International Journal of Computer Science and Applications,

Vol. 8, No. 2, pp. 38 – 58, 2011

SEMANTIC WEB BASED ARCHITECTURE FOR MANAGING HARDWARE HETEROGENEITY IN WIRELESS SENSOR NETWORK

SINIŠA NIKOLIĆ

Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, Novi Sad, 2100, Serbia
sinisa_nikolic@uns.ac.rs

VALENTIN PENCA

Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, Novi Sad, 2100, Serbia
valentin_penca@uns.ac.rs

MILAN SEGEDINAC

Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, Novi Sad, 2100, Serbia
milansegedinac@uns.ac.rs

ZORA KONJOVIĆ

Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, Novi Sad, 2100, Serbia
ftn_zora@uns.ac.rs

Wireless Sensor Networks (WSNs) are vivid and relatively new paradigm. Usually they are built for the specific purposes which, in the case of large systems of WSNs, can result in high level of hardware heterogeneity. In this paper an ontology-based model aimed at resolving the hardware heterogeneity in large systems of WSNs, and the GLOSENT architecture that facilitates integration of heterogeneous WSNs are proposed. As a proof of concept a concrete SN device SunSPOT is represented using proposed ontology.

Keywords: Wireless Sensor Networks, Heterogeneity, Ontology, Middleware, SunSPOT.

1. Introduction

WSN is a network of spatially distributed autonomous sensors and it can include hundreds of thousands sensor nodes (SN). SNs are low cost, battery-powered and reduced size devices with limited processing, sensing and wireless communication capabilities aimed at completing specific application task(s).

Most of the current WSNs are built for specific purposes, with a tight coupling between the physical network components and end user applications. On the other hand, contemporary applications require WSN not to be considered an isolated entity, but rather a part of a larger System of Wireless Sensor Networks (SWSN) in which end user applications interact with heterogeneous WSNs.

Heterogeneity of the hardware, variable acquisition protocols, diverse user application requirements, scalability and needs for prolonging the lifetime of WSNs are the main problems faced by the SWSNs. The existing solutions typically focus single aspect of the problem [Terfloth, *et al*. (2009)].

In this paper we propose the model aimed at dealing with heterogeneity issues in SWSN which is based on Semantic Web approach. We propose specific middleware that mediates communication for other parts of the SWSN based on the exchange of ontologies. Here, ontologies are used for the semantic representation of WSN components and the observed data.

The focus of the paper is the hardware heterogeneity of WSN components.

The rest of the paper is organized as follows: Section 2 presents a brief overview of the related work. The proposed architecture for facilitating integration of the heterogeneous WSNs is presented in Section 3. Section 4 describes the WSN ontology aimed at resolving hardware heterogeneities. Section 5 presents the proof-of-concept through the concrete model of the SUNSPOT SN. Section 6 concludes the paper and indicates the future work.

2. Related Work

There are two main approaches aimed at linking network and applications in WSNs.

In the first approach (*specific based*) [Jang, *et al*. (2008)] the whole system relies on the immutability and the specificity of application requirements and the environment in which WSN is implemented. In such systems, even small modifications in application's requirements or in the structure of WSN require changes that typically need to be implemented all across the system. This approach sacrifices generality for simplicity of implementation.

The second approach (*generic based*) includes the middle layer (*middleware*) [Terfloth, *et al*. (2009)]. Forasmuch as devices forming a WSN have only modest capabilities in terms of processing power and memory (hardware characteristic), their primitive operating systems only provide a basic support for integration with other software components. Middleware compensates that deficiency, providing for abstraction and additional services. With a middleware included, applications implement their functionality in a more abstract manner. In [Hadim, and Mohamed, (2006)], [Masri, and Mammeri, (2007)], [Molla, and Ahamed, (2006)] the classifications of middleware approaches for WSN are described. The classification proposed in [Masri, and Mammeri, (2007)] identifies the following main middleware classes: *database approach*, *event-*

based approach, *application driven approach*, *modular approach* and *virtual machine approach*. The rest of this section presents a brief overview of several characteristic implementations.

In *TinyDB* [Madden, *et al.* (2005)] the network established by the sensor nodes is understood as a distributed database which can be queried using a subset of SQL. Thus, to obtain a sensor data from the network, the user issues a query which is then automatically routed to all nodes and processed by node's query processor. Here, heterogeneity of SN hardware may appear as a problem, because the *TinyDB* resides as an application on the top of *TinyOS* operating system [TinyOS, (2000)] at SN, and many nodes don't support *TinyOS*, so the heterogeneity is partial. Also, *TinyDB* does not allow to sense and react upon the certain event.

Mate [Levis, and Culler, (2002)] and *Squawk* [Simon, et al. (2006)] WSN middleware use a Virtual Machine (VM) approach where Mate is implemented on the top of *TinyOS* and *Squawk* is JAVA based VM. VM approach to sensor nodes presumes that applications have to be written in specific language. In most cases VMs are supported by a restricted set of SN hardware platforms. For application changes the both abovementioned solutions propose a spectrum of reprogram actions, from adjusting simple parameters to uploading complete program updates using a VM. Uploading operation cases large energy consumption, which makes heterogeneity problem in this middleware even more chargeable.

Impala [Liu, and Martonosi, (2003)], which uses event-based programming model, provides a mechanisms for network updates that are sufficiently efficient to support dynamic applications. This modular approach uses a mobile code that is injected in the sensor network; the code enables collection of the local data and migrating and copying itself to other nodes. The problem is the nature of its predefined instruction set that doesn't allow hardware heterogeneity.

MiLAN [Murphy, and Heinzelman, (2002)], which belongs to adaptive middleware, receives a description of application requirements, monitors network conditions and optimizes sensor and network configurations in order to maximize network lifetime. This application driven approach introduces a new dimension in the middleware design by supplementing an architecture that tightly couples application execution with network protocol stack. Fine tuning of the network requires close relation with application and may lead to a specialized middleware. The communication is restricted only to the hardware supporting that specific network protocol stack which is a serious drawback in terms of hardware heterogeneity.

Mires [Souto, *et al.* (2005)] is the event-based, message oriented distributed middleware that implements a publish/subscribe paradigm in WSN. It uses a strong asynchronous communication model where consumers and producers are loosely coupled. In that system SNs advertise what data they can provide, while applications subscribe to data. *Mires* middleware is built upon *TinyOS*, therefore sharing the same hardware heterogeneity problems.

In [Ramos, *et al.* (2007)] an implementation of WSN, based on Service Oriented Architecture (SOA) model and Sensor Web Enablement (SWE) standards [The Open Geospatial Consortium, (1994)], is described. The architecture of that system is split in 3 separate layers (information production, information routing, information consumption) consisting of two SN types and a Sync node (a gateway). Simplified SN integrates services to retrieve data from the network, while advanced SN integrates services for filtering, aggregation and routing of the data. Service architecture is modeled only for WSN, while user application is considered a consumer of the top level layer services. The hardware heterogeneity problem is resolved by XML descriptions in accordance with SWE standard. There is no shared knowledge about services and the semantic web technologies are not utilized.

The *OX-Framework* [Brőring,, *et al.* (2009)], which relies on OGC standards [The Open Geospatial Consortium, (1994)] for presenting geospatial data, is an example of Web technologies utilization in WSNs. The OGC's SWE standards enable developers to make all types of sensors, transducers and sensor data repositories discoverable, accessible and useable via Web. The purpose of the framework is integration of the real time or near real-time measurements with conventional geospatial data (e.g. maps) for visualization purposes. *OX-Framework* relies on service approach, where entities of WSN communicate by invoking different services of the system. The framework provides discoverable data repositories that include hardware descriptions, but does not include semantic descriptions and does not use semantic technologies.

One example of SWSN where the middleware layer uses Semantic Web technologies is presented in [Liu, and Zhao, (2005)]. Here, the middleware deals with semantic data integration and does not consider any other type of heterogeneity. The sensor data are used to reconstruct (by reasoning process) the context of an event. The core of aforementioned solution is the hierarchical organization of the semantic information and semantic services. The semantic services are connected through *input* and *output* ports constituting a hierarchical graph. Such organization enables creation of the hierarchical structure in which the user queries are transformed into composite service graph aimed at automated planning of the services usage. Activations of the particular services are based on their positions in the graph. The ontology contains semantic entities (context information) and relations between an event and object. Other WSN issues (e.g. data streaming pattern, network elements description, description of the network structure, etc.) are not described by the ontologies. The system architecture is a three-tier one, consisting of sensors, field servers and gateway servers. Each field server in the middle layer is directly responsible for the particular group of sensors. The collected measurements are converted into XML format by servers and the XML data is passed to other levels of the system.

In the paper [Avancha, *et al.* (2004)] ontologies are used for enabling an adaptive WSN, i.e. optimal sensor nodes operating modes. The change of the operating mode reduces the influence of the external factors and prolongs the lifetime of the SN.

Adaptation is achieved at the base stations/Rich uncles that are using ontologies to represent sensor nodes. The ontology contains complete description of the SN regarding its components and their characteristics. The processor, radio, battery and sensor modules are the top ontology elements, while the characteristics of these elements are described by the corresponding subcomponents. The reasoning is based on observed and expected data that are also represented by the ontology.

In the paper [Gomez, and Laube, (2009)] contextual ontologies are used to increase middleware's information processing flexibility. Authors argue that higher flexibility in information processing can be achieved by classification, characterization and setting up the relations among the pieces of the contextual information. Here, the classification is aimed mainly at providing the context-related subtypes of some general concept. Information characteristics are related to its value, unit, origin, and creation time which can be used afterwards by user applications in order to have further information on the delivered sensor data. Establishing relationships between contextual information enables further data processing. The hardware heterogeneity issues are not the subject of this paper.

The paper [Kim, *et al.* (2008)] proposes the sensors ontology for services' representation in Ubiquitous computing systems. The ontology describes three interrelated classes (*ServiceProperty*, *LocationProperty* and *PhysicalProperty*) that should be instantiated by arbitrary ontological descriptions of the sensor nodes. The paper also mentions possibility to use ontology for representing physical sensors' information within the sensor ontology based on service oriented property of sensor data.

The paper [Zafeiropoulos, *et al.* (2011)] deals with problems in data gathering and information management in sensor networks. It proposes generic model for efficient data management which is based on the flexible three-tiered architecture (*Data Layer*, *Processing layer* and *Semantic Layer*). The architecture is flexible in a way that different implementation strategies applied to one level do not affect implementation strategies to the other ones. The contextual annotation of the gathered data by concepts from specific application domains forms the system's semantics. The abovementioned annotation is done by mapping measured values to a selected ontology and defining semantic rules (transformations). These transformations enable creation of a new information, actions and alerts by reasoning over existing information. Ontological description of the SN and WSN management issues are not in the subject of the paper.

In the paper [Barnaghi, *et al.* (2009)] a semantic model for ontological representation of the heterogeneous data from diverse sensors is proposed. The data ontology describes simple and complex data types which are compatible with data types in *SensorML* language [SensorML, (2010)]. It is emphasized that the proposed ontology is only a part of the larger WSN ontology that is merged with an arbitrary upper ontology describing WSN components. The measurement units in this ontology are represented by the *NASA SWEET* ontology [NASA SWEET, (2011)].

Since in SWSN the hardware heterogeneity is highly dependent on quantitative data exchange, representation of the units of measure is of particular importance. Therefore, in the sequel, the brief overview of the efforts aimed at ontological representation of units of measure is given.

The early paper [Gruber, and Olsen (1994)] presents the ontology *EngMath* for mathematical modeling in engineering which, among others, includes conceptual basis for units of measure. In this ontology units of measure are modeled as scalars.

The ontology *OpenMath units and dimension CD groups* is a part of the standard OpenMath [OpenMath, (2003)] aimed at representation of units of measure and dimensions of mathematical objects. It contains the vocabularies that include physical dimensions' definitions, units of measure for commonly used systems, symbols for manipulating labeled units, prefixes for units of measure of the SI system, units and dimensions for Small Type System, and time units.

The paper [Rijgersberg, *et al.* (2011)] also emphasizes is the importance of measurement units representation for quantitative data interchange in engineering. The authors propose the ontology called OM (Ontology of units of Measure and related concepts). OM defines the complete set of concepts in the domain as distinguished in the textual standards. In order to make OM available for arbitrary software systems, several web services that offer a standardized interface are developed.

SWEET (The Semantic Web for Earth and Environmental Technologies) is a NASA effort to formalize the NASA's Earth Science data in accordance with semantic web standards. They use *GCMD* [GCMD, (1995)] controlled vocabulary as a basis for ontologies development. The current version *SWEET* 2.2 is a highly modular containing over 6000 concepts arranged in 200 ontologies which are grouped in eight top ontologies. Ontologies are implemented using OWL language. Units of measure matters are the subject of the ontology *UNITS*, where units are defined using Unidata's UDUnits and the ontology includes conversion factors between various units.

The Ontology *QUDT* (Quantities, Units, Dimensions and Data Types) [QUDT NASA, (2010)] is the result of the project NASA Exploration Initiatives Ontology Models (NExIOM) achieved as a joint effort of TopQuadrant and NASA. The goal of this ontology is [QUDT NASA, (2010)]: "to provide a unified model of, measurable quantities, units for measuring different kinds of quantities, the numerical values of quantities in different units of measure and the data structures and data types used to store and manipulate these objects in software; to populate the model with the instance data (quantities, units, quantity values, etc.) required to meet the life-cycle needs of the Constellation Program engineering community.". Since this ontology is adopted in this paper, next paragraph describes it in some more details.

The basic concepts of the QUDT ontology are the classes: *Quantity* (an observable property of an object, event or system that can be measured and quantified numerically), *Quantity Kind* (any observable property that can be measured and quantified numerically) *Unit* (a particular quantity of a given kind that has been chosen as a scale for measuring

other quantities of the same kind), *Quantity Value* (the numerical value of a quantity with respect to a chosen unit of measure), *Systems of Quantities and Units* (means to organize quantity kinds and units into a systems), *Base and Derived Quantity Kinds* (no base quantity kind can be expressed as an algebraic relation of one or more other base quantity kinds using only the constituent equations included in the system; derived quantity kind can be expressed as an algebraic relation of one or more base quantity kind;), *Quantity Dimensions* (aimed at supporting certain mathematical properties that permit quantity kinds to be manipulated symbolically) and *Dimension Vectors* (used in conjunction with *Quantity Dimensions*).

3. The Glosent Arhitecture

The presented approaches at best deal with some aspects of heterogeneity and, as a rule, do not cover semantic aspects of the hardware heterogeneity.

In this paper we propose the GLOSENT (GLOSENT - GLObal SENsor neTwork) architecture where we are trying to cope with heterogeneity issues in SWSNs at conceptual level, with emphasis on hardware heterogeneity resolution by using semantic technologies.

The illustration of proposed system architecture in which the client applications communicate with many diverse WSNs is shown in Figure 1.

The proposed architecture consists of three segments: *Application*, *Middleware* and *WSN*. The mediation among these segments is achieved by using proxies. The proxies that perform mediation among the segments are overlapping elements of the GLOSENT segments.

There are three kinds of proxies in the proposed architecture: *WSN proxy*, *Application proxy* and *Service proxy*.

The role of the WSN proxy is to enable communication between WSNs and other segments.

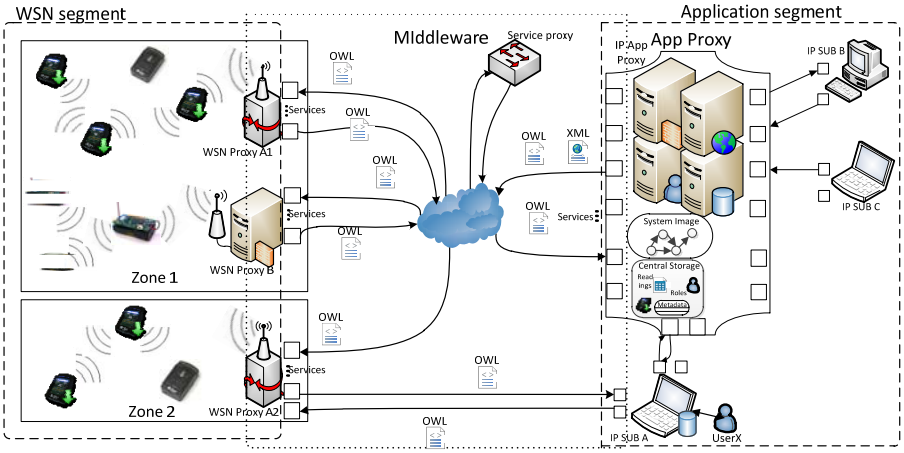

Fig. 1. The GLOSENT architecture

Due to the WSN hardware heterogeneity, the data delivered to WSN proxies are converted (by WSN proxies) to ontological representation.

Application proxy provides communication between user applications and the other segments of the architecture. Here the exchanged data are assumed to be represented by ontology.

Service proxy is aimed at finding and invoking appropriate services for applications.

We use an asynchronous communication model in which both application segment and WSN proxies are information consumers and providers.

3.1. *Application segment*

Application segment consists of an *Application proxy* and a number of *Subapplications*.

Subapplication is any end user application with an arbitrary implementation technology.

Application proxy mediates in the communication of the user Subapplications and other system entities (Figure 1). Subapplication In order to be integrated into the system, Subapplication must be able to use appropriate communication methods and data formats. Necessary information about communication methods and data formats are provided by Application proxy.

The main role of the Application proxy is to provide authorization for Subapplications. In addition, the Application proxy can provide specific locally residing applications that will be available to users (like those aimed at data acquisition and adaptation). Subapplications and WSN proxies can only access Application proxy by using its IP address.

GLOSENT is a distributed system which enables application segment to access multiple WSNs. For that purpose, a *Central storage* and the mechanism for keeping consistent WSNs states representation which is called *System image* are used.

System image contains the metadata describing all WSNs in SWNS as well as the corresponding metadata values. It is stored at Application proxy. In order to obtain desired part of the system image from the Application proxy, Subapplication must be authorized by the Application proxy.

Managing WSNs is accomplished via Application proxy and Subapplications through WSN proxy services. When some change in WSN configuration takes place, the WSN proxy requests Application proxy to perform synchronization (update) procedure. All changes are recorded in the System image.

Although the authorization within the GLOSENT is performed by the Application proxy, for some specific Subapplications it can be done by a WSN proxy as well (for example, some Subapplication not intending to access database, but only some SN). For the sake of higher efficiency, the list of temporary permissions is maintained.

3.2. *WSN segment*

In the GLOSENT architecture, WSN segment is modeled by using the metadata that describe its structure (generalized SNs models, WSN topology, role that specific generalized SN plays in communication), the method of use and control (i.e. middleware or direct access to WSN, constraints imposed to sensor devices, etc.), and the data delivered by WSN (sensor data formats, SN hardware status, etc.).

In our model, SN is a generalized notion related to any WSN device (base stations, rich uncles, routers, etc.) thus enabling a uniform WSN representation. Such a representation is implemented at WSN proxies by a corresponding ontology.

The WSN segment consists of multiple WSNs that can be implemented in different ways, but with common task to mediate between the ontological representation of SN and the concrete SNs implementations. The concrete WSN implementation is described by using SN's metadata, identifiers and revisions. Since the WSN proxy maps each SN to its ontological representation, the communication procedure that SN uses to introduce itself, must be known to WSN proxy. Also, WSN proxies should implement appropriate communication channels which, in the case of GLOSENT are Internet.

The proposed model of the WSN proxies enables simulation of the operations that are not directly available in the WSN implementations (for example, *event based* access to SNs).

Due to its complex functionality (ontology creation, communication, and additional functionalities), the WSN proxy's hardware requirements in our architecture are significant.

3.3. *Middleware segment*

The Middleware segment mediates the communication of WSN segment and application segment in a SWSN.

The ontological representation of the middleware data facilitates the integration of SWSN segments. SWSN segments use appropriate proxies (WSN proxies, Application proxy, and Service proxy) to access middleware (see Figure 1).

The pivotal role of the middleware can be explained through an example of the data flow within the SWSN. SNs are the sources of data. The data are routed through a network of SNs to a WSN proxy. Since the WSN proxy is WSN sync but also a data source for the middleware, the appropriate data conversion takes place at this point. Here, sensed data are converted to ontology and the WSN proxy delivers ontologically represented data to the middleware.

Additionally, Middleware provides up-to-date ontologically represented data to the Application proxy. At Application Proxy the received data are integrated into the System image and the Central storage is updated.

4. WSN model

WSN model consists of two parts: System image and Central storage.

4.1. *System image*

System image is the representation of the current state of all WSNs. It consists of metadata models describing different WSN hardware platforms, metadata values describing particular devices and their relations, and sensor data (raw and/or aggregated sensor readings).

In our model, the System image was represented by using ontologies, where generalized sensor nodes were interpreted as sets of components similar to the model proposed by [Sharman, *et al.* (2007)].

Components were modeled by class *Component* as attribute-value pairs, where the values are taken from the predefined sets of feasible values. In order to describe measured values that have the units of measurement, *MeasureUnit* class was introduced. A component is linked to its measure unit by *hasMeasureUnit* property. Class *Revision* and a property *hasRevision* are aimed at representing moment in which the values have been measured According to the proposed model each concrete component (i.e. instance of the class *Component*) contains an attribute name, a value of the attribute and a timestamp for that value. It can also include the unit of measurement when needed.

Generalized sensor nodes were modeled by *SensorNode* class linked to its components by *hasComponent* property. Each *SensorNode* class that represents a particular WSN hardware platform has a feasible combination of *Component* classes. These constraints can be modeled by using class restrictions and implementing SWRL rules [SWRL, (2004)]. Each concrete, physical sensor node, including its current state is completely represented by an instance of the *SensorNode* class which is a collection of *Component* instances.

The relationships among generalized sensor nodes are represented by the class *Connection*. Due to communication roles of its nodes, WSN is considered an oriented graph and two properties (*hasSourceNode* and *hasDestinationNode*) which link *Connection* class to *SensorNode* class were introduced. Since it is expected that a connection between two generalized sensor nodes can have its own attributes, classes *Connection* and *Component* are linked by using *hasComponent* property. According to the proposed ontology, system image is seen as a collection of *Connection* instances connecting *SensorNode* instances.

In order to represent various WSN topologies like hierarchical organization of WSN nodes and their roles (cluster heads, aggregators, routers ...), it is assumed that a link between two SNs can be characterized by the set of components with appropriate constraints. For that purpose a new restrictions and SWRL rules should be introduced.

For the implementation of the ontology, we have used OWL DL [OWL, (2004)] and Protégé [Protégé, (2011)].

In order to represent units of measurement in a uniform way (which is crucial for semantic integration of the measured data, i.e. abstraction of measured quantities presentation from devices' hardware implementation), this ontology is supplemented with *QUDT* ontology. In this paper we have opted for the *QUDT* ontology for the following reasons:

- It is sufficiently comprehensive for the needs of our system in terms of units of measurement and measured quantities,
- Its OWL implementation facilitates an easy integration in our system.

The integration with *QUDT* ontology is done by establishing equivalence between the class *MeasureUnit* and the class Unit from QUDT ontology (expression owl: *equivalentClass* for classes *MeasureUnit* and *qudt:Unit*). By establishing this equivalence a uniform and unique representation of the units of measurement for WSN components is achieved.

The subclasses taxonomy of the class *qudt:Unit* (i.e. the classes *qudt:DerivedUnit*, *qudt:ResourceUnit*, *qudt:BaseUnit*, *qudt:ScienceAndEngineeringUnit* and *qudt:DimensionlessUnit*) provide for simple adding of new units of measure to a corresponding category of units of measure. Figure 2 shows the proposed ontology.

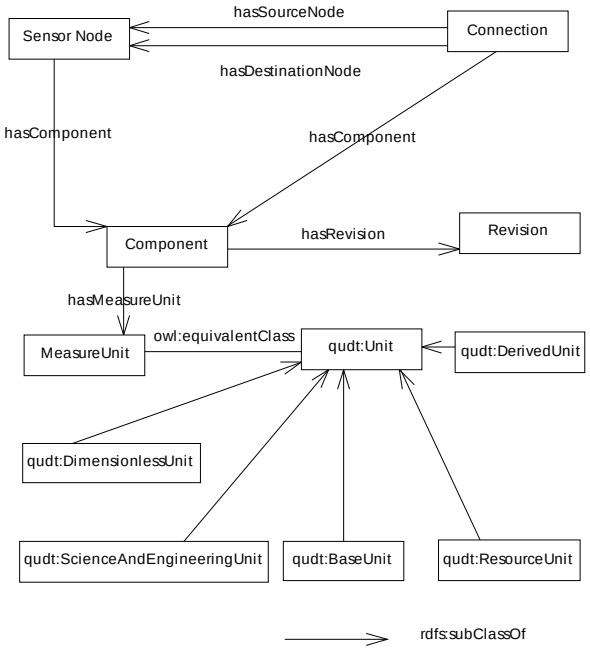

Fig. 2. WSN Ontology.

4.2. *Central storage*

Central Storage implements the data persistency. Its stores the System image as well as the information about Subapplications (users) and their access rights. In our model the Central storage is implemented as relational data model using *PostgreSQL* DBMS [PostgreSQL, (2009)].

Heterogeneous hardware platforms and particularly the need for representing the sensor data that vary depending on applications' demands make the use of "conventional" relational data model inappropriate due to several reasons (different SN platforms representations would require multiple tables, new access mechanisms and fields' interpretations, sparse data problem appears if all SN characteristics are persistently stored).

In our approach the problem is resolved by detaching the SN metadata (i.e. attributes) from their corresponding values. For that purpose, the separate tables containing SN metadata and the uniform mechanism for linking metadata to the corresponding values are used. The sparse data problem is solved by using dynamic tables (one table for each realization of the SN), that links specific metadata descriptions of the hardware platform and instance of the SN in the database as depicted by Figure 3.

The history (versioning) mechanism uses the REVISION field in dynamic tables to represent WSN components changes. This mechanism serves the situations when the WSN proxy is unavailable in order to prevent provision of an invalid (outdated) System image to a Subapplication.

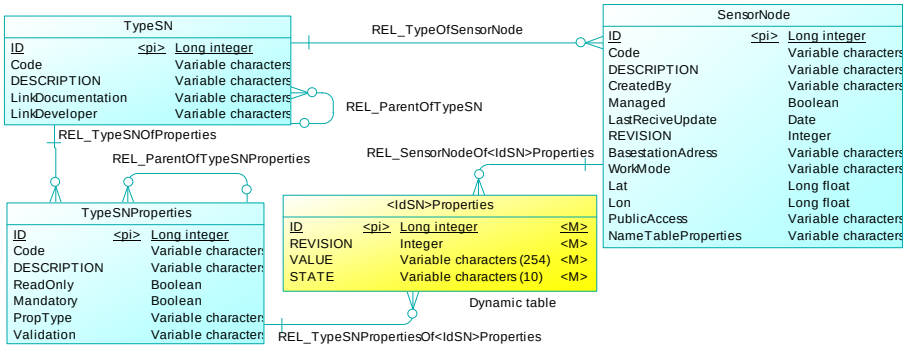

Fig. 3. A part of the database scheme for connecting metadata and data.

Two typical situations that use revision mechanism appear. The first situation is when a client changes some parameters of generalized SNs and the Application proxy sends changes to the WSN proxy that is currently unavailable. The second one is when a Subapplication, directly accessing WSN proxy, is trying to change the state of the generalized SN without knowing the actual state of the WSN.

In the first situation, as soon as the WSN proxy establishes communication with the Application proxy, the WSN proxy compares the state of the WSN against its status in the database. If the revisions do not match, the WSN proxy delivers to a client only the parameters that were changed in the meantime.

In the second situation, the inequality between the revision number appearing in subapplication request and the revision number stored at WSN proxy for the particular SN is detected by the WSN proxy. Then, the WSN Proxy rejects the request and sends the response containing version mismatch to Subapplication. Afterwards, Subapplication invokes the Application proxy service to obtain the actual System image.

5. The SunSPOT Model

As a proof-of-concept for hardware heterogeneity managing the model of the WSN hardware platform SunSPOT is created.

SunSPOT is a WSN sensor node developed by Sun Microsystems. The device was developed based on IEEE 802.15.4 standard. Originally, SunSPOT is used to measure light, temperature and acceleration, but its functionality can be extended with additional sensors.

5.1. *SunSPOT system image*

In order to represent the concrete characteristics of SunSPOT, the proposed WSN ontology (Figure 2) was extended with classification of the SunSPOT's components (Figure 4). The classification covers some basic functionalities of SunSPOT device [SunSpot, (2006)]. The focus of the classification is on hardware characteristics and sensing functionalities.

Classes of processor components (*ProcessingComponent*), sensor components (*SensorComponent*) and components for identifying a sensor node (*SensorNodeID*) were modeled.

Within the class of processing components, in addition to the classes that describe the processor itself (*Architecture*, *ProcessingFrequency*, *CoreType*, *AccelerometerType*, *TimerChipType*), the classes that describe battery (*BatteryComponent*), radio (*RadioComponent*) and memory (*MemoryComponet*) were modeled. The classes *BatteryTechnology*, *ElectricCharge*, *ElectricCurrent* and *Voltage* are subclasses of the class *BatteryComponent*.

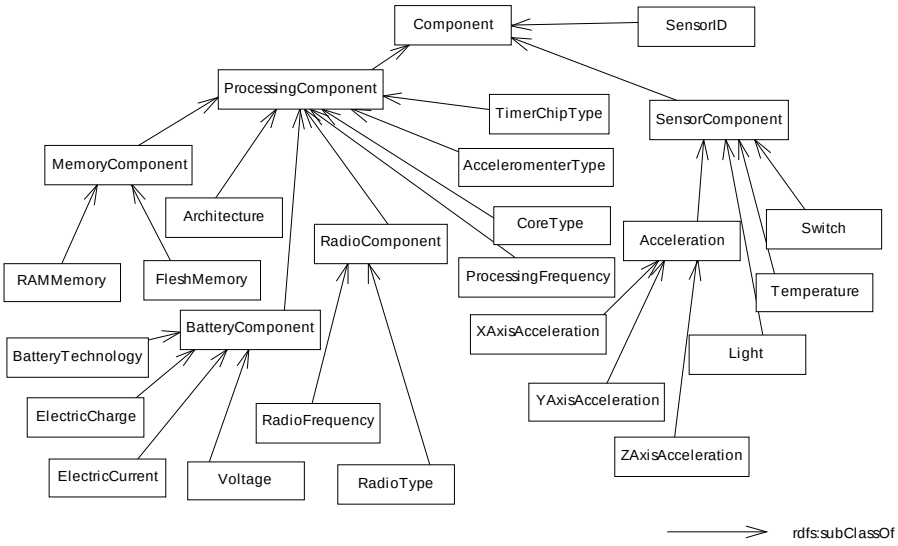

Fig. 4. Classification of components.

The classes *RAMMemory* and *FlashMemory* are subclasses of the class that models memory (*MemoryComponent*). The Classes *RadioFrequency* and *RadioType* are subclasses of the class *Radio*. The classes *Light*, *Switch*, *Temperature* and *Acceleration* are subclasses of the class *SensorComponent*.

The classification of components is extensible, meaning that the new SN hardware implementation could at most require an extension of the classification.

5.2. *SunSPOT central storage*

The Central storage for the SunSPOT is presented in this section. The Central storage is obtained by the transformation of the System image into relational database. The transformation is done by mapping ontologies to relational database.

Database table *TypeSN* describes general properties of each hardware platform (Table 1). The CODE field is the identification of the hardware platform ontology.

DESCRIPTION field is a detailed description of SN type. *LinkDocumentation* and *LinkDeveloper* are user defined fields containing the URLs of official documentation and developer site respectively. The new SN type record can be derived from the existing one (Figure 3).

Table 1. *TypeSN*

TypeSN	
Field Name	**Value**
CODE	SunSPOT
DESCRIPTION	Sun SPOT (Sun Small Programmable Object Technology) is a wireless sensor network (WSN) mote developed by Sun Microsystems...
LinkDocumentation	http://www.sunspotworld.com/docs/index.html
LinkDeveloper	http://www.sunspotworld.com

TypeSNProperties table contains metadata that describe the specific property of SN.

The property type is COMPLEX (Table 2) if the property is the composite one.

Simple property (Table 3) is a property which holds a value of a basic data type (e.g. Integer).

The *TypeSNProperties* tables are obtained by transforming the *Component* classes from System image. *CODE* field defines the short code that is the name of the property (Component's name). Fields *PropType*, *Validation*, *ReadOnly* and *Mandatory* are used for storing metadata of a specific property. *ReadOnly* field is used to specify that the property value can not be changed. *Mandatory* field is used to denote if this property is mandatory in the SN configuration. *Validation* field contains property constraints (e.g. ranges, domains, rules).

One example of the complex property is the *Radio* property (Table 2). It consists of *Channel*, *Pan ID* and *Transmit Power* subproperties.

Table 2. Complex property Radio

TypeSNProperties	
Field Name	**Value**
CODE	Radio
TypeSN_ID	SunSPOT
PARENT_ID	-
DESCRIPTION	Set of properties for maintaining radio connection
PropType	COMPLEX
Mandatory	TRUE
ReadOnly	FALSE
Validation	-

Table 3. Simple property Transmit Power

TypeSNProperties	
Field Name	**Value**
CODE	TransmitPower
TypeSN_ID	SunSPOT
PARENT_ID	Radio
DESCRIPTION	Property for maintaining transmit power
PropType	INTEGER
Mandatory	TRUE
ReadOnly	FALSE
Validation	MIN=-32,MAX=31

The table *TypeSensor* contains metadata that describe a sensor type. Table 4 presents SunSPOT sensor type.

Table 4. Temperature sensor type

TypeSensor	
Field Name	**Value**
CODE	ADT7411
DESCRIPTION	Temperature built-in sensor
SensorKind	Ambient
MesurementUnitType_ID	http://www.qudt.org/qudt/owl/1.0.0/unit/Instances.html#Kelvin
TypeMeasurement_ID	http://www.qudt.org/qudt/owl/1.0.0/quantity/Instances.html #ThermodynamicTemperature
MINReadings	228
MAXReadings	398
Granularity	0.5
WorkingConditions	MIN:263,MAX: 333

Here the field *SensorKind* defines the type of environment in which the sensor operates. *MesurementUnitType_ID* field links sensor type to its measure unit. Value of field *MesurementUnitType_ID* is the URI that points to the instance of *QUDT* ontology *Unit* class *Kelvin*. The sensed phenomena are represented by the field *TypeMeasurement_ID*. Value of field *TypeMeasurement_ID* is the URI that points to the instance of *QUDT* ontology Quantity class *ThermodynamicTemperature*. The constraints of sensor readings are modeled by fields *MINReadings* and *MAXReadings* respectively, while the level of granularity is represented by the field Granularity, and the working conditions of sensors by the field *WorkingConditions*.

SensorNode table contains values that describe the states of concrete SNs (Table 5). The field values are combinations of data obtained from WSN proxies and/or those created by users. The filed *LastReciveUpdate* contains the time when the latest communication of SN with a WSN proxy occurred. The fields Lat and Lon are geographical latitude/longitude of SN position. The field *TypeSN_ID* points to the corresponding SN type, while the filed *CODE* is the SN ID (address). *WorkMode* field defines the role that the SN plays (e.g. emitter, router, basestation, etc.). The name of the corresponding dynamic table is stored in the *NameTableProperties* field.

Table 5. Sensor node

SensorNode	
Field Name	**Value**
CODE	0000.591F
TypeSN_ID	SunSpot
CreatedBy	BASESTATION
LastReciveUpdate	10/28/2010 5:53 PM
BasestationAdress	0000.6D29
REVISION	1
WorkMode	EMITER
Lat	45.251667
Lon	19.836944
NameTableProperties	0000.591FProperties

The table Sensor contains data representing the properties of the particular sensor (Table 6). It contains the revision of the last change (*REVISION*), sampling rate (*MeasuresEvery*), alarm configuration (*TresholdUP*, *TresholdDOWN*, *Alarm*), as well as *SensorNode_ID* and *TypeSensor_ID*. *SensorNode_ID* identifies SN, while *TypeSensor_ID* points to the corresponding *SensorType*.

Table 6. Sensor

Sensors	
Field Name	**Value**
SensorNode_ID	0000.591F
TypeSensor_ID	ADT7411
CODE	0000.591F_ADT7411_1
REVISION	1
MeasuresEvery	3
TresholdUP	100
TresholdDOWN	-
Alarm	FALSE

Naming convention for dynamic tables is as follows. Entity id is used as a prefix to table name. The variable part of the table name is enclosed in <>. *<IdSN>Properties* contains property values of SN. The metadata stored in the table *TypeSNProperties* describe the usage of the data from dynamic tables. For each data row of the table *SensorNode* a distinct table containing SN's attributes is created. The table names are created as combinations of SN identifier and predefined suffix "Properties", as in Table 7 (e.g. 0000.591Fproperties). The table fields are the revision number of the specified property (*REVISION*), its current value (*VALUE*) and its current state (*STATE*). *STATE* field denotes weather the property is added, changed or removed (*ADDED*, *CHANGED*, *REMOVED*). The field *SensorNode_ID* points to the SN table, while the field *TypeSNProperties_ID* points to the property of SN type (Figure 3).

Table 7. Transmit Power property of SN entity

0000.591FProperties	
Field Name	**Value**
SensorNode_ID	0000.591F
TypeSNProperties_ID	TransmitPower
REVISION	1
VALUE	0
STATE	ADDED

Tables *<IdSNS>ReadingsSensors* contain readings of a particular sensor (an example for temperature readings is shown in Table 8). One table is created for each record of the table Sensors; it contains timestamp (*DateTime*), measured value (*VALUE*) and the identifier (*Sensor_ID*).

Table 8. Readings of temperature sensor

0000.591F_ ADT7411_1ReadingsSensors	
Field Name	**Value**
Sensor_ID	0000.591F_ ADT7411_1
DateTime	10/28/2010 5:53 PM
VALUE	45

6. CONCLUSION

This paper is concerned with the problem of hardware heterogeneity in large systems of WSNs. We have proposed an ontologically-based approach for modeling SWSNs. The topology of the WSN is represented by high-level ontology, while the semantics of WSN components is represented by appropriate classifications. The GLOSENT architecture that facilitates integration of heterogeneous SWSN segments is also proposed. The GLOSENT architecture relies on SOA and ontological representation of WSNs and data. The segments of the GLOSENT architecture aimed at resolving heterogeneity problems are described. In addition, as a case study the concrete SN device SunSPOT is represented using proposed ontology.

The WSN architectures overviewed in the section 2 enable semantic integration of either sensed data or the data about the WSN components. In these architectures the semantic integration is achieved for the specific application domains.

The GLOSENT architecture simultaneously solves the problems of semantic integration of sensed data and WSN components data in WSN systems with high degree of hardware heterogeneity. The application of GLOSENT architecture is not restricted to specific application domains, and it can be used in arbitrary contexts. For that reason, the sensed data as well as WSN components data is ontologically represented by using OWL, which enables the semantic representation of data regardless of the concrete application domain.

Since the aim of GLOSENT is to facilitate dealing with heterogeneity in SWSN in general, further research will be directed towards development of ontologies dealing with other aspects of heterogeneity in SWSN and deployment of these ontologies in GLOSENT. In order to provide advanced functionalities of the SWNS (adaptive SWNS) development of ontolgies for context-sensitive usage should also be the subject of further research.

7. ACKNOWLEDGMENTS

The results presented in this paper are part of the research conducted within the Grant No. III-47003 funded by the Ministry of Science and Technological Development of the Republic of Serbia.

References

Avancha, S., Patel, C., Joshi, A. (2004). Ontology-driven adaptive sensor networks. *Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004. The First Annual International Conference on* (August 22-26, 2004). pp. 194 - 202. DOI= http://dx.doi.org/10.1109/MOBIQ.2004.1331726.

Brőring, A., Jűrrens, E. H., Jirka, S., and Stasch, C. (2009). Development of Sensor Web Applications with Open Source Software, OpenSoruce GIS 2009, University of Nottingham (Suchith Anand, 2009).

Barnaghi, P., Meissner, S., Presser, M., and Moessner, K. (2009). Sense and Sens'ability: Semantic Data Modelling for Sensor Networks. *ICT-Mobile Summit 2009 Conference Proceedings,* 2009.

GCMD (1995), Global Change Master Directory. available at: http://www.openmath.org/. Visited Junu 2011.

Gomez, L., and Laube, A. (2009). Ontological Middleware for Dynamic Wireless Sensor Data Processing. In *Proceedings of the 2009 Third International Conference on Sensor Technologies and Applications* (SENSORCOMM '09). IEEE Computer Society, Washington, DC, USA, pp. 145-151. DOI=http://dx.doi.org/10.1109/SENSORCOMM.2009.121.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*. (June 1993), vol. 5, no. 2, pp. 199-220. DOI= http://dx.doi.org/10.1006/knac.1993.1008.

Hadim, S., and Mohamed, N. (2006). Middleware: Middleware Challenges and Approaches for Wireless Sensor Networks. *Distributed Systems Online, IEEE.* (March 2006), vol. 7, no. 3, pp. 1-23. DOI=http://dx.doi.org/10.1109/MDSO.2006.19.

Jang, W., Healy, W.M., and Skibniewski, M.J. (2008). Wireless sensor networks as part of a web-based building environmental monitoring system. *Automation in Construction*. (August 2008), vol. 17, no. 6, pp. 729-736. DOI= http://dx.doi.org/10.1016/j.autcon.2008.02.001.

Kim, J.H., Kwon, H., Kim, D.H., Kwak, H.Y., and Lee, S.J. (2008). Building a Service-Oriented Ontology for Wireless Sensor Networks. In *Proceedings of the Seventh IEEE/ACIS International Conference on Computer and Information Science (icis 2008)* (ICIS '08). IEEE Computer Society, Washington, DC, USA, pp. 649-654. DOI= http://dx.doi.org/10.1109/ICIS.2008.100.

Levis, P., and Culler, D. (2002). Maté: A tiny virtual machine for sensor networks. *SIGOPS Oper. Syst. Rev.* (October 2002), vol. 36, no. 5, pp. 85-95. DOI=http://doi.acm.org/10.1145/635508.605407.

Liu, T., and Martonosi, M. (2003). Impala: a middleware system for managing autonomic, parallel sensor systems. *SIGPLAN Not.* (June 2003), vol. 38, no. 10, pp. 107-118. DOI=http://doi.acm.org/10.1145/966049.781516.

Liu, J., Zhao, F. (2005). Towards semantic services for sensor-rich information systems. *Broadband Networks, 2005. BroadNets 2005. 2nd International Conference on* (Boston, MA, USA, October 7-7, 2005). pp. 967 - 974. DOI= http://dx.doi.org/10.1109/ICBN.2005.1589709.

Madden, S. R., Franklin, M J., Hellerstein, J. M., and Hong, W. (2005). TinyDB: an acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.* (March 2005), vol. 30, no. 1, pp. 122-173. DOI=http://doi.acm.org/10.1145/1061318.1061322.

Masri, W., and Mammeri, Z. (2007). Middleware for Wireless Sensor Networks: A Comparative Analysis. *Network and Parallel Computing Workshops, 2007. NPC Workshops. IFIP International Conference on* (Liaoning, September 18-21, 2007). pp. 349 - 356. DOI= http://dx.doi.org/10.1109/NPC.2007.165.

Molla, M. M., and Ahamed, S. I. (2006). A Survey of Middleware for Sensor Network and Challenges. In *Proceedings of the 2006 International Conference Workshops on Parallel Processing* (Washington, DC, USA, 2006). IEEE Computer Society, pp. 223-228. DOI=http://dx.doi.org/10.1109/ICPPW.2006.18.

Murphy, A. L., and Heinzelman, W. B. (2002). *Milan: Middleware Linking Applications and Networks*. Technical Report. University of Rochester, Rochester, NY, USA.

NASA SWEET (2011), available at: http://sweet.jpl.nasa.gov/ontology/. Visited August 2011.

OpenMath (2003), available at: http://www.openmath.org/. Visited Junu 2011.

OWL (2004), Web Ontology Language, available at: http://www.w3.org/2004/OWL/#specs. Visited November 2010

PostgreSQL (2009). WWW, available at: http://www.postgresql.org/. Visited November 2010

Protégé (2011). WWW, available at: http://protege.stanford.edu/. Visited July 2011.

QUDT NASA (2010), Quantities, Units, Dimensions and Data Types in OWL and XML, available at: http://qudt.org/. visited Junu 2011

Ramos, M. L., Leguay, J., and Conan, V. (2007). Designing a Novel SOA Architecture for Security and Surveillance WSNs with COTS. *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE International Conference on* (Pisa, October 8-11, 2007). pp. 1 - 6. DOI= http://dx.doi.org/10.1109/MOBHOC.2007.4428703.

Rijgersberg, H., Wigham, M., and Top, J.L. (2011). How semantics can improve engineering processes: A case of units of measure and quantities, *Advanced Engineering Informatics* (April 2011), vol. 25, no. 2, pp 276-287, ISSN 1474-0346, DOI=http://dx.doi.org/10.1016/j.aei.2010.07.008.

Simon, D., Cifuentes, C., Cleal, D., Daniels, J., and White, D. (2006). Java™ on the bare metal of wireless sensor devices: the Squawk Java virtual machine. In *Proceedings of the 2nd international conference on Virtual execution environments* (VEE '06). ACM, New York, NY, USA, pp. 78-88. DOI=http://doi.acm.org/10.1145/1134760.1134773.

SensorML (2010), Sensor Modeling Language, available at: http://www.opengeospatial.org/standards/sensorml. Visited August 2011

Sharman, R., Kishore, R., and Ramesh, R. (2007). Using Ontologies in the Semantic Web: A Survey. *Ontologies: A Handbook of Principles, Concepts and Applications in Information Systems (Integrated Series in Information Systems)*. Springer-Verlag, New York, 2007, pp. 79-113, DOI= http://dx.doi.org/10.1007/978-0-387-37022-4_4.

Souto, E., Guimarães, G., Vasconcelos, G., Vieira, M., Rosa, N., Ferraz, C., and Kelner, J. (2005). Mires: a publish/subscribe middleware for sensor networks. *Personal Ubiquitous Comput.* 10, 1 (December 2005), 37-44. DOI=http://dx.doi.org/10.1007/s00779-005-0038-3.

SunSpot (2006). WWW, available at: http://www.sunspotworld.com. visited November 2010

SWRL (2004). Semantic Web Rule Language, available at: http://www.w3.org/Submission/SWRL/. Visited November 2010

Terfloth, K., Güneş, M., and Schiller, J. H. (2009). Middleware for Wireless Sensor Networks: The Comfortable Way of Application Development. *Guide to Wireless Sensor Networks*, Springer, London, 2009, 583-606. DOI=http://dx.doi.org/10.1007/978-1-84882-218-4_23.

The Basic SunSPOT Ontology (2010), available at: http://informatika.ftn.uns.ac.rs/SinisaNikolic?action=AttachFile&do=get&target=SunspotBasicModel.owl. visited March 2011

The Open Geospatial Consortium (1994). available at: http://www.opengeospatial.org/. Visited November 2010.

Thomas, R. G., and Gregory, R. O. (1994). An ontology for engineering mathematics. *In Fourth International Conference on Principles of Knowledge Representation and Reasoning*, 1994. pp. 258-269.

TinyOS Project (2000). WWW, available at: http://www.tinyos.net/. Visited November 2010

Zafeiropoulos, A., Spano, D. E., Arkoulis, S.,Konstantinou, N., and Mitrou, N. (2011). Data management in sensor networks using Semantc web tehnologies. *Data Management in the Semantic Web. 2009 Nova Science Publishers, Incorporated,* 2011, ISBN: 9781611228625