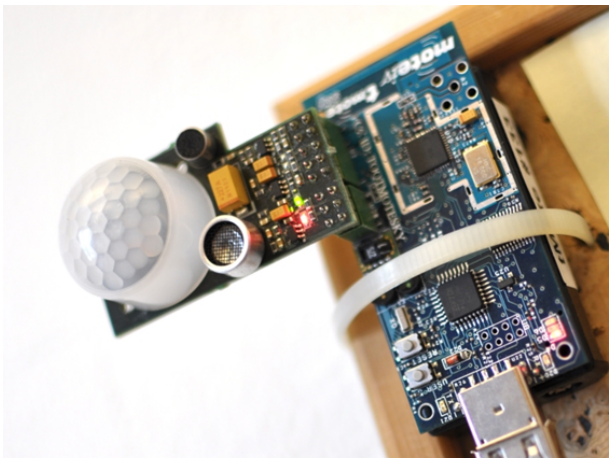

Fig. 4. The test network contains 20 tmote sensing nodes with IR-detectors

services wrapper ran on a iEi industrial computer with Linux, while the camera Web service was installed on a separate standard computer with a camera attached. The Web services consumers consisted of our trial client software on two standard computers.

We focused on two scenarios. First, a target detection scenario. In this scenario, the PIR sensors were used to detect possible targets trespassing the monitored area. The separate imaging sensor was used to take a picture of the target to provide target verification. The minimum number of PIR detectors detecting the target before classifying the event as an alarm, was configurable by Web services query created by the consumer system. Such an example query is shown in Listing 1. In the second scenario, an external system requested weather reports that should be presented periodically. An example of this query is shown in Listing 2. Besides performing functional testing of the architecture, we tested the effectiveness of the data format and data aggregation to obtain deeper insight of the system.

B. SOAP-based query vs. reduced query

We quantify the effectiveness of our reduced data format (RF) by comparing it with equivalent SOAP-based Web services. To reduce the unnecessary overhead, we removed the standard SOAP headers before dissemination with Drip. The query used for the experiment is shown in Listing 2. Our reduced information format message (12 bytes) was disseminated using the same method. Because of the very limited available memory on the tmote sensing node, we did not implement an XML parser but focused merely on the dissemination procedure in our experiment.

We performed 20 disseminations for each message format for networks with sizes 5,10 and 20 nodes respectively. The average node degrees in the networks were between 3 and 5. The 95% confidence intervals are given in the figures. Fig. 5 shows the time elapsed until all nodes had successfully received the query. Although Drip guarantees data delivery in a connected network, the delivery time can be severe, and increases with the size of the message disseminated. Overall, the RF format reduces the dissemination time to about a fifth of the time observed when disseminating XML.

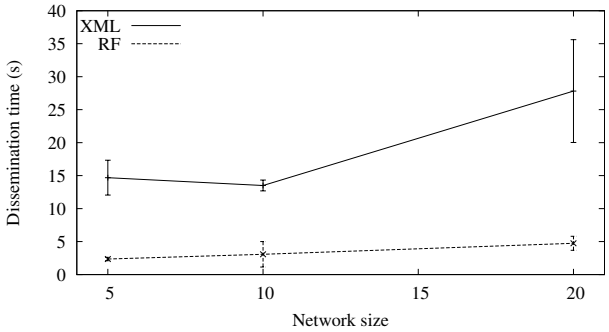

Fig. 5. The time required to fully synchronize the network

Fig. 6 shows the total energy spent on the dissemination process. The energy spending is calculated by observing the CC2420 radio load on each node, and accounting for the current draw of the tmote in RX/TX/Idle states from [16]. The XML encoded message results in more than eight times the power consumption compared to using the reduced format messages. These results illustrate that XML queries can indeed be transmitted to every node. However, in order to ensure reliable dissemination, the huge message size, which is difficult to avoid with XML, increases the energy consumption and prolongs the dissemination delay compared to using a more optimized format. It is also worth noting that XML gives no particular advantage compared to the reduced format in our homogeneous sensor network. A highly heterogeneous network may, on the other hand, benefit from of XML.

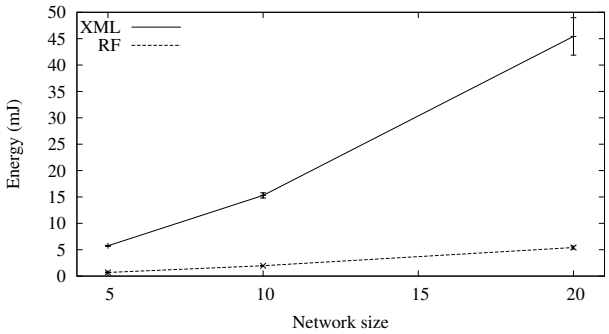

Fig. 6. The total energy spent in the dissemination process

C. In-network aggregation vs. gateway-aggregation

The Web services query predicates determine the proper aggregate function of the network system. In-network data aggregation is more complex to implement than relying on data aggregation only at the gateway. With this in mind, it is interesting to examine the performance of these two radically different strategies. With the first strategy, individual sensor readings were requested each 20s from all nodes. In this case,

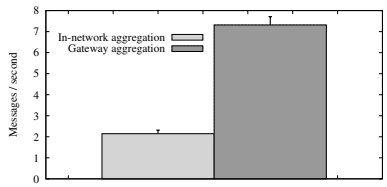

Fig. 7. The effect of data aggregation with 20 sensing nodes.

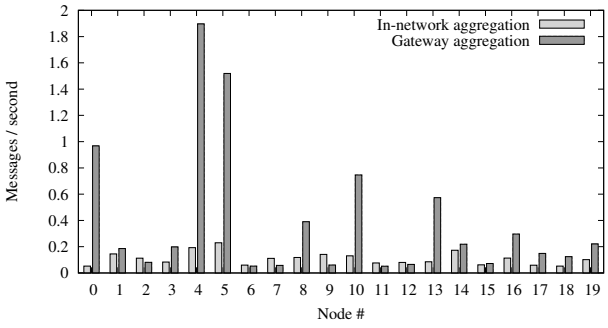

Fig. 8. The message flow distribution in the network with and without data aggregation

the aggregation took place at the gateway and there is no aggregation in the WSN. With the next strategy, the sensor nodes employed the aggregation strategy presented in section III-D. We used a 20-node tmote test-bed and performed 10 one-hour runs for both strategies.

Fig. 7 illustrates the total network load (in messages processed per second) both for the case with in-network aggregation and for gateway aggregation. The 95% confidence intervals are given in the figure. We observe that in-network aggregation significantly reduces the message load in the network. In Fig. 8, we examine the load on each sensing node separately. The figure shows that when in-network aggregation is enabled, message load is also better distributed.

From the literature, we know that the effect of in-network data aggregation increases with the size of the network. However, our results show that even a small network such as our 20-node network, can benefit greatly by employing in-network data aggregation. CTP focuses on establishing stable (low-ETX) routes rather than short routes. Hence, the number of hops involved in an arbitrary message transmission may be high, and the effect of in-network data-aggregation incrreases accordingly.

V. Conclusion

The results from our test-bed implementation shows that our Web services based architecture is feasible in a real setting. We were able to show that the WSN can take advantage of the attribute information in Web services queries provided by NNEC consumers, and that we could optimize the message flow by employing appropriate in-network data aggregation. It should be noted, however, that even if the Web services middleware we used has been identified as a key enabler for NNEC, there is a need for further standardization within NATO. Here, we have shown that it is feasible to use the technology in an NNEC setting, but for actual use in a coalition the interface to the WSN gateway (i.e., the WSDL) must be standardized as well. Finally, we were able to show that the Web services gateway can effectively combine the WSN service with an advanced Web service (camera) to provide a composite service to e.g., a BMS.

References

[1] J. L. Paul, "Smart sensor Web: tactical battlefield visualization using sensor fusion," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 21, no. 1, pp. 13–20, January 2006. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1581121

[2] P. Bartolomasi, T. Buckman, A. Campbell, J. Grainger, J. Mahaffey, R. Marchand, O. Kruidhof, C. Shawcross, and K. Veum, "NATO network enabled capability feasibility study," Version 2.0, October 2005.

[3] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, ser. MobiCom '00. New York, NY, USA: ACM, 2000, pp. 56–67.

[4] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: an acquisitional query processing system for sensor networks," *ACM Trans. Database Syst.*, vol. 30, pp. 122–173, March 2005.

[5] H. Haas and A. Brown, "Web services glossary W3C working group note 11 february 2004," in *W3C*, 2004. [Online]. Available: http://www.w3.org/TR/ws-gloss/

[6] F. C. Delicato, P. F. Pires, L. Pirmez, and L. F. R. d. C. Carmo, "A flexible web service based architecture for wireless sensor networks," in *Proceedings of the 23rd International Conference on Distributed Computing Systems*, ser. ICDCSW '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 730–.

[7] J. Schneider, "Efficient XML: Taking Net-Centric Operations to the Edge," in *13th International Command and Control Research and Technology Symposium (ICCRTS)*, June 2008.

[8] D. Hadzic *et al.*, "Web services i nettverk med begrenset datarate (in Norwegian)," in *FFI report 2006/03886*, 2006.

[9] N. Glombitza, D. Pfisterer, and S. Fischer, "Integrating wireless sensor networks into web service-based business processes," in *Proceedings of the 4th International Workshop on Middleware Tools, Services and Run-Time Support for Sensor Networks*, ser. MidSens '09. New York, NY, USA: ACM, 2009, pp. 25–30.

[10] J. Leguay, M. Lopez-Ramos, K. Jean-Marie, and V. Conan, "An efficient service oriented architecture for heterogeneous and dynamic wireless sensor networks," in *Local Computer Networks, 2008. LCN 2008. 33rd IEEE Conference on*, October 2008, pp. 740–747.

[11] N. Bressan, L. Bazzaco, N. Bui, P. Casari, L. Vangelista, and M. Zorzi, "The Deployment of a Smart Monitoring System Using Wireless Sensor and Actuator Networks," October 2010, pp. 49–54.

[12] Y. Natis, "Gartner research "service-oriented architecture under the magnifying glass"," Application Integration & Web Service, Summit 2005, April 18-20, 2005.

[13] G. Tolle and D. Culler, "Design of an application-cooperative management system for wireless sensor networks," in *Wireless Sensor Networks, 2005. Proceeedings of the Second European Workshop on*, Feb. 2005, pp. 121–132. [Online]. Available: http://dx.doi.org/10.1109/EWSN.2005.1462004

[14] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '09. New York, NY, USA: ACM, 2009, pp. 1–14.

[15] I. Solis and K. Obraczka, "The impact of timing in data aggregation for sensor networks," in *2004 IEEE International Conference on Communications (IEEE Cat. No.04CH37577)*. IEEE, 2004, pp. 3640–3645 Vol.6. [Online]. Available: http://dx.doi.org/10.1109/ICC.2004.1313222

[16] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," in *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*. Piscataway, NJ, USA: IEEE Press, 2005.