Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 725190, 11 pages
doi:10.1155/2012/725190

Research Article

Pervasive Smart Spaces and Environments: A Service-Oriented Middleware Architecture for Wireless Ad Hoc and Sensor Networks

Miguel S. Familiar, José F. Martínez, and Lourdes López

Department of Telematic Engineering and Architectures, Technical University of Madrid, 28031 Madrid, Spain

Correspondence should be addressed to Miguel S. Familiar, msfamiliar@diatel.upm.es

Received 30 November 2011; Accepted 27 February 2012

Academic Editor: Ferry Pramudianto

In the twenty-first century, the impact of wireless and ubiquitous technologies is changing the way people perceive and interact with the physical world. These communication paradigms promise to change and redefine, in a reasonably short period of time, the most common way of our everyday living. The continuous advances in the field of Wireless Sensor Networks and their direct application in Smart Spaces are clear examples of it. However, in order for this kind of new generation infrastructures to have a large-scale dissemination, there are still some open issues to tackle. In this way, this paper presents nSOM, a service-oriented framework based on sensor network design that provides internetworking services with the Internet cloud. This lightweight middleware architecture implements an agent-based virtual sensor service approach which is a compact semantic knowledge management scheme based on a dynamic composition model.

1. Introduction

Pervasive computing, also called ubiquitous computing, is becoming more common nowadays. This paradigm, initially thought by Mark Weiser in the early 90s, defines a vision of ecosystems where digital technology and wireless communication devices are highly integrated in objects of the physical world and consequently unnoticeable by the human eyes. The development of personal services on these ubiquitous devices has led to the creation of Smart Environments, which are conceived as an overlay digital infrastructure that proactively, but sensibly, support people in their every daily lives. Smart Environments, also called Smart Spaces, are developed by interconnecting different types of equipments and heterogeneous computing entities, including PDAs, smartphones, handholds, wearables, and Wireless Sensor Networks (WSNs). This type of ad hoc and sensor networks is devised to gather information from the area where they are deployed, and, after being processed, decisions are made and actions on the physical environment are carried out. Applications such as environmental and patient monitoring, object location and tracking, surveillance, and security have been enhanced by means of wireless sensor devices. Embedded architectures for WSNs have evolved. Early sensor-based applications were fully hardware platform and application domain dependent. Due to these reasons, most sensor-based applications for new scenarios are made from scratch.

Most recently, through intermediate stages of object-oriented and software components engineering models, proposals intended to bring the benefits of Service-Oriented Computing (SOC) to the WSNs have begun to emerge. In such manner, the SOC paradigm offers support for the development of rapid, evolvable, interoperable, and easy assembly business application processes, and where services are defined as autonomous, interoperable, and vendor-neutral entities [1, 2]. However, the requirements imposed by pervasive computing in order to create Smart environments require combining the SOC model with other software engineering approaches. On the one hand, these kinds of smart infrastructures make use of wireless sensors and actuator devices in order to model the physical environment. These sensor nodes are resource-constrained devices, in terms of

CPU power, memory, bandwidth, and power autonomy. On the other hand, building Smart Spaces, which offer an added value to users, requires the development of enriched, advanced services. In a Smart Environment, the services offered may appear as a global functionality, instead of a set of isolated simple services, in order to improve the Quality of Experience (QoE) perceived by the end-user.

Applying the issues mentioned above in the design of service-oriented middleware approaches, initial design lines for our contribution are defined, which can be outlined in three main axes. First, simple services offered in the sensor network as in-network agent-based services have been defined, making use of the Separation of Concerns (SoCs) [3] notion. In order to achieve that, the provided sensor-based services are modeled by using agent technology to enhance their autonomous and proactive behavior. In addition, SoC approach is then applied so as to identify and encapsulate the service properties as business lightweight application units, decoupling the in-network services' functionality, and thus generating a low implementation overhead. Second, the description of the services in the WSN has to be defined in an optimized way, using notation schemes that allow proper management of the nodes and network resources. A novel notation proposal based on JSON (JavaScript Object Notation) called SMD (Service Mapping Description) [4] is to be used in order to achieve it. This scheme has been used for both defining service interface contracts and provide RDF (Resource Description Format) [5] compliant semantic annotations. Third, service composition mechanisms in the sensor network using the annotations in the SMD descriptions have been implemented. This method generates abstractions of advanced sensors, which cover various services, a concept called virtual sensor. This notion offers a powerful, flexible tool for the composition of new services. Some potential virtual sensor prototypes are focused on the transport management of goods and food, baggage control at airport boarding areas, and healthcare environments where vital signs of patients can be monitored. Additionally, this virtual sensor technology will allow us, with the in-network agent-based service model, to offer WSN-based services unambiguously and comprehensively in smart spaces, providing intelligent decisions that would then be difficult to get when using isolated physical sensors.

In order to achieve the above challenges, the nSOM (nano Service-Oriented Middleware) approach is proposed. The rest of the paper has been organized as follows. The following section surveys related work. Then, some real applications that have been carried out with WSNs in a proposed space environment scenario are explained. It is then followed by the analysis of the nSOM architecture. The validation results of this approach are then shown. Finally, concluding remarks and future research are presented.

2. Related Work

Middleware frameworks are key elements for creating WSNs-based Smart Spaces, providing interoperability between heterogeneous mote platforms and offering a generic abstraction for developing ubiquitous services. Having the purpose of placing our contribution within the state of the art in service-oriented middleware trends for ad hoc and sensor networks, this section surveys some of the most remarkable and novel approaches in this field.

RUNES [6] is a middleware solution which provides publish/subscribe support and dynamic reconfiguration capabilities in heterogeneous wireless and sensor environments. This approach offers a wide range of middleware services for the development of embedded applications systems, including advertising, discovery, and coordination services. Moreover, this middleware architecture has been evaluated in a real-world road tunnel fire scenario. Nevertheless, the design and implementation of individual software components in RUNES are tasks meant for experts, which may not be easily carried out by an end-user.

MiSence [7] is a component-based middleware for sensor networks, which defines a resource management layer to perform and coordinate tasks for resource sharing in sensor nodes, as well as a set of core services, including aggregation and event detection. One of the most interesting features of MiSence is that it offers an open model for the programming of middleware services, which can be plugged into the framework using MiSence Service Extensions, without modifying the current code. Functionalities regarding network configuration and dynamic adaptation have been taken into account in the definition of the architecture.

Waluyo et al. propose in [8] a middleware for Wireless Body Area Networks (WBANs). This contribution includes features such as data gathering, dynamic plug-and-play support, sensor monitoring, and security mechanisms. This solution has been designed for monitoring vital signs, such as blood pressure, pulse, temperature, and electrocardiogram. Sensed information is transmitted and encrypted to a mobile device, where patient's data can be analyzed. However, its authors do not address the possibility in transferring this approach to other application scenarios, or applying it for building heterogeneous Smart Spaces.

Tiny Web Services [9] is a Service-Oriented Architecture for pervasive embedded devices. This approach makes use of Web Services implemented directly in sensor nodes, by using WSDL descriptions. Moreover, it uses SOAP over UDP in order to achieve high performance in service transactions, as well as WS-eventing to integrate it with Web Services based on Internet applications. However, this approach introduces overhead in the WSN, due to the use of WSDL for describing services and SOAP for application protocol, because both are based on XML data representation scheme.

Sleman and Moeller [10] present MicroSOA, a framework for low-capacity devices implementing service exposure using JSON notation. This proposal defines a software architecture based on IPv4 connectivity in sensor nodes, providing lightweight handler, eventing, and discovery Web services in embedded devices. Additionally, its authors propose the use of a gateway outside the sensor network to execute device and service management function and also network monitoring. However, this middleware does not address operating system and network protocol heterogeneity, it nor offers a programming abstraction to develop services in sensor nodes.

DPWS (Devices Profile for Web Services) [11] is a subset of Web Services standard. This XML-based technology provides Web Services' compliant connectivity for resource-constrained devices, limiting the complexity and message size in the original specification. In the literature, there are several proposals based on DPWS standard. One of the most significant proposals is SOCRADES [12] approach. It defines two alternative contract interfaces: DPWS and Representational State Transfer (REST). On the one hand, REST achieves lightweight service transactions by using HTTP operations, but it lacks service discovery protocol. On the other hand, DPWS reaches high performance in local area networks made up of conventional devices. Additionally, its feasibility for the reduced function devices is important to be pointed out, although it is based on XML transactions, since DPWS is defined as a compact profile of Web Service protocol stack with restrictions of complexity and message size, being optimized for wireless embedded networking. Aside from the previously mentioned points, other novel contributions based on DPWS exist, such as [13, 14], which are focused on providing Web Services interoperability for resource-constrained devices. Referring to [13], the architecture for exposing Web Services for healthcare applications over sensor devices is presented. In [14], its authors propose the use of DPWS in industrial machinery for monitoring and doing diagnosis procedures in personal devices. These two proposals illustrate the DPWS architecture for exposing services to the cloud in detail, but they do not provide a generic middleware framework in the Wireless Sensor Network. Additionally, these approaches are presented as ad hoc solutions for very specific application domains. Previously mentioned limitations make it difficult to provide different nature services, and therefore hindering their migration in order to create heterogeneous smart ecosystems.

The main drawback of the proposals reviewed in this section is the lack of defining an architecture that integrates the provisioning of generic middleware services for the development of pervasive applications and the definition of in-network procedures to semantically compose sensor services and expose them to the cloud by using lightweight notation schemas. Our approach fulfils the benefits of the service-oriented middleware and service composition and exposure paradigms using RDF semantics. Regarding proposals [6–9], our main contribution is to provide a lightweight model of sensor-based exposure and composition using SMD/JSON semantically annotated with RDF compliant triples. This technology is an emerging research topic that can be used for exposing the sensor network to the Internet cloud as a service, through a WSN as a Service approach, in line with the world of Internet of Things (IoTs). Regarding [9, 10, 12–14], our main contribution is to define a full WSN Service Oriented Architecture (SOA), an issue that is not completely provided by previous reviewed sensor network solutions. The use of SOA software engineering paradigm will support the definition of a middleware core with hardware platform and operating system independency, resource-aware management, service configuration, provisioning of generic application oriented functionalities, and heterogeneous multiservice composition support. These features for our proposal are described in detail in the following sections and will lead us to define a generic service-oriented building and composition framework for developing heterogeneous Smart Environments over wireless sensor devices. In our best knowledge, this is a key issue to enhance the deployment of ubiquitous computing-based spaces, as well as to bring this technology to end-users in a natural way.

3. Towards a Smart Infrastructure Scenario Modeling

Smart Environments offer end-users a rich set of facilities to improve their daily life experience, integrating seamlessly with the real-world environment where they are deployed. Under the framework of the DiYSE [15] project, the design and implementation of a Smart Mall have been proposed. The scenario overview is illustrated in Figure 1. In the environment of personal terminals and sensor nodes, the description of its hardware resources and simple services is published on the Internet cloud, using SMD/JSON. End-users, using visual applications installed in their handsets, can compose mashups with the services offered in the network. Thus, final services made by users could involve heterogeneous terminals and devices, fixed and mobile, located in the same or different networks, both wireless and wired.

As a first approach to the scenario, our attention will be focused on a virtual sensor service prototype oriented to leisure and childcare in the smart building precinct. This virtual service can be divided into three use cases: context-aware, location tracking, and perimeter surveillance services. Each of these is introduced in following paragraphs. The definition of these real use cases will allow us to put in context some of the middleware design requirements for the development of advanced digital ecosystems. Among these requirements, real-time support, context discovery, confidentiality and privacy, on-demand service configuration, service composition and exposure to the Internet cloud are emphasized. The description of the middleware service-oriented architecture to support these services is presented in the following sections.

3.1. Real Use Case 1: Context-Aware Behavior. Context-awareness computing offers interesting advantages to the paradigm of sensor networks. On the one hand, the introduction of contextual support in the devices allows the system to retrieve information more easily. On the other hand, applications using this approach are able to adapt their behavior according to the system context, without external interventions.

In our scenario, mobile nodes (i.e., mobile wireless sensor nodes which are physically carried by the end-users in the Smart Environment) will set up a profile, which will shape the user information and activity. At this point, it is important to bear in mind some aspects such as communication security and user information privacy, which may be supported by the middleware. This simple service will be implemented in a Context-Aware agent. In the Smart Mall scenario, children use the information installed in the mobile

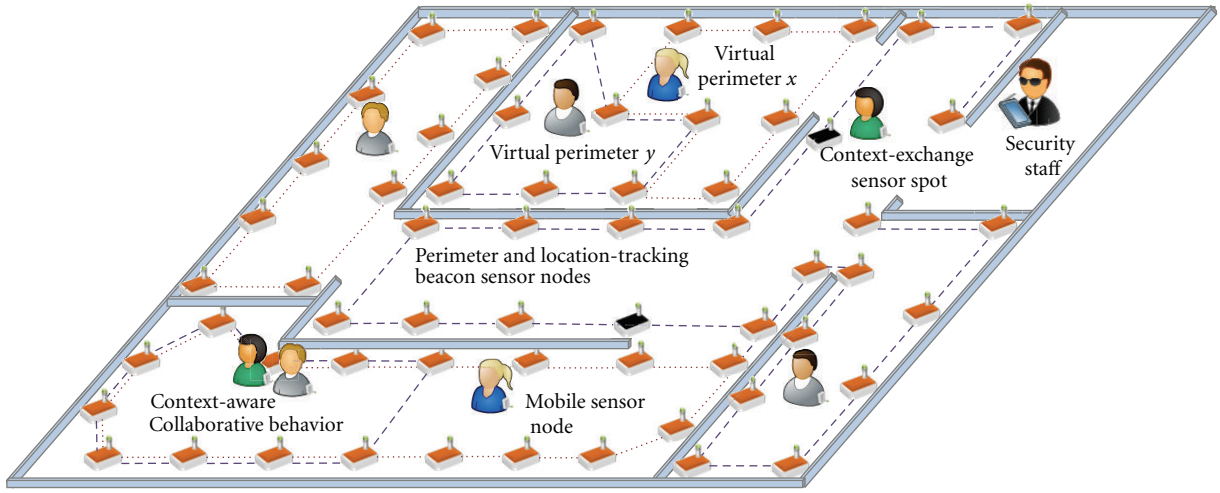

Figure 1: Smart Environment scenario. The smart mall virtual sensor service is provided as the composition of three simple sensor services: context-aware, location-tracking, and perimeter surveillance.

sensor nodes for participating in collaborative activities with other users, as well as interacting with environment elements. An example would be a treasure hunt game, or a gymkhana, where each user can solve some part of the game. Since everybody needs the cooperation of the rest, context-exchange spots (sensor nodes with clues for solving the collaborative game) deployed in the building can serve such purpose. Moreover, the spatial and temporal information of the node and the identification of the associated person will also be useful.

3.2. Real Use Case 2: Location Tracking Service. The position estimation of a mobile wireless sensor node is an essential task in ubiquitous services. Management applications involved in fire fighting, wildlife monitoring, and patient care are some examples of it. This is extensible to the smart infrastructures. However, taking into account the constraints of embedded devices, the correct choice of the suitable location tracking technique is essential for the appropriate system performance.

In such manner, an interesting approach for the particular case of WSNs is to use Received Signal Strength Indicators (RSSIs), in order to estimate the distance between each node and its neighbors. This technique allows a receiver to calculate and determine transmission power, the level of the received signal, and the distance from the transmitter. The main advantage of this solution is that it does not require additional hardware support, so its application does not have a direct impact on increasing the size of the sensors and the cost. From the deployment of beacon or reference nodes in the Smart Mall, each Location-Tracking agent deployed in the mobile sensor nodes can estimate the current position in the deployment. This will supplement context information and support monitoring and surveillance functions of children in buildings.

3.3. Real Use Case 3: Perimeter Surveillance. The surveillance of critical areas is a task that can be implemented in an efficient way with WSNs. This is due in large part to the ubiquitous nature of such devices, as well as its support for ad hoc communications. In this type of application environments, sensor networks enable the deployment of the security infrastructure quickly, with low cost and with a minimum of wired communications.

The technology used in our scenario for implementing the security system is Passive Infrared (PIR) devices. PIR detectors are passive-sensitive elements, which measure the infrared radiation emitted by objects. The main advantage of their use is its own passive features. The power usage is minimal during operation, thereby extending its autonomy. Our proposal is the implementation of a Perimeter Surveillance agent, which will be installed at both nodes connected to the PIR devices making up the security perimeter, and in mobile wireless sensor nodes physically carried by children. The agent algorithm will identify children upon detecting the mobile node through a PIR device. It allows parents to know their children's current location. Moreover, this system will enable the network to know if an unauthorized person is entering a critical security area in real time.

4. The nSOM Architecture

The real impact of Smart Environments in our daily lives is quite limited because existing solutions have some drawbacks that limit their deployment and consolidation at large scale. These issues impose some requirements for new designs in this research area. First, Smart Spaces involve a significant number of logical and physical entities performing different tasks. In such manner, full-SOA solutions are a suitable approach insofar as they offer support for discovery, access, and sharing of the network resources, data, and services, while providing interoperability, loose decoupling, and flexibility in heterogeneous infrastructures with run-time interoperability. Second, new contributions may provide easy and efficient procedures of service internetworking with heterogeneous and external networks. This will allow the

provisioning of more advanced and complete services. Third, programming approaches that ease application development for Smart Spaces by third parties with no experience in ubiquitous and embedded computing should be provided. In this way, our contribution is nSOM (nano Service-Oriented Middleware). This architecture is illustrated in Figure 2. As shown, it is stratified into three layers, which are the following: Physical Device Platform, Service-Oriented Middleware Platform, and Pervasive Service Platform. The discussion of the previous layers is presented below.

4.1. Physical Device Platform. This layer is a virtual abstraction of the physical device and the underlying language of the specific platform. It includes the hardware support and the minimum required low-level software for the deployment of the rest of the architecture. Therefore, this platform encapsulates the hardware device, operating system, and networking stack. The motivation of this layer is to provide transparency and independence of the peculiarities of the operating system and routing protocol stack to the higher levels of the architecture.

4.2. Service-Oriented Middleware Platform. This layer integrates the middleware and the Device Abstraction Layer (DAL). Middleware support comprises procedures for managing lifecycles of components and the interaction between them, providing a set of supplementary services for the application processes. The middleware is located above the DAL abstraction, which provides independence from specific hardware platform. The motivation of this platform is to provide a generic, reusable Service-Oriented Middleware, weakly coupled with the physical platform.

4.3. Pervasive Service Platform. This layer makes use of the Separation of Concerns paradigm, in order to reduce software complexity by identification and encapsulation of the system properties into separate unit blocks. Our proposal takes advantage of this technology for resource-constraint devices and brings this concept to the definition of value-added functionalities from simple in-network agent-based services with well-defined functionality. This will minimize overlaps when in-network services interact between them to provide advanced and aggregated services.

5. Service-Oriented Middleware Platform

The middleware abstraction that has been defined in the nSOM architecture follows business-driven, vendor-neutral, and composition-centric paradigms as its main design lines. These three notions are ideal features regarding the full-SOA proposals. First, the middleware has to provide a framework for implementing business process entities based on distributed services in a fast, efficient, and low-cost way. Second, a device abstraction for decoupling the middleware of a single platform or specific hardware device has to be defined, with the aim of facilitating their development regardless of the vendor. Third, the paradigm of distributed computing systems and middleware services has to be strengthened. The emphasis should be placed on building services from flexible resources that can be added as part of a variety of service-oriented solutions. Taking into account the above issues as shown in Figure 2, within the Service-Oriented Middleware Platform, the following layers are defined: Control Services Layer, Cross-Layer Services, Low-Level Services Layer, High-Level Services Layer, and Service Internetworking Layer.

5.1. Control Services Layer. It represents the middleware core. This layer performs the component's deployment and management, as well as providing intercomponents interaction support using events. Control Services are provided by the following functional entities: nContainer and Eventing Kernel. The nContainer is the middleware framework that offers scheduling procedures for the middleware components as well as the user agents deployed in the Pervasive Service Platform. It performs the life-cycle management of the software entities invoking the instantiation, load, start, stop, resume, terminate, and unload procedures, which may be implemented by all of the architecture components (i.e., middleware and agents). The Eventing Kernel component provides an event-driven internetworking service for supporting internode and intranode communications between the middleware components and the application agents, based on the publish/subscribe interaction. This paradigm offers support for decoupled information from producers and consumers, supporting notifications by the subscription mechanism when an event of interest is generated. This notification to the consumer is performed using callback methods, which are registered in the subscription message.

5.2. Cross-Layer Services. This kind of services provides orthogonal functionalities for the tasks performed in the rest of the middleware components. Identified services in the Cross-Layer are the following: Security and Reasoning. Security component is centered in the provisioning of confidentiality, data integrity, and authentication functionalities for communications, which are basic security services needed in open environment that manages heterogeneous nature information. The provisioning of these services by the Security component relies on 128-bit elliptic curve cryptography, which offers a suitable tradeoff between key size and security level, fulfilling the resource-constrained features of embedded sensor networks. Reasoning component has been designed for performing service inference to the agent-based application processes in the Pervasive Service Platform. This middleware service is based on the published descriptions of hardware and software resources in the local sensor node, supporting dynamic discovery of available agent-based services in the mote platform.

5.3. Low-Level Services Layer. This abstraction layer encapsulates services closer to the Physical Device Platform, which are intensively used during the normal operation of the middleware. These are made up of Real-Time Management, Communication, and Context Discovery Management. Real-Time Management service provides configuration support for operating the real-time capabilities of the system, in

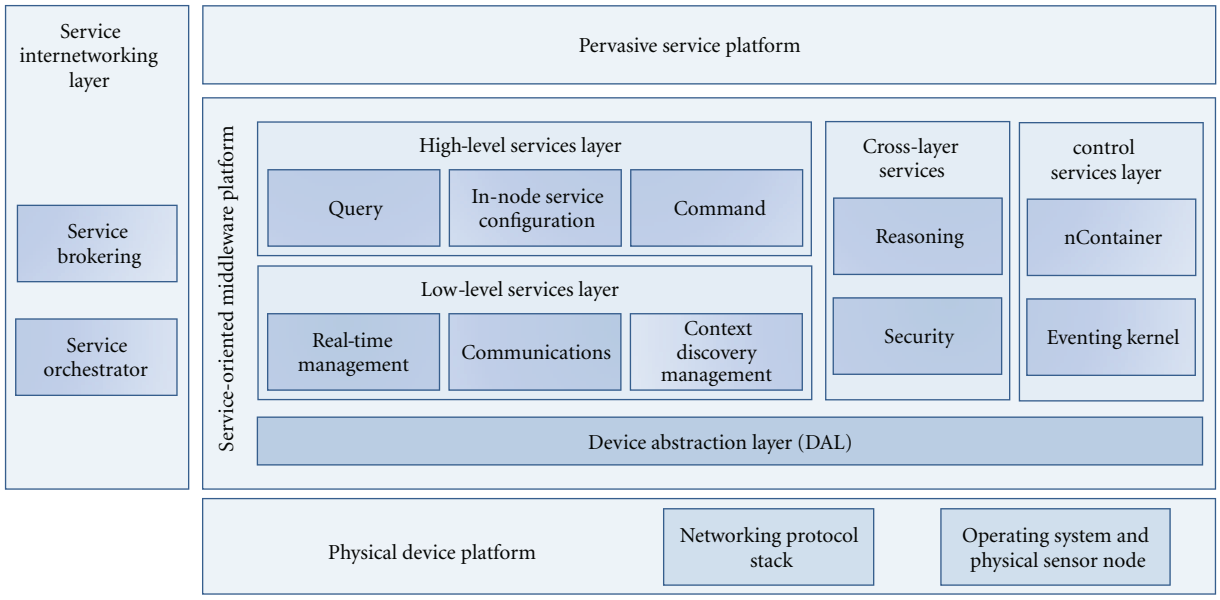

Figure 2: Architecture of the nSOM approach.

function with the simple requirements of the service. This middleware component establishes priorities for the data service in the Eventing Kernel, in order to assure the throughput, delay, and jitter requirements for each service flow according to its traffic parameters. Communication service offers an interface to abstract all the necessary functions for the messages transmission and reception, such as encapsulating and routing. This communication service is used by the Eventing Kernel of the architecture for sending and receiving data events in internode communications. Context Discovery Management implements the procedure for service registering and unregistering, as well as data request/response capabilities, using SMD/JSON key-value pair's notation. According to this proposal, a sample service description used by the Location-Tracking agent is illustrated in Algorithm 1. On the one hand, this description provides both the name of the service as the accessible service primitives for service request and service respond. As shown, each of these operations defines the names and data types of the inputs and outputs, defined as parameters. On the other hand, this contract interface includes RDF notations, in the form of "subject: predicate: object" triples. As shown in the previous figure, the 3-tuple notations are used for attaching the service to higher-level aggregated services, as well as linking the operations to the service and data types to primitive operations. This scheme is used for service composition tasks, specifying how this service is part of a virtual sensor service. During the virtual sensor service orchestration, RDF operation marks will be used for inferring the operations available in the virtual sensor service. This provides a dynamic composition model, depending on the simple services and operations registered in the wireless sensor domain and thus available for its consumption by the end-users.

5.4. High-Level Services Layer. This layer provides high abstraction level services, which are accessible by applications. This provides a programming support layer, which is an important requirement in order to develop an adaptable middleware solution for heterogeneous application domains. Identified services in the High Level are the following: Query, Command, and In-node Service Configuration. Query service provides support for specific sensor data queries, which is a common task in traditional sensor-based data acquisition services. This middleware component provides an SQL-like interface to the agent-based services in the Pervasive Service Platform for performing specific data queries, configuring periodic sensor measurements, and obtaining historical sensor information. Command service has been defined in order to perform a specific management action on remote devices, for both invocation and execution procedures. Interacting with the Control Service Layer, this component will be responsible for executing the process describeed in the received command, including failure recovery, status checking, or code reprogramming. In-node Service Configuration is focused on setting up the sensor devices and modifying dynamically the behavior of the node depending on the activated services, in order to optimize the use of physical resources of sensor nodes. This component includes features such as context information configuration, QoS routing support, and security policies. This middleware service interacts with the Control Services Layer and the Reasoning component to perform the previous adaptation tasks.

5.5. Service Internetworking Layer. This abstraction layer is focused on developing a service composition-centric model in the nSOM architecture. Service Internetworking Layer implements agent-based service composition using the virtual sensor service paradigm, in order to expose and make these available to the external service cloud. To this end, our Wireless Sensor Network contribution includes two distributed network entities, which are the Broker and the Orchestrator nodes. Broker nodes, which are defined in a federated schema

```
{
  "_comment": "SMD/JSON description of Location-Tracking Service",
  "NodeLocationTrackingService": {
  "_comment": "RDF Virtual Sensor Service composition rule",
  "rdfDescription": "subject (this.service)",
  "rdfDescription": "predicate (isPartOf)",
  "rdfDescription": "object (SmartMallVirtualSensorService)"
  "_comment": "Service Request primitive description"
  "NodeLocationTrackingServiceRequest": {
   "target": "getNodeLocationTrackingService.jsp"
   "_comment": "RDF Virtual Sensor Service parameters membership rule"
   "rdfDescription": "subject (this.operation)",
   "rdfDescription": "predicate (hasParameters)",
   "parameters": [
     "rdfDescription": "object (this.parameters)",
     "input": { "name": "sensorNodeAddress", "type": "string" }
   ]
  }
  "_comment": "Service Response primitive description",
    "NodeLocationTrackingServiceResponse": {
     "_comment": "RDF Virtual Sensor Service parameters membership rule"
     "rdfDescription": "subject (this.operation)",
     "rdfDescription": "predicate (hasParameters)",
     "parameters": [
       "rdfDescription": "object (this.parameters)",
       "output": [
           { "name": "sensorNodeAddress", "type": "string" },
           { "name": "rssiSampleSet", "type": "vector",
             "item": { "name": "rssiSample", "type": "string" }
           }
       ]
     ]
    }
  }
}
```

Algorithm 1: Location-Tracking service contract interface with SMD/JSON syntax, annotated with RDF triples.

in order to distribute the computing overload and provide fault tolerance, implement the interface between the wireless sensor domain and the external networks. These nodes are responsible for collecting simple service descriptions provided by the user agents in the wireless sensor nodes and their exposure to the Internet cloud using WSDL 2.0. On the other hand, the Broker devices for receiving service requests from the external network using REST and conducting invocations on each simple sensor node have been defined. Additionally, using the RDF annotations in SMD/JSON service definitions, Orchestrator nodes interact with Broker nodes in order to implement virtual sensor services. In this manner, Orchestrator nodes perform aggregation of simple agent-based services in virtual sensor services, registering them in the designated Broker, as if it were another simple service provided by the sensor network. The network topology diagram that shows these network entities is illustrated in Figure 3. The interaction between the wireless sensor nodes and Service Internetworking Layer is through the designated Broker node, which is discovered by the sensor nodes using an advertisement/request-response hybrid approach. Once the wireless sensor node is attached to the designated Broker, the interaction between them is supported by the event-driven publish/subscribe mechanism. In this way, agent-based services can register and unregister the simple services they provide in the Broker node, receive the service invocations, and send the service response.

6. Experimentation

This architecture has been evaluated in the SunSPOT [16] platform, using Java as the underlying language for developing the prototype. The reasons behind for using this hardware are as follows. Firstly, Java Virtual Machine that runs on SunSPOT and is referred to as Sqwak, is a compact virtual machine that targets small and resource-constrained devices, supporting object-oriented and multi-thread programming. Secondly, this platform offers an Integrated Development Environment that eases the tasks of developing and debugging codes on nodes. This helps in the development of pervasive applications by third parties and promoting the creation of Smart Spaces with multiple services. Thirdly, these nodes

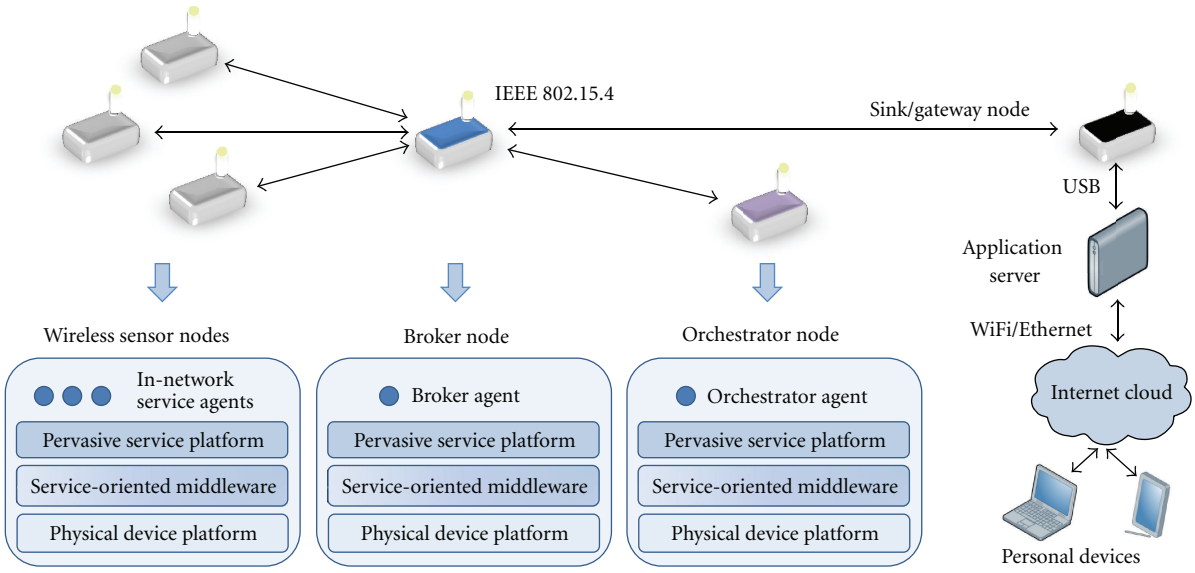

Figure 3: Wireless Sensor Network topology and its internetworking with the Internet cloud.

have significant hardware resources, a fact that will support the deployment of more complex architectures and services. SunSPOT motes are equipped with a 32-bit microcontroller, 4 MB Flash and 512 KB RAM memory. Network protocol used in our implementation is 6LoWPAN (IPv6 over low-power Wireless Personal Area Networks), which is optimized for embedded devices, providing IP connectivity to pervasive environments based on IEEE 802.15.4.

The validation has been performed in the campus of the Technical University of Madrid, implementing a small prototype of the Context-Aware, Perimeter Surveillance, and Location-Tracking agent-based services. The deployment testbed for the evaluation is made up of 26 SunSPOT nodes, a laptop with Apache Tomcat server 6.x performing as an application server, and an Android smartphone being used as personal/monitoring device. The setting up of these wireless sensor nodes is the following. First, there are 2 user mobile wireless sensor nodes where Context-Aware, Perimeter Surveillance, and Location-Tracking agents are installed. These agents implement the functionality of the pervasive services in the Smart Environment. Second, there are 12 Location-Tracking beacon wireless sensor nodes divided in 2 location grids. In these beacons, part of the Location-Tracking agent business logic is installed, which will send on demand (i.e., publish/subscribe procedures) the RSSI measures to the mobile sensor node. It will send the received measures from the beacon nodes to the personal/monitoring device, in order to estimate the mobile sensor node position in the deployment according to the signal strength indicators. Third, there are 8 Perimeter Surveillance wireless sensor nodes, each one connected to PIR device delimiting a physical perimeter in a static way around the deployment area. In these perimeter sensor nodes, part of the Perimeter Surveillance business logic is installed. Once a presence in the deployment is detected by means of the excitation of two adjacent PIR devices, mobile sensor nodes in the intersection area of the PIRs are requested to send its identifier, which is delivered to the personal/monitoring application in order to identify the access type (i.e., authorized or unauthorized) and the user who crosses the virtual perimeter. Finally, this deployment is completed with 2 wireless sensor nodes that perform tasks as Broker/Orchestrator for service exposure and composition in virtual sensor services into the wireless sensor domain, as well as 2 other wireless sensor nodes as sink/gateways to expose the simple agent-based services and the virtual sensor services to the Internet cloud. The interaction between all the previous networking sensor devices and the agents deployed in them is based on the event-driven publish/subscribe mechanisms implemented by the nSOM middleware architecture. The data events are encapsulated over 6LoWPAN packets, which are transmitted over IEEE 802.15.4 frames.

In order to evaluate our contribution, a comparative analysis with DPWS approach has been performed. As previously mentioned, DPWS is a Web Service compliant framework devised for reduced function device communications that is applied in several novel research initiatives in the field of wireless sensor networks computing. The analysis of our service framework with DPWS is mainly motivated in order to be compared with a related standardized native SOC approach suitable for embedded computing. In this regard, and in a similar way to our contribution, DPWS is platform independent, providing support for rapid and simple development of lightweight services, which are perceived as autonomous, dynamic, and interoperable business process entities. In our study, WS4D-JavaME [17] implementation of the DPWS approach has been used as reference. This is an open-source toolkit, which defines a small subset of JavaME configurations. Our testbed approach has been based on the minimal core-based package for creating DPWS-enabled embedded devices with an event-based behavior.

The hardware and software configuration in the wireless sensor nodes used for evaluating the nSOM architecture with

Table 1: Evaluation metrics: (a) memory overhead for service-oriented middleware subsystems; (b) memory overhead for software agents.

(a)

	Flash footprint	RAM heap footprint	Source code lines
Low-Level Services Layer	16960 bytes	511 bytes	5360
High-Level Services Layer	14653 bytes	348 bytes	4761
Control Services Layer	24586 bytes	648 bytes	7280
Cross-Layer Services	18760 bytes	410 bytes	5884
Service Internetworking Layer	1125 bytes	189 bytes	392

(b)

	Flash footprint	RAM heap footprint	Source code lines
Context-Aware	1064 bytes	268 bytes	360
Location-Tracking	3815 bytes	956 bytes	1042
Perimeter Surveillance	1433 bytes	387 bytes	497
Broker/Orchestrator	1952 bytes	184 bytes	623

the RDF-based SMD/JSON proposal and the DPWS implementation has been the same, including SunSPOT mote platform, IEEE 802.15.4 physical and medium access radio technology, and 6LoWPAN network protocol. Performance metrics that have been analyzed in the validation are the following: memory overhead, service registering time, service request/response time, and virtual sensor service completion time. The evaluation of our contribution is based on three main aspects. First, how heavy the SOA approach is, has been analyzed in terms of memory footprint and code, in order to fulfill the resource-constrained features of the sensor nodes. Second, the improvements achieved by SMD/JSON against DPWS/XML have been studied in order to describe services in the wireless domain. Third, the power of using semantic annotations for service composition process has been discussed. Additionally, since multihop configurations are common in ad hoc networks, routing configurations with 1, 2, and 3 hops are considered.

Table 1 illustrates the results obtained for the memory overhead metric, regarding service-oriented middleware subsystems and agent-based services. Considering the full Service-Oriented Middleware Platform, the code in Flash memory is 74.3 KB, and the data is around 2 KB in RAM. These values represent 1.8% of Flash and less 0.4% of ROM considering the available memory in SUNPOT. Regarding the agents, the mean memory footprint value in Flash is 2 KB and 0.48 KB in RAM, which represents 0.05% and 0.1% of the available memory in Flash and RAM, respectively. These results show a lightweight implementation of the middleware core and the pervasive service platform, which is an essential requirement for embedded architectures in wireless sensor networks. Considering DPWS implementation, the code in Flash has been 48.4 KB, while the footprint in RAM has been 3 KB. Comparing these results with the results obtained in nSOM approach, a higher impact of our approach in ROM memory can be observed. This is mainly based on the more advances functionality that is provided in our proposal. These include, as previously mentioned, generic middleware services for application development (i.e., context discovery and real-time support, query/command, service configuration, and cross-layer functionalities such as security and reasoning), as well as virtual sensor service internetworking capabilities for brokering and orchestration. These features are not covered by DPWS due to an important lack for building pervasive smart environments in a service-oriented manner.

Regarding service registering process performed by the sensor nodes in the Broker, results are shown in Figure 4(a). These show an average of the three agent-based services. Accordingly, the SMD/JSOM registering time has reduced in 284 ms for 1-hop, 621 ms for 2-hop, and 938 ms for 3-hop routing configuration, compared with DPWS/XML implementation. These results represent an average improvement of 54% in service registering process regarding DPWS/XML. Considering the average time for service request/response interaction, which is illustrated in Figure 4(b), SMD/JSON data messages interaction has reduced the marks achieved by DPWS/XML in 133 ms, 278 ms, and 435 ms, for 1 hop, 2 hops, and 3 hops, respectively. From these results, SMD/JSON implementation has reduced service request/response marks in 48% as obtained with DPWS/XML. Regarding virtual sensor instantiation metric results, which is shown in Figure 4(c), this analysis has been focused on measuring the time required to execute workflow in order to build the virtual sensor service. This includes the time elapsed from the time the Orchestrator receives the description of the agent-based services from the Broker node up to the time the Orchestrator processes these service descriptions and generates virtual instance. Considering SMD/JSON, this time has been measured as 51 ms, which represents an improvement of 41% regarding DPWS/XML implementation. This optimization is associated with the fact that SMD/JSON service description processing is more efficient than DPWS/XML. Additionally, virtual sensor instantiation workflow with SMD/JSON takes advantage of RDF triples, which drives the composition process, rather than sequential processing of the entire documents, as it is in DPWS/XML. This allows a

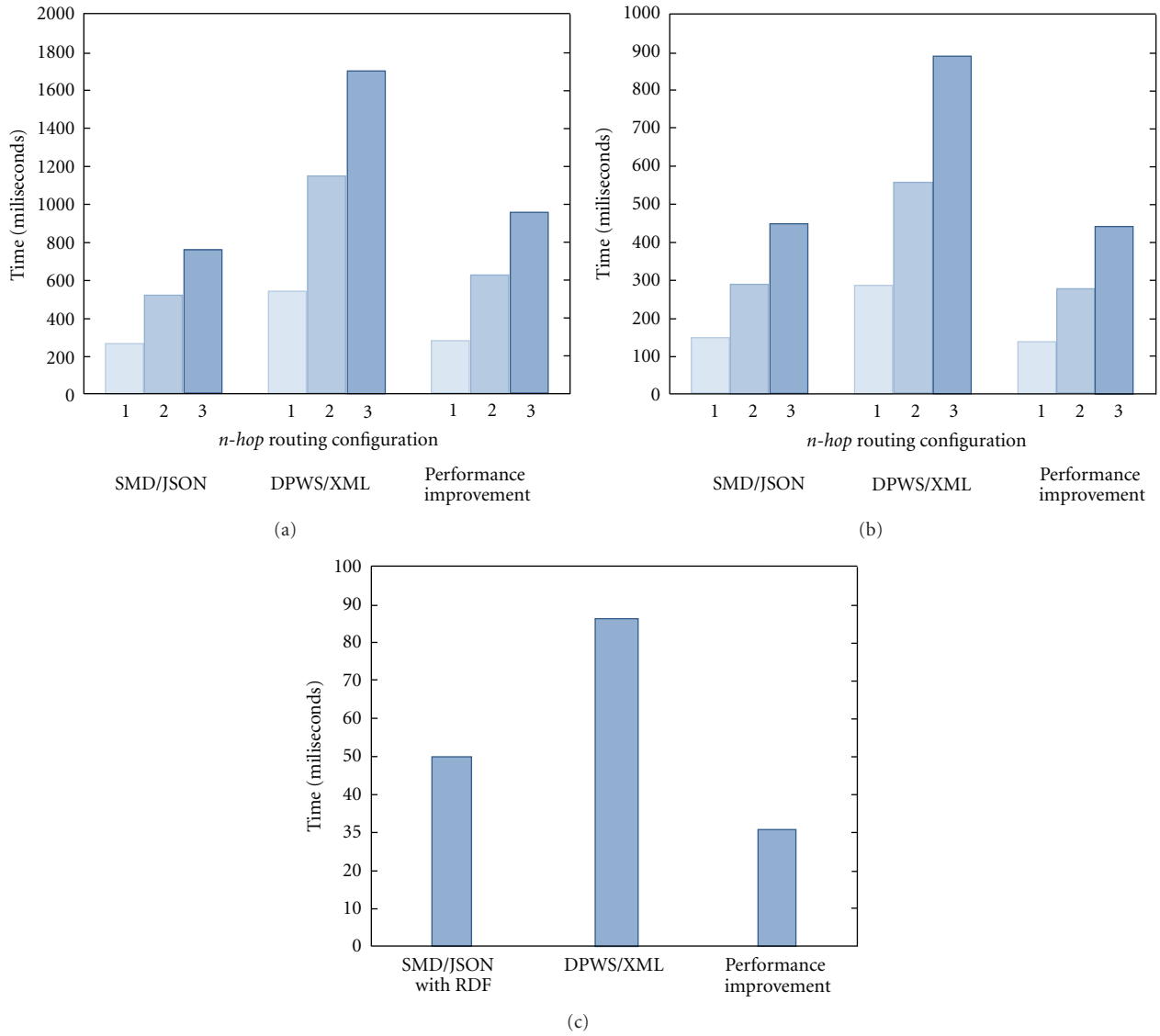

Figure 4: Experimental results: (a) service registering time; (b) service request/response time; (c) completion time for virtual sensor service instantiation.

dynamic composition procedure that is not supported by traditional nor semantically annotated description languages, such as DPWS/XML.

Pervasive smart spaces have an important challenge in response time and latency. These are factors that play crucial role in Smart Spaces. This is due to the interactive nature of such infrastructures, where end-users demand minimal delay for obtaining response from their environmental stimulus. Services with context-aware dependencies, location-tracking estimation, or with perimeter surveillance applications require high time response performance in the middleware as shown in the modelling of the smart scenario. Considering 3 radio hops (i.e., the most restrictive network configuration in our validation scenario), the service registering time for SMD/JSON is under 1 second, the request/response interaction time is below 0.5 seconds, and the completion time for virtual sensor instantiation is approximately 50 ms. Taking into account the obtained results and comparing them with the DPWS middleware implementation, suitable executing conditions have been achieved. Moreover, before and during the performed evaluation, the communication overhead was observed as controlled with no signs of severe congestions. During the validation, data rates were almost 10 packets/sec for each Broker/Orchestrator node while a Packet Delivery Ratio (PDR) was greater than 0.95 in 3 radio-hop routing configuration.

7. Conclusion and Future Research

This paper presents nSOM, a full Service-Oriented Architecture that implements service internetworking using agent-based virtual sensor service abstractions in pervasive Smart Environments. This proposal supports service definition in

resource-constrained devices using RFD-compliant notations to specify the triples applied in the orchestration process. A comparative analysis with DPWS/XML for the service-oriented middleware core and prototypes of Context-Aware, Perimeter Surveillance, and Location-Tracking services is provided. The experimental results obtained show that nSOM offers a lightweight service-oriented framework for sensor networks, with compact service exposure and dynamic virtual service composition through the use of semantic tags. As part of the future research inside the framework of DiYSE project, this architecture will be validated in a Smart infrastructure in Helsinki, Finland. This future experience will be focused on analyzing the performance and scalability of our approach under stress conditions in an open environment and also on understanding the impact and usability of these new generation smart ecosystems in common everyday living. In order to encourage the definition and implementation of third party sensor services, the developed inmote software code in the project is planned to be distributed to the research community under the European Union Public License (EUPL).

Acknowledgments

The work presented in this paper has been partly funded by the European Union under the ITEA DiYSE: *Do-it-Yourself Smart Experiences* (ITEA2, code: 08005) project [16]. Initial deployment works of this project achieved the Exhibition Award of the ITEA & ARTEMIS CO-summit 2010 event, held in Ghent, Belgium. Additionally, authors would like to thank the anonymous reviewers for their valuable comments and suggestions, which have helped improve the present work.

References

[1] M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, "Service-oriented computing: stateof the art and research challenges," *IEEE Computer*, vol. 40, no. 11, pp. 38–45, 2007.

[2] F. M. T. Brazier, J. O. Kephart,, H. Van Dyke Parunak, and M. N. Huhns, "Agents and service-oriented computing for autonomic computing: a research agenda," *IEEE Internet Computing*, vol. 13, no. 3, pp. 82–87, 2009.

[3] K. Hoffman and P. Eugster, "Cooperative aspect-oriented programming," *Science of Computer Programming*, vol. 74, no. 5-6, pp. 333–354, 2009.

[4] M. A. Chatti, M. Jarke, and M. Specht, "PLEF: a conceptual framework for mashup personal learning environments," *IEEE Learning Technology Newsletter*, vol. 11, no. 3, 2009.

[5] J. Hendler, "Web 3.0 emerging," *Computer*, vol. 42, no. 1, pp. 111–113, 2009.

[6] F. Oldewurtel, J. Riihijärvi, K. Rerkrai, and P. Mähönen, "The RUNES architecture for reconfigurable embedded and sensor networks," in *Proceedings of the 3rd International Conference on Sensor Technologies and Applications*, pp. 109–116, June 2009.

[7] K. Khedo and R. K. Subramanian, "A service-oriented component-based middleware architecture for wireless sensor networks," *International Journal of Computer Science and Network Security*, vol. 9, no. 3, pp. 174–182, 2009.

[8] A. B. Waluyo, I. Pek, X. Chen, and W. S. Yeoh, "Design and evaluation of lightweight middleware for personal wireless body area network," *Personal and Ubiquitous Computing*, vol. 13, no. 7, pp. 509–525, 2009.

[9] B. Priyantha, A. Kansal, M. Goraczko, and F. Zhao, "Tiny web services: design and implementation on interoperable and evolvable sensor networks," in *Proceedings of the ACM/IEEE Conference on Embedded Networked Sensor Systems*, pp. 253–266, 2008.

[10] A. Sleman and R. Moeller, "Micro SOA model for managing and integrating wireless sensor network into IP-based networks," in *Proceedings of the 2nd International Conference on Computational Intelligence, Communication Systems and Networks*, pp. 137–142, July 2010.

[11] G. Moritz, E. Zeeb, S. Prüter, F. Golatowski, D. Timmermann, and R. Stoll, "Devices profile for web services in wireless sensor networks: adaptations and enhancements," in *Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation*, pp. 1–8, September 2009.

[12] D. Guinard, V. Trifa, S. Karnouskos, P. Spiess, and D. Savio, "Interacting with the SOA-based internet of things: discovery, query, selection, and on-demand provisioning of web services," *IEEE Transactions on Services Computing*, vol. 3, no. 3, pp. 223–235, 2010.

[13] G. Moritz, E. Zeeb, F. Golatowski, D. Timmermann, and R. Stoll, "Web services to improve interoperability of home healthcare devices," in *Proceedings of the 3rd International Conference on Pervasive Computing Technologies for Healthcare*, pp. 1–4, April 2009.

[14] N. L. Fantana and T. Riedel, "A pragmatic architecture for ad-hoc sensing and servicing of industrial machinery," in *Proceedings of the 7th International Conference on Networked Sensing Systems*, pp. 165–168, June 2010.

[15] M. Roelands, J. Plomp, D.C. Mansilla et al., "The DiY smart experiences project—a European endeavour removing barriers for user-generated internet of things applications," in *Architecting the Internet of Things*, Springer, Berlin, Germany, 2011.

[16] Sun Small Programmable Object Technology (Sun SPOT), Theory of Operation Part no. 820-1248-10, Revision 1.5., 2009.

[17] E. Zeeb, G. Moritz, D. Timmermann, and F. Golatowski, "WS4D: toolkits for networked embedded systems based on the devices profile for web services," in *Proceedings of the 39th International Conference on Parallel Processing Workshops*, pp. 1–8, September 2010.