Designing a novel SOA architecture for security and surveillance WSNs with COTS

Mario Lopez-Ramos, Jérémie Leguay, Vania Conan

***Abstract*—We consider the challenge of enhancing sensor networks for surveillance and global security with increased distributed data processing capabilities, including multi-sensor fusion, data aggregation or mining, and rule-based alert generation. We advocate a novel architecture that will enable the creation of more resilient and complex monitoring applications. We exemplify its benefit in a chemical accident scenario. The architecture introduces new processing nodes in the field and derives the requirements for the software they will run. We propose to consider the use of a Service Oriented Architecture (SOA) to program and deploy the data processing applications. We analyze existing and on-going work within the Web Services community and conclude that it is possible to implement the architecture with an appropriate combination of COTS (Commercial off-the-shelf software components). We conclude with our plans to move forward in this direction and validate the approach on a hardware and software testbed.**

***Index Terms*— COTS solutions, Service Oriented Architecture, surveillance applications, Wireless Sensor Networks**

I. Introduction

THIS document presents a novel architecture that supports new distributed data processing applications deployed within wireless sensor networks. We start by explaining the need for such new developments to support complex surveillance and area monitoring applications for global security. We provide an example in the case of a chemical accident and we use it to explain the benefits the architecture aims at bringing.

A. Objective

Our main objective in developing a novel distributed architecture for Mobile Ad-hoc and Sensor Systems (MASS) is to bridge a gap between the sensor networks that are deployed in the field and whose purpose is to provide measurements, and the IT backbone infrastructure which is responsible of processing the measurements and interacting with users. Present day systems tend to be hierarchical and to draw a clear boundary between both worlds. The sensors, wired or wireless, isolated or networked, provide a set of measurements. The measurements are not necessarily raw measurements, they can be filtered, correlated, cleansed. But these low level processing functions aim at refining or consolidating the provided measurement information. More elaborate data processing or mining, intelligent fusion of heterogeneous data sources, is carried out on the IT backbone by a central server connected to the sensors via a gateway.

We consider that these assumptions are too restrictive and do not provide the appropriate software architecture to take benefit of MASS in the increasingly challenging scenarios of surveillance and monitoring for homeland and global security.

The architecture introduces a new class of devices that mediate between the low-level sensors and the IT backbone high capacity servers. These intermediate devices can connect to sensors through a wired or radio air interface. Their purpose is to support enhanced data processing functionalities in the field. The nodes can be packaged with several sensors; they may be deployed on vehicles, they may be worn by the crew or they can be temporarily installed in the vicinity of the area of operation. The nodes form a wireless adhoc or mesh network among themselves; they serve as distributed gateways to sensors or Wireless Sensor Networks (WSNs); they form a communication infrastructure that supports data exchange between them, the sensors and the (possible) remote control room. Above this sits the middleware level, a distributed Service Oriented Architecture adapted to Ad-hoc deployments.

B. Surveillance scenario

To exemplify the needs and expected benefits of the architecture, we consider the public safety scenario depicted in Fig. 1 where public safety forces are trying to deal with an explosion in a chemical plant.

Trucks and the respective crew are deployed throughout the area and, in addition to their regular equipment, they bring in sensing capabilities. The sensor nodes, vehicles and crew all form a wireless ad-hoc or mesh network that helps the monitoring, coordination and supervision of operations.

This network is basically composed of three different hierarchical levels. At the higher level, some of the trucks or vehicles present on the scene act as gateways in order to stay in contact with remote command and control rooms using legacy long-range wireless infrastructure networks. Remote

Manuscript received June 20, 2007. This work was supported in part by the French Ministry of Industry under ITEA2 SODA European collaborative project.

M. Lopez-Ramos is with Thales Communications, Colombes, France (phone: +33146133210; fax: +33146132686; e-mail: Mario.lopez-ramos@ fr.thalesgroup.com).

J. Leguay is with Thales Communications, Colombes, France (e-mail: jeremie.leguay@ fr.thalesgroup.com).

V. Conan is with Thales Communications, Colombes, France (e-mail: vania.conan@ fr.thalesgroup.com).

data services can be accessed (or offered) in addition to legacy telephony capabilities.

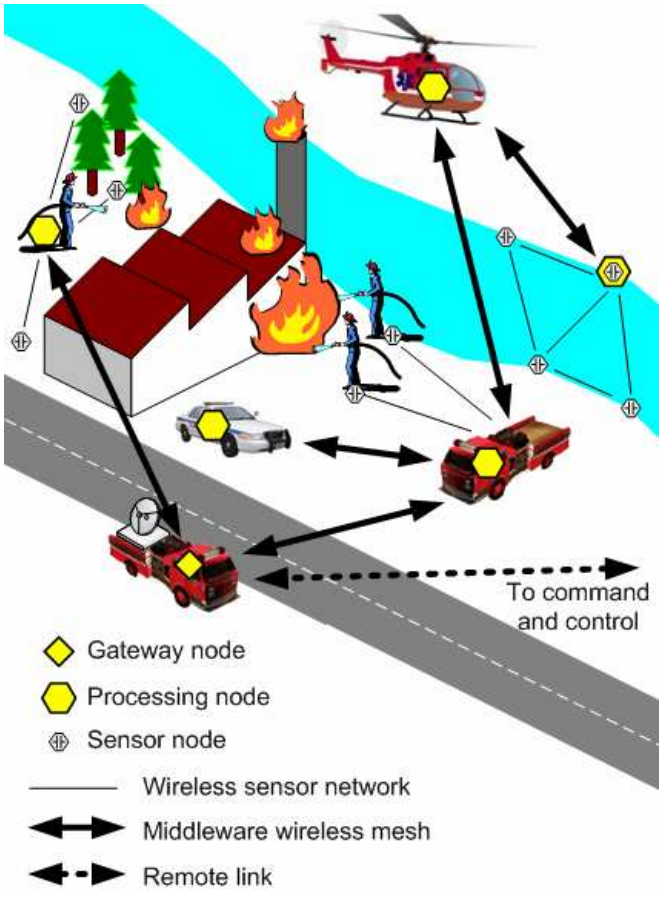

Fig. 1. Homeland security scenario: an explosion in a chemical plant

At the intermediate level, communicating devices embedded in the trucks or carried by the firemen offer data services using a wireless mesh backbone in complement to the traditional push-to-talk radios.

Some of the nodes at this level gather the sensing data provided by dozens of tiny and easily deployable sensors or actuators disseminated locally. In a time-constraining context such as this one, raw sensing information does not provide useful knowledge: these nodes have sufficient hardware resources to offer processing capabilities and transmit filtered information.

These disseminated sensors and actuators constitute the third level that we considered. They could potentially have only limited sensing functionality (e.g., temperature detectors or sirens) or, when equipped with larger batteries, they could perform more complex processing operations and interact with each others (e.g., motion tracking, image processing, and data aggregation).

Any standard or proprietary networking technologies can be used at the different levels of the architecture. Narrowband Private Mobile Radio (PMR) systems such as TETRA or APCO25, could enable the connectivity with legacy infrastructure backhauls, WiMax or WiFi could allow nodes belonging to the wireless mesh backbone to be interconnected, and Bluetooth or ZigBee could help at the sensors/actuators level for internal communications or communications to the local gateways.

To be more concrete, when the chemical explosion occurs, alarm systems from the factory and the surroundings immediately alert the public safety that a fire has started. First, the biological and chemical special safety forces spread from a helicopter a mesh of gas sensors to determine if there is a toxicity or explosion risk before accessing the damaged building. In the meantime, surveillance cameras in the trucks are remotely controlled by the rescue teams to spot the safer path to the victims. Firemen, equipped with gas masks, start to put out the fire. The sensors embedded in their suits continuously monitor vital constants and provide the means to track their location.

Once the situation is under control, pollution sensors might be deployed in the area and the nearby river to determine the magnitude of the toxic leak. Police forces may establish a security perimeter accordingly.

Also of interest, an example of sensors cooperation would be that (1) a mesh of pollution sensors detects abnormal toxicity levels in the area, and (2) a camera listening to events triggered by this detector then focuses on the concerned area to offer to safety forces a live video stream of what is going on. The camera could potentially inlay the video in a map retrieved from a GIS (Geographic Information System) embedded in a truck.

C. Requirements

From the example above and the analysis of other global and homeland security scenarios, we can identify a number of important requirements that need to be addressed by our system architectures:

- Varying unexpected contexts, requiring different equipments, specific sets of sensors adapted to each surveillance or security scenario, call for the support of plug-and-play deployments, including dynamic discovery of nodes and services.
- Network communications often encounter complex constraints such as low bandwidth, high delays or connection failures.
- Time is a critical factor, real-time shared situation awareness and reactivity are crucial.
- Resilience of the sensing services call for a distributed architecture that would support local data fusion and mining processes within the temporarily or dynamically deployed WSN.
- Seamless integration to legacy or infrastructure IT and support of standard or common place programming abstractions to facilitate the take off of such sensor network technologies.

To offer the services and distribute the intelligence we envisioned in the reference scenario above and more generally speaking for surveillance and sensor-enabled security

applications, we believe that a Service Oriented Architecture dedicated to wireless mobile ad-hoc surveillance and security sensor networks is the key enabler. Such an architecture would perfectly suit our needs for complex services, loosely coupled or hierarchically structured. Running above the IP protocol suite, it also provides interoperability facilities and auto-organization mechanisms. Furthermore, it can easily apply to any other surveillance scenarios such as underwater monitoring and domestic or homeland security. The service oriented architecture that we have defined is detailed in the following section.

II. MOTIVATION

The motivation for our proposal comes from the analysis of a number of on-going trends in hardware and software which offer new opportunities for cost-effective solutions to the challenges we identified in the scenario above.

A. Hardware for the processing nodes

Sensor networks typically involve devices ranging from rack-size vehicle-mounted servers to millimeter-size dispersion dust motes [1]. Four main hardware platform classes can be identified for these architectures:

- Special-purpose sensors: tiny and inexpensive low-consumption battery-powered motes.
- Generic sensor nodes: offer a high-level interface to sensor information as well as processing capabilities.
- High bandwidth sensors: streaming-capable audio or video sensors.
- Gateway nodes: provide a link with traditional networks and business applications.

These platforms address different needs – from low-level sensors to data aggregation, analysis and storage services – and do actually coexist in real-world deployments. The processing nodes that we have identified fall in the Generic sensor node category above.

One of the main motivations behind our proposal and architecture is to exploit a growing class of wireless devices, more powerful than a sensor or a mobile handset, and less than a PC or laptop that can provide cost-effective hardware platforms for this category of nodes in the sensor network. Typical target platforms are single-board computers (SBC) based on ARM9 or XScale cores running a minimal GNU/Linux OS. A wide range of commercial products are available, such as Intel Mote 2, StarGate 2 [2], Gumstix [3], and even PC/104 [4] SBCs. These devices are one or several orders of magnitude more powerful than a sensor node, and remain much smaller, cheaper and longer lived than a standard laptop.

B. Middleware for sensor networks

Existing sensor network architectures tend to follow a rigid hierarchical architecture in which high-level processing takes place in a single point that provides monitoring and control capability, whereas sensors or wireless sensor networks are seen as mainly data sources.

Middleware solutions have been identified as a way to increase the capabilities of the sensor networks, by providing more intelligence in the network instead of relying solely on the distant control server [5]. The main purpose of WSN middleware is to support the development, maintenance, deployment, and execution of sensing-based applications. This includes mechanisms for formulating specific sensing tasks, communicating this task to the WSN, distributing it to the individual sensor nodes, and reporting the result back to the task issuer.

The key functions of present day WSN middleware are to allow routing of information in the network (through resource efficient multi-hop radio technologies) and to support powerful yet efficient querying of the sensor data (for example, aggregation primitives and spatial and temporal query primitives). Middleware for WSN focus on providing enhancements for better low level data processing to improve the quality of the specific measurement functions that the WSN is designed to provide.

Some of the solutions for WSN middleware include [6]:

- database-inspired approaches, which use SQL-like queries (shared memory) [7], [8]
- tuple space approaches, which build on the tuple space abstraction made popular by Linda [9]
- publish/subscribe event-based approaches, which use event correlation to aggregate sensor data [10]
- service discovery based approaches, which locate sensors that can meet applications' needs. [11]

In the architecture we advocate the middleware provides a more generic set of programming capabilities, and is not limited to data gathering, access or querying, but aims at supporting many more data processing functions.

Another challenge in sensor network middleware is to support appropriate abstractions and mechanisms for efficient programming of applications that are capable to fully exploit the sensing capabilities of the WSN [5]. For that purpose we propose to adopt the Service abstraction concept that is now well established for server side application integration.

Service Oriented Architectures have been developed for server side application integration and offer a set of tools and paradigms to design distributed applications. They provide lightweight but powerful tools, in particular orchestration or choreography concepts, to build complex distributed applications from individual services distributed on different hosts or nodes. Providing a single abstraction both at the server side and in the sensor network will facilitate take up and adoption of the technology.

Present day SOA technologies are mainly focusing high-end servers running on wired LANs or WANs. But a number of initiatives already provide a first starting point to address our requirements, as we will discuss in the next section.

III. ARCHITECTURE

A. Network topology

This paper takes into account the distributed requirements of sensor networks and the increasing availability of affordable high-performance low-consumption devices to propose a SOA-based middleware where intelligence is deported from a central core to a mesh of autonomous nodes.

This architecture is divided in 3 layers clearly distinguished according to the role played in information production and consumption:

- **The sensor layer**: a mesh network of sensor motes routing simple measurement information to the nearest processing node. Their mere role is to produce information for the next tier.
- **The processing layer**: formed by interconnected intelligent nodes able to consume sensor data and process it to produce aggregated sensing information.
- **The application layer**: consists of applications consuming high-level services from the middle tier and performing heavyweight processing preparing data for human interpretation.

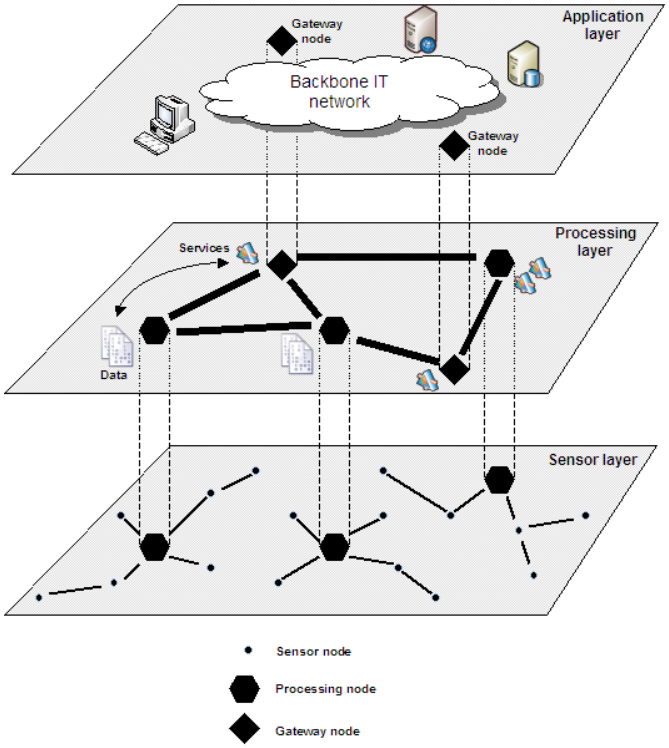

Fig. 2. Layered representation of the architecture. Communication takes place in horizontal layers in the OSI model sense. Processing nodes are present in both the sensor and the processing layers, while gateway nodes ensure the link between the latter and the application layer.

Fig. 2 highlights the role of the processing nodes (represented as hexagons) at the different levels: data sink in the sensor layer; rich information consumer and processing provider in the middle tier; and finally, high-level service provider for end-user applications.

B. The role of the processing layer

The core of the proposed architecture resides in the cross-layer nodes that fulfill several objectives:

- At the sensor layer, each processing node may be connected to a wide range of dedicated sensor nodes using diverse protocols and network topologies. Their mission is to hide the complexity of the often heterogeneous sensing systems by gathering the provided information and transforming it into a common data model based on solid standards such as SensorML [12].
- At the intermediate layer, this information is made available to other processing nodes using a publish/subscribe mechanism according to the type of data, the temporal or spatial resolution, QoS parameters, etc.
- Nodes may host processing functionalities in two forms: hosted services, which are invokable processing routines –ranging from data filtering to service aggregation – made available to other nodes; and hosted tasks, scheduled unattended executing processes that orchestrate the consumption of data and services and use them to provide high-level information. For instance, we might want to deploy a rule that sends an alarm when some is detected and the average temperature of the area reaches a certain threshold.
- Even if the intelligence at the processing layer enables autonomous operation, in most cases real-time availability of captured information is essential at the IT level. Gateway nodes provide an interface between the processing and the application layer, enabling seamless integration of sensing processes into high-level end-user applications. The Sensor Web Enablement (SWE) initiative [13] at the Open Geospatial Consortium aims at creating a “world-wide web of sensors” by fostering interoperability between heterogeneous sensor subsystems.

C. Software architecture of the processing nodes

The software architecture of a processing node consists of 3 major blocks, as shown in Fig. 3:

- The Abstract Sensor Interface (ASI) provides a common mechanism to connect to diverse types of WSNs. An abstraction layer is introduced to hide the hardware and network differences between sensor technologies. Its discovery mechanism communicates with lower layer interfaces and maintain an up-to-date registry of the connected sensors and their metadata. An important issue here is the unification of the diverse data formats.
- The middleware core provides the essential

functionalities for the interaction between nodes. It exposes available information and services through the publish/subscribe interface, enables remote system management and ensure message protection. Access control mechanisms are also provided in order to separate "private" middleware interfaces from "public" services exposed to the application layer.

- The distinctive element that makes processing nodes much more than a single bridge is the hosted intelligence. It encompasses tasks and services as previously described, both built-in (required for standard operation) and hosted (provided by over-the-net dynamically deployable code).

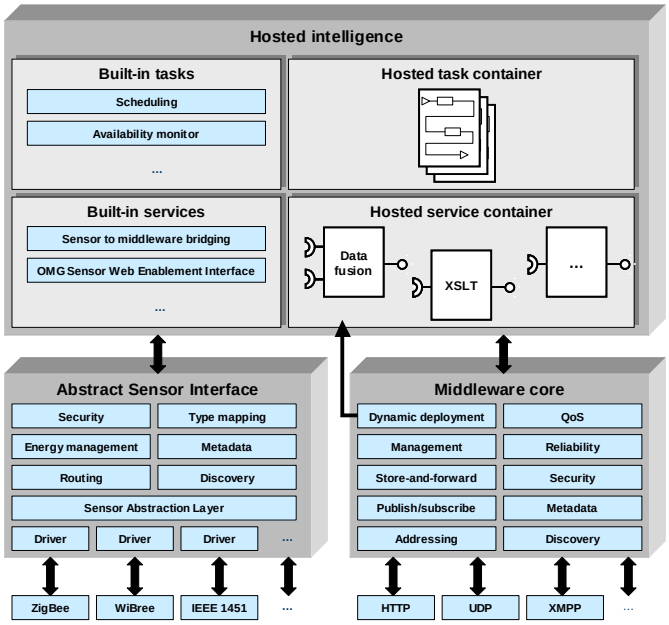

Fig. 3. Software architecture of the processing nodes

D. *Web Services at the sensor level*

The interoperability and interconnection of disparate heterogeneous sensor networks is a matter of growing interest that cannot be left out. Web Services are not regarded any more as a technological hype, but as the ideal implementation to deliver platform-independent SOA. Today, medium-class embedded devices – such as those targeted by the processing layer – are ready to implement such technologies efficiently.

Our protocol stack integrates core Web Services standards such as WSDL, XML Schema and SOAP and adds the following protocols [14]:

- **WS-Addressing** provides a transport-neutral addressing mechanism by including all the message addressing information (from, to, reply to, …) in the SOAP message header, rather than relying on HTTP addressing. This allows the transparent usage of any transport protocol (HTTP, TCP, UDP or XMPP).
- **WS-Discovery** allows plug-and-play discovery of network-connected resources. By defining a specific UDP multicast group, it allows the detection of arrivals and departures as well as the location of resources responding to certain criteria (name, type or scope) on a local network. In our architecture, it provides a distributed mechanism for the discovery of services and data.
- **WS-Eventing** is a simple yet powerful specification that defines a protocol allowing a Web Service to subscribe to another one and receive event notification messages asynchronously. This mechanism provides an N-to-M publish/subscribe mechanism that lays the foundations of the upcoming Event-Driven Architecture (EDA).
- **WS-Policy** allows a Web Service to advertise its policies (on security, Quality of Service…) in the form of "Policy assertions".
- **WS-MetadataExchange** is used for retrieving the metadata associated to a Web Service (such as WSDL or policy information). It is complementary to the discovery mechanism in that it provides the client sufficient information to dynamically search and invoke any service.
- **WS-Management** is a SOAP-based protocol for managing devices across the network.
- **WS-Security** specifies how authentication, integrity and confidentiality can be enforced at SOAP envelope level.
- **WS-ReliableMessaging** provides reliability in the delivery of SOAP messages through sequence control.

The combination of the mentioned specifications address the required functionalities for an autonomous dynamically-deployable and secure information-sharing middleware.

E. *Implementation issues*

One of the advantages of relying on the WS specifications suite is that we can benefit from an active technical community that often provides reference open source implementations.

Many of these implementation efforts are targeted to mainstream applications, and do not address the specific needs of our architecture, especially in terms of footprint and power consumption requirements.

To illustrate that the path that we are proposing and the middle tier specifications that we outlined above are indeed workable in our context, one can refer to the work that is carried out in implementing the Devices Profile for Web Services (DPWS) [15].

Originally published in 2004, DPWS is a profile – a subset of the WS-* specifications – that aims at providing Web Services support to resource constrained devices. The main protocol components of DPWS are WS-Addressing, WS-MetadataExchange, WS-Transfer, WS-Discovery and WS-Eventing. It provides plug-and-play capabilities to services running on devices. Windows Vista integrates DPWS natively (the stack is called WSDAPI) and several other

implementations have been deployed in commercial products, mainly in the printer and automation sectors.

Among these implementation efforts, in January 2005, performance tests were carried out [16] using an open-source implementation [17] based on the gSOAP stack [18]. The report claims that the static memory footprint of the device including the OS, TCP/IP and DPWS stacks, was less than 500 KB, and dynamic memory less than 100KB on a 44MHz ARM7 TDMI processor running ThreadX.. The total time for preparing and sending a message and handling its response was 29 ms, but they report possible improvements that are presently under study [19].

These data offer good confidence that the proposed track is a feasible one. The protocol stack that we envision for homeland and global security would need to include more functionality than what DPWS defines, to deal in particular with security and reliability issues and further studies are required to validate the overall approach.

IV. Conclusion

In this paper we presented a novel architecture that supports complex and resilient surveillance and monitoring applications, deployable on the field and targeting global security needs. We discussed the requirements in Hardware and Software capabilities that such an architecture would rely on. We reviewed and analyzed available equipment and existing COTS software in the domain of Web Services and concluded that implementing the proposed concept is feasible.

We are presently setting up a testbed that implements the architecture with the aim of evaluating it in practice. It consists of radio equipped nodes that form dynamically an ad-hoc network, each node running a Web Service stack. Our objective is to run an example similar to the one provided in this paper, and to demonstrate the robustness and resilience of this cost-effective proposal.

References

[1] J. Hill, M. Horton, R. Kling and L. Krishnamurthy "The platforms enabling wireless sensor networks," *Communications of the ACM*, vol. 47, no 6, Jun. 2004, pp. 41–52.

[2] Intel Imote 2 and StarGate 2 platforms at Seattle's Intel Research Center http://embedded.seattle.intel-research.net/wiki/

[3] Gumstix Inc., http://www.gumstix.com/

[4] PC/104 Consortium, http://www.pc104.org/

[5] K. Römer "Programming paradigms and middleware for sensor networks," *in GI/ITG Fachgespraech Sensornetze*, Karlsruhe, 26-27 Feb. 2004

[6] K. Henrincksen, R. Robinson "A Survey of Middleware for Sensor Networks: State-of-the-Art and Future Directions," In *International Workshop on Middleware for Sensor Networks, ACM International Conference Proceeding Series*, pp. 60-65. ACM Press, Nov 2006

[7] Shen, C. Srisathapornphat, and C. Jaikaeo, "Sensor information networking architecture and applications," *IEEE Pers. Commun*., vol. 8, no. 4, 2001, pp. 52–59.

[8] D. Chakraborty, A. Joshi, Y. Yesha, and T. Finin, "GSD: a novel group-based service discovery protocol for MANETS," in *Proc. 4th International Workshop on Mobile and Wireless Communications Network* (MWCN '02), pp. 140–144, Sep. 2002.

[9] A. L. Murphy, G. P. Picco, and G.-C. Roman, "LIME: a middleware for physical and logical mobility," in *Proc. 21st International Conference on Distributed Computing Systems* (ICDCS '01), pp. 524–533, Phoenix, Ariz, USA, Apr. 2001.

[10] E. Souto, G. Guimarães, G. Vasconcelos, M. Vieira, N. Rosa, C. Ferraz, J. Kelner "Mires: a publish/subscribe middleware for sensor networks," *Personal and Ubiquitous Computing* , Vol 10 , Issue 1 (December 2005), pp. 37 – 44, 2005.

[11] W. B. Heinzelman, A. L. Murphy, H. S. Carvalho, and M. A. Perillo, "Middleware to support sensor network applications," *IEEE Network*, vol. 18, no. 1, pp. 6–14, 2004.

[12] M. Botts, "Sensor Model Language (SensorML): XML-based Language for In-situ and remote sensors", http://vast.nsstc.uah.edu/SensorML/

[13] Open Geospatial Consortium - Sensor Web Enablement Working Group http://www.opengeospatial.org/projects/groups/sensorweb

[14] IBM DeveloperWorks, "Standards and Web Services" http://www.ibm.com/developerworks/webservices/standards/

[15] http://schemas.xmlsoap.org/ws/2006/02/devprof/

[16] F. Jammes and H. Smit, "Service-oriented architectures for devices - the SIRENA view," in *Proc. INDIN'2005 Third IEEE International Conference on Industrial Informatics*, 2005.

[17] Open-source DPWS implementation, Service-Oriented Architecture for Devices Forge, https://forge.soa4d.org/

[18] gSOAP Web Services toolkit, http://www.cs.fsu.edu/~engelen/soap.html

[19] The ITEA2 SODA project, http://www.soda-itea.org/