A DSM for a Modeling RESTful Sensor Web Network

By Vladimir Vujović [*]
Mirjana Maksimović[†]
Branko Perišić[‡]

Entering the era of Internet of Things, the use of Wireless Sensor Network and Sensor Web elements becomes a daily phenomenon. Information systems which must ensure that an increasing number of devices access and analyze the data collected from the wireless sensor nodes, have become complex and difficult to maintain. Due to the discrete nature of computer based systems, where the relationship between action and response may not be proportional, there is a huge risk when control is concerned. Design of unique sensor network that will meet the imposed needs sometimes is a serious task usually involving experts from different areas of the problem, design and implementation domains. On the other hand the introduction of novel technologies and techniques, like Domain-Specific Modeling and RESTful service technology, in sensor networks design, raises the importance of problem domain. The involvement of such a wide variety of stakeholders, together with the nature of computer based systems and the constantly changing methods and technologies, sometimes leads to overall project failure mainly because of the lack of a mutually agreed high level domain specific model. In this paper, the specification of meta-model for RESTful Sensor Web Network modeling together with accompanying language editor, are described. The verification and validation of its functionality is performed based on suitable examples of sensor network design models.

Introduction

Wireless sensor network (WSN) is composed of spatially distributed autonomous sensor nodes equipped with sensing devices (to monitor environmental conditions at different location), processing unit, communication components (wireless transmitter/receiver), small memory (storage unit), and a source of energy (power unit). Despite the several major WSN constraints like: inadequate source of energy, restricted computational

[*]Teaching Assistant, University of East Sarajevo, Bosnia and Herzegovina.
[†]Teaching Assistant, , University of East Sarajevo, Bosnia and Herzegovina.
[‡]Associate Professor, University of Novi Sad, Serbia.

potentiality and limited memory, they represent a rapidly growing research and commercial development domain. The range of WSN applications includes, but is not limited to, military applications, environmental monitoring systems, intelligent agriculture, surveillance, health monitoring, traffic monitoring, industrial control and monitoring (Ali, 2010).

Recent development in WSN technology and the use of Internet Protocol (IP) in resource constrained devices has radically changed the Internet landscape creating a new concept called Internet of Things (IoT). One of the most important building elements of IoT is a sensor node, precisely, a Sensor Web as its configuration. Traditionally, Sensor Web is defined as a Web of interconnected heterogeneous sensors that are interoperable, intelligent, dynamic, scalable and flexible, but can often be seen as a group of interoperable web services compliant with a specific set of behaviors and sensor interfaces specifications (Di, 2007). The differences between ordinary sensors and Sensor Web should be transparent to the end user (Delin, 2005). There are two main ways to connect sensor nodes to the Internet. One way is to employ gateways, which work as converters between protocols of the Internet and custom protocols used in the WSN. The other way is an end-to-end communication using IP enabled things (Yazar, 2009). This paper addresses both approaches by application of proven and widely-used standards (HTTP, TCP/IP, etc) directly on sensor and gateway nodes hence avoiding protocol conversion. In this case, clients are able to interact with both just as they do with traditional web servers in the Internet. They may not even be aware of the fact that they are accessing a very resource-constrained device.

To support all above mentioned features, the development of new communication protocols, algorithms, designs, and services are needed (Ramadan and El-Rewini, 2009). Programming and software development for WSN is a tedious task mostly performed via programming languages that are close to the machine architecture (like C or Assembler) demanding detailed knowledge of the hardware and its limitations. One way to simplify the development is the usage of specially developed dedicated WSNs programming languages. Domain-Specific Languages (DSL) help to ease the programming by abstracting low-level details and enabling domain-experts to describe the behavior of a WSN (Sadilek, 2008). Thus, the goal is to integrate domain experts into the WSN software development process by providing them with DSLs they may easily understand and apply. Therefore, the DSLs' concepts and concrete syntax must match the domain experts' cognitive space and intuition. Also, the domain experts should be able to test the programs they have written in the DSLs by directly executing them (Sadilek, 2007).

Thus, the general objective of DSM (Domain-Specific Model) solutions and the model presented in this paper is to create a quick and easy to use approach to the development of environmental sensing application which enables people, having little or no experience in WSN programming, to define tasks and specify network architecture of WSN applications.

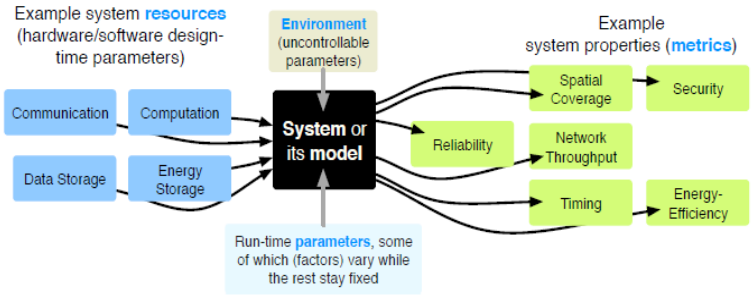
The rest of this paper is structured as follows. Section 2 reviews related work in the area of DSM and WSN. Section 3 states advantages of RESTful

approach for embedding Internet in WSN, while Section 4 discusses concept of RESTful Sensor Web networks modeling. The specification of the proposed structural meta-model using EMF is given in Section 5. Section 6 concludes the paper and explains the roadmap for evaluation and other issues for future work.

WSN-DSM Literature Review

Stanley-Marbell et al. (2008) presents a coherent organization of a large collection of existing models (ranging from models of signal propagation and reflection or absorption by objects, to models of applications and the phenomena they monitor or are driven by) in a taxonomy of WSN models (Figure 1). It provides some insight into the distribution of types of available models, and current directions in the use of models in sensor network research.

Figure 1. *Abstractions of the Functional Behavior of a System*



Sadilek (2008) has developed and evaluated an approach for defining DSLs for WSN, for simulating, compiling, and executing programs. Sadilek (2007) also proposed the usage of meta-programming in combination with meta-modeling technologies, and thus the combination of the Scheme programming language and Eclipse EMF is used. The results of this work have shown that models in stream-oriented DSL, created for the simulation, can be executed on the target platform without modifications and that redundancy in the description of a DSL's operational semantics for simulation and for the target platform can be avoided. Ben Maissa et al. (2012) present VeriSensor, a domain specific modeling language for WSNs with formal verification support. It is designed for WSN experts enabling the modeling of a WSN by providing high-level concepts that support the main use cases of WSNs. The WSNs speciation consists of: the node characteristics definition, the node deployment definition and the physical environment characteristics which may model all possible situations leading to the evaluation of the WSN in the worst possible conditions. Fajar et al. (2011) proposes a new high-level abstraction model based on DSM which enables developers to build WSN applications using logical and physical abstraction model, task allocation as well as automation via code generation. Published results have shown that, by using the proposed model, productivity in development time is increased about six times in

comparison to the manual approach, while the communication cost is significantly reduced by average calculation at intermediate nodes.

Restful Services in WSN and Sensor Web

Sensor Web Enablement (SWE) concept assumes that all sensors are: connected to the Web, reporting their positions, possessing registered meta-data and allow remote read or control. Hence, the stated objectives are to:

- make all kinds of sensors: discoverable, accessible and controllable via the WWW,
- create the framework for a WWW-based sensor web and
- introduce the foundation for "plug-and-play" web-based sensor networks.

The capacity to discover and integrate observations from any sensor according to end user needs, independent of its origin, is the main objective of SWE. In that sense, functionality of a Sensor Web includes:

- Discovery of sensor systems, observations, and observation processes that meet an application or users' needs.
- Determination of a sensor's capabilities and quality of measurements.
- Access to sensor parameters that automatically allow software to process and geolocate observations.
- Retrieval of real-time or time-series of observations and coverages in standard encodings.
- Tasking of sensors to acquire the observations of interest.
- Subscription to and publishing of alerts to be issued by sensors or sensor services based upon certain criteria (Martinez, 2011).

As Sensor Web can be represented as a set of web services it is necessary to find appropriate methods for its implementation.

Choosing service interaction style is a major architectural decision for designers and developers, because of its influences to the underlying Web service solutions implementation constraints. Although SOAP (Simple Object Access Protocol) and REST (Representational State Transfer) are both contemporary implementation practice for Web services building, they differ in the manner they process data and deliver services. SOAP is XML-based message exchanging protocol for distributed computing, whereas REST is a design principle for Web-based applications that closely adheres to client-server architecture and advocates using bare minimum HTTP methods.

Although there are several works discussing the comparison of these two notions, it is obvious that the comparing of these two technologies is not a trivial task (Potti et al., 2012). Certainly, both types have their advantages and disadvantages that make one of them more suitable for different specific cases. There are two main problems with SOAP-based Web services. As first they use the Web as a transport medium without the integration in its structure. Secondly, they are heavier and more complex than RESTful Web services in

terms of memory, bandwidth and computation requirements especially due to SOAP layer and constantly growing WS-*stack (Pautasso et al., 2008).

REST is an architectural model for building the distributed applications based on three distinct concepts: representation, state, and transfer (Rouached et al., 2012):

- Representation: Data or resources are encoded as representations of the data or the resource. These representations are transferred between clients and servers.
- State: All of the necessary state, needed to complete a request, must be provided with the request. The clients and servers are inherently stateless. A client cannot rely on any state stored in the server, while the server cannot rely on any state stored in the client. This does not, however, pertain to the data stored by servers or clients; only to the connection state that is needed to complete transactions.
- Transfer: The representations and the state can be transferred between client and servers.

REST is an architectural model that can be efficiently implemented as a combination of the Hypertext Transfer Protocol (HTTP) and TCP/IP. With this instantiation of REST, HTTP requests are used to transfer representations of resources between clients and servers (Rouached et al., 2012). In a WSN a "resource" is the information provided by the sensors (e.g. temperature, humidity, etc). Each "resource" has assigned, Uniform Resource Identifiers (URIs) with which it is possible to access its representations. In REST, the exchange and manipulation of "resource" representations follows the Client/Server communication style. In this model, communications are constrained to be stateless that is, each message between client and server has to be self-descriptive (Ludovici and Calveras, 2010). The advantages of REST are (Potti et al., 2012): the same portability as SOAP but much simpler, less reliance on tools, every resource has a unique URL (uses the power of HTTP (POST, GET, PUT, and DELETE) to operate on the resource (CREATE, READ, UPDATE, DELETE)) and it is lightweight (no extra XML markup). Thus, REST, pointing to low complexity and easy integration with the Web, is a good candidate to embed Internet in WSN (Ludovici and Calveras, 2010). In other words, RESTful Web services are a better option for making sensors a part of Web, because lighter nature is an extra advantage for limited environments (Yazar, 2009).

Concept of Restful Sensor Web Networks Modeling

Sensor networks can rarely be seen as homogeneous independent entities. They are usually part of complex environments and software systems designed for monitoring and data processing. Concerning Sensor Web elements, the environment is the whole Internet. The problem with sensor networks design is how to quickly and efficiently produce a complete information system that is

able to: monitor, send and process data from a sensor network. A significant factor behind the difficulty of developing complex software is the wide conceptual gap between the problem and the implementation domains of discourse (France and Rumpe, 2007).

France and Rumpe (2007) mention that usage of Model-Driven Engineering (MDE) can reduce gap between problem and software implementation domains through the use of technologies that support systematic transformation of problem-level abstractions to software implementations.

Liddle (2010) and Brambilla et. al (2012) pinpoint the key fact for MDE software development - the system is a model consistent with its meta-model. That model is the link between the *problem domain* and *solution domain*, and metamodel represents an abstract model of the system that describes the most common definition of the model.

In the MDE vision of software development, models are the primary artifacts of development, and developers rely on computer-based technologies to transform models to running systems.

Models can be used for different purposes (Brambilla et. al, 2012):

- descriptive (i.e., for describing the reality of a system or a context),
- prescriptive (i.e., for determining the scope and details at which to study a problem) or
- for defining how a system shall be implemented.

The key benefits of Model-driven approaches are described by Liddle (2010), where the most important are: increased developer productivity, decreased cost (in time and money) of software construction, improved software reusability and the higher level of software maintainability.

Using models and tools for sensor networks design, whether they consist of graphical representations, textual specifications, or both, makes the design process much easier, while the emphasis shifts to the design of networks themselves.

MDE approach relies on modeling languages which are usually divided into two large groups:

- Domain-Specific Languages (DSLs) – Languages usually designed with a specific purpose. By using DSL language, the *problem domain* is considerably simplified and consequently its solution too. They are usually developed for problems which need to be described in a specific domain.
- General-Purpose Modeling Languages (GPLs) – Languages, usually with extensible syntax, that are designed for a wide range of domains and modeling purposes.

Unlike DSL, DSM uses models to describe the individual components of the domain system. Models are often based on a graphical representation and supported by graphical design tools. DSM tool enables the user to create domain models, and usually, based on created models, to generate a certain part of code.

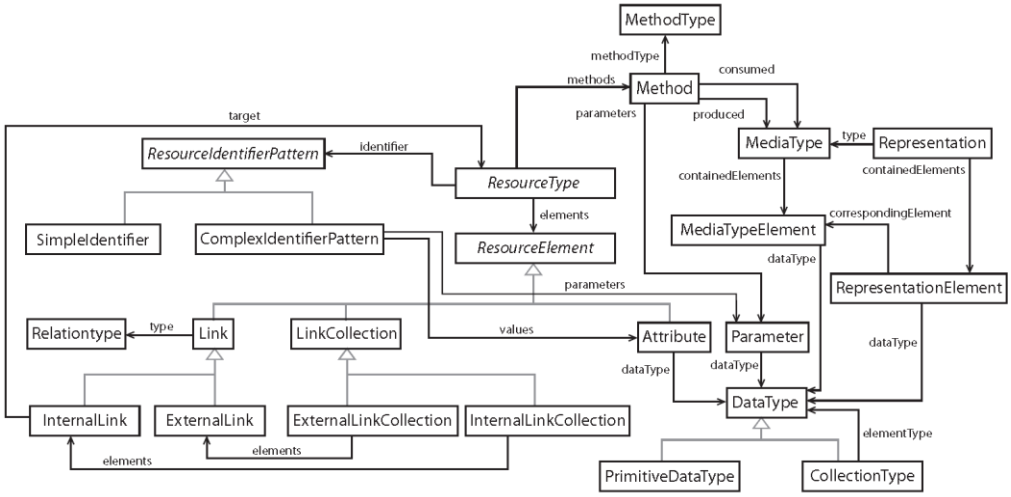
The usage of DSL in designing sensor networks is not a new topic. Depending on the problem domain and the type of sensor networks, DSL can be used to design and simulate the topology (Camilo et al., 2007) or for programming sensing units (Sadilek, 2008; Sadilek, 2007; Stanley-Marbell et al., 2008; Fajar et. al, 2011). In these papers, the majority of authors deal with the sensor nodes' topology and communication problem solving, but no one considers infrastructure solutions specification and processing in the context of an integrated information system. Since each element of the network has a unique IP address through which it communicates with the rest of the network, with Sensor Web elements usage communication and routing problems can be prevented in the early design phases. From this point of view, it is essential to pay more attention to the creation of services that support sensor elements usage.

The importance and use of services in sensor networks is discussed and underlined in Section 3 of this paper. Taking RESTful services as "the backbone" of the system and relying on sensor network elements as resources, it is possible to design the information system that would meet the majority of WSN system demands. Rouached et al. (2012) describe an approach for the implementation of such system.

To fulfill this task, at first it is necessary to closely relate two areas, RESTful modeling and WSN network modeling, and to develop a DSM enabling RESTful Sensor Web network design. Schreier (2011) described the modeling of RESTful applications using the Eclipse Modeling Framework (EMF), relying on Stahl's suggestion that EMF may be the primary choice because of its simplicity and widespread tools and good enough support.

Dividing REST into two models, the structural and behavioral model, Schreier (2011) tried to capture the overall behavior of RESTful service. The core of the structural meta-model of RESTful service is presented in Figure 2.

Figure 2. *The Core of the Structural Metamodel*



Based on works presented by Rouached et al. (2012), Schreier (2011) and Paternostro and Hussey (2009), it is possible to state the key facts that have directly influenced the structure and functionality of our prototype of DSM for the RESTful Sensor Web network. The key factors are:

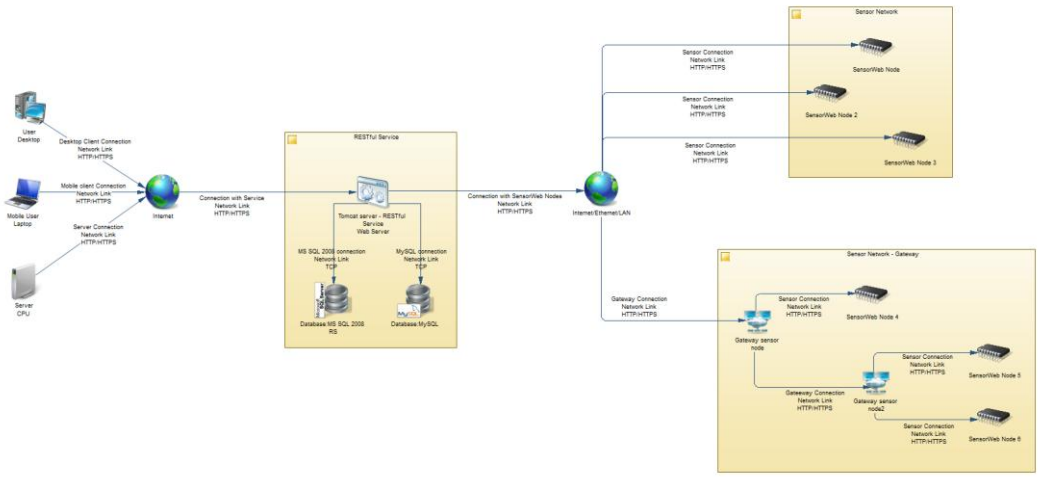
- the usage of EMF ECore for model creation,
- the created model is used as a description of RESTful services and sensor nodes,
- the evolution of prototyping through a set of prototype stages starting with:
    - the support for structural modeling only,
    - the focus on a network design (The reduction of the core structural model presented in Figure 2 is necessary in order to focus on network design as a main topic in the first stage of prototype development).

Modeling Restful Sensor Web Networks with EMF

In the previous sections the key elements relevant for creating a prototype of the DSM RESTful Sensor Web network are defined. In addition to that, it is necessary to discuss the implementation details of DSM and to precisely define the problem domain that specified DSM potentially resolves.

The infrastructure diagram presented in Figure 3 states our proposition of the RESTful Sensor Web Network infrastructure model.

Figure 3. *RESTful Sensor Web Infrastructure Diagram*



There are four basic parts of the RESTful Sensor Web infrastructure:

- ***The Client side*** – Represents the clients that access the RESTful Sensor Web Network. Clients can be independent users (desktop and mobile users), or any other system (software systems) for

collecting and/or processing data of the RESTful Sensor Web Network.

- ***The RESTful Service*** – Represents the implementation of RESTful service that provides all the necessary interfaces for accessing the Sensor Web Network. Besides that, depending on the problem domain, it can provide access to the database in order to collect information from Sensor Web networks and store it for further processing.
- ***The Sensor Web Network*** – Represents the field of independent sensor nodes configured as logical units. These units can be stand-alone sensor units (nodes), or groups of nodes connected via the gateway. In both cases, they are treated as Sensor Web.
- ***A unit*** can be accessed by referencing a single IP address and implemented interfaces.

Looking to the Infrastructure diagram shown in Figure 3 as a composition of services and sensors elements, designers can get a clear insight into the sensor network regarding the knowledge of its implementation, services or sensor elements.

A system's structural meta-model, presented in Figure 4, is created using EMF ECore (Budinsky et al., 2003) based on the *problem domain* described in Figure 3. The Meta-model, presented in Figure 4, relies on Jersey JAX-RS framework and is used for creation of RESTful service in Java.

It is important to note that the presented meta-model system is a prototype without certain elements (like the database on the server side).

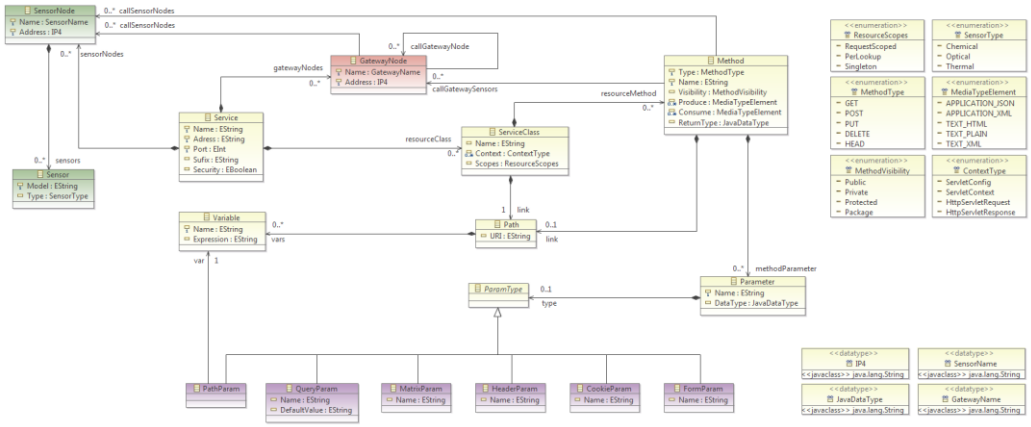
According to the Jersey framework, it was necessary to create *EEnum* objects that correspond to the possible states of objects in the framework: *ResourceScopes*, *MethodType*, *MediaTypeElement*, *MethodVisibility*, *ContextType* and *EEnum SensorType* object that contains the types of sensors (only three types of sensors are shown).

Data types *IP4*, *SensorName*, *GatewayName* and *JavaDataType* were created in order to ensure valid data entry and to enable validation of models created using the editor.

EClass Service is the root object of the model and contains information about the service, its access settings, as well as the services implementation classes, gateway units and sensor nodes.

SensorNode object represents *Sensor Web* node defined by a name and IP address. Each *SensorNode* can collect data from multiple sensor units described with *EClass Sensor*. The internal structure of *SensorNode* object is not taken into consideration. Only the awareness of its existence and the ability to access sensor data via an IP address are important at this stage.

Figure 4. *ECore Metamodel of RESTful Sensor Web Service*



GatewayNode enables service decomposition into smaller units accessible via the Gateway node IP address, and *GatewayNode* objects composition building.

The remaining *EClass* objects: *ServiceClass*, *Method*, *Variable*, *Path*, *Parameter* and all inherited *ParamType* objects ensure the creation of RESTful services based on Jersey Framework and thus create full support for RESTful services.

Service includes all the necessary objects for RESTful Sensor Web Service creation. *GatewayNode* can be aware of the *SensorNode* object, but also of other *GatewayNode* objects located on the service. *Method* objects which represent methods of *ServiceClass* objects can be aware of *SensorNode* and *GatewayNode* objects and thus fulfill the requirement of infrastructure shown in Figure 3.

For the validation purposes of created metamodel shown in Figure 4, a simple Editor is created. The Editor role for an example of RESTful Sensor Web Network creation is presented in the following section.

Restful Sensor Web Network DSM in Action

As a validation example, a simple RESTful Sensor Web network, with two sensor nodes equipped with temperature sensors, are used. The following thermistors are used as temperature sensors - models B57045K10 (NTC thermistors for temperature measurement) and NTC10K (Resistance table for 10kNTC Thermistor). Both sensor nodes have associated IP address and are accessible via RESTful service that provides:

- information on the number of sensors and sensor nodes,
- data from individual sensor nodes and sensing temperature values.

In this example, both sensor nodes, connected to the RESTful Sensor Web network, do not require input parameters, and only read data using GET access method. Because of the fact that the main issue in sensor network implementation concerns reducing the amount of transferred data, using JSON for communications between clients and servers, is a good choice.

Two methods for reading data from both sensors of RESTful Sensor Web Network are given by Vujovic et al. (2013):

```
[server_address]/RESTfulSensorWeb/sw/b57045k10
[server_address]/RESTfulSensorWeb/sw/b57045k10/data
```

and

```
[server_address]/RESTfulSensorWeb/sw/ntc10k
[server_address]/RESTfulSensorWeb/sw/ntc10k/data
```

Although JSON is used for information exchange, the existence of two addresses contributes the amount of transferred data reduction. When the first address is used, the client gets JSON which contains only information about the measured value:

```
{"temp":"23.55"}
```

Using the second one, in addition to the measured value, a JSON with information of sensor type and sensor model are forwarded:

```
{"type":"thermistor","model":"b57045k10","temp":"23.8
7"}
```

Aggregated information about the sensors is accessible via three addresses:

```
[server_address]/RESTfulSensorWeb/sw/info
[server_address]/RESTfulSensorWeb/sw/info/data
[server_address]/RESTfulSensorWeb/sw/info/number
```

The first address returns the sensor number, data of all sensors and the measured values:

```
{"sensor_number":2,"data":[{"type":"thermistor","mode
l":"b57045k10","temp":"23.23"},{"type":"thermistor","
model":"ntc10k","temp":"23.07"}]}
```

The second address returns only information about sensors and measured values:

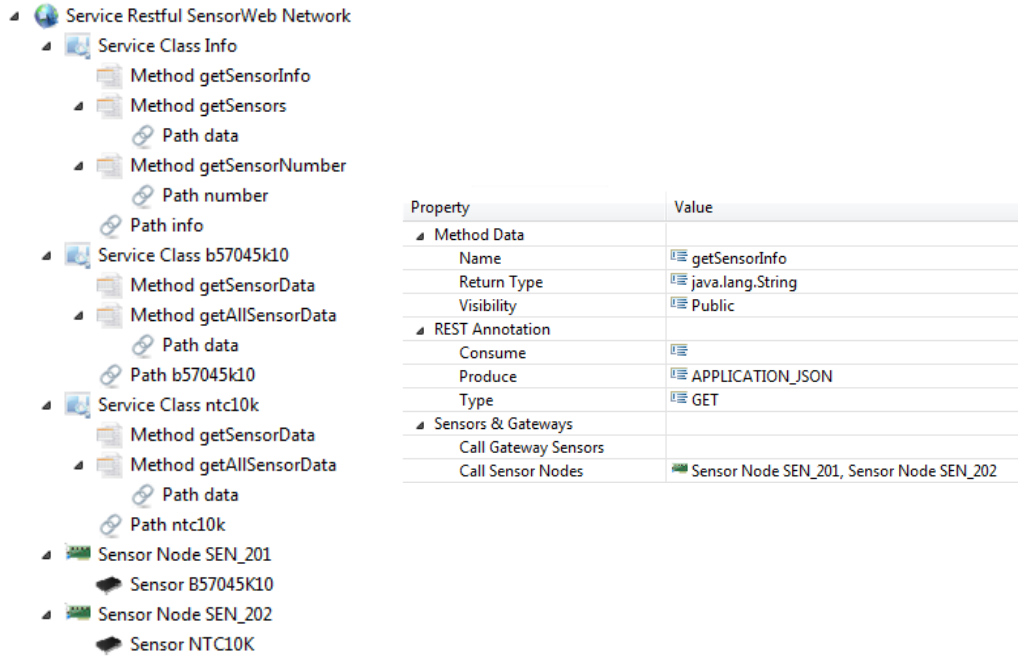
```
{"data":[{"type":"thermistor","model":"b57045k10","te
mp":"22.58"},{"type":"thermistor","model":"ntc10k","t
emp":"22.37"}]},
```

while the third address provides number of sensors in the sensor node:

```
{"sensor_number":2}.
```

Using the Editor of DSM for RESTful Sensor Web network as a part of Eclipse IDE, a solution for validation example is presented in Figure 5 a).

Figure 5. ***a)*** *An Editor Model of RESTful Sensor Web Network,* ***b)*** *A Property of getSensorInfo Method*



a) b)

As is shown in Figure 5 a), by creating graphical models of RESTful Sensor Web Network, complete information of system, methods, approaches, and the number of sensor nodes are obtained. The rules for creation of objects and their interconnections are based on the proposed meta-model. Each graphical widget represents a certain element of the network or service and is described by properties that describe its condition.

Figure 5 b) presents characteristics of RESTful services *getSensorInfo* method and unambiguously provides all the information necessary to implement the methods for accessing sensor data.

Using the Editor, it is possible to encapsulate implementation information and, at the same time, retain the ability to easily control the RESTful Sensor Web Network creation. The obtained detailed specification of RESTful Sensor Web Network enables designers to implement RESTful services easily. Upgrading it with a partially automate code generator, a complete solution for creating and designing information systems that manage Sensor Web networks, based on RESTful services, may be obtained.

It is important to note that it would be necessary to adopt some additional standards in order to achieve the automatic generation of complete (executable) code that can be manually extended by the developers. The explicit tagging of manually added code is necessary, enabling in such way the preservation of added code in case of automatic code regeneration.

Conclusion

The challenging issue of achieving WSN interoperability with IP networks has been recognized, and so has the need for an open resource-oriented architecture for building web services into the sensor networks.

Thus, the RESTful Sensor Web is an implementation of Sensor Web principles and functionalities following RESTful architectural paradigm. The sensors in resource-constrained WSN are treated as resources in REST architecture, integrated with the Web and accessed by Web services. The adoption of the lightweight REST web services concept shows the effectiveness and efficiency in terms of file size and communication time reduction.

Using DSM to create a RESTful Web service of Sensor Web, reduces sensor networks and information management system creation time and thus increases the overall productivity. The emphasis is switched from the technology to the design issues of the Sensor Web networks.

For this purpose a meta-model that describes RESTful Sensor Web Network is created and verified. For the validation purposes of a given meta-model, a simple editor is created. Verification is performed using a simple example of RESTful Sensor Web Networks design which has justified the introduction of DSM techniques for RESTful creation.

Future research will be focused on the meta-model extension with Jersey 2.5.1 specification, creation of graphical editors for a given meta-model and an automatic code generator for creating partial or full RESTful Sensor Web network.

References

Ali, F. 2010. A Middleware to Connect Software Applications with Sensor Web, *The International Journal of Technology, Knowledge and Society* 6(5): 27-35

Ben Maissa, Y. et al. 2012. Wireless Sensor Networks with VeriSensor, *PNSE'12 – Petri Nets and Software Engineering*

Brambilla, M., Cabot, J. and Wimmer, M. 2012. *Model-Driven Software Engineering in Practice*, Morgan & Claypool publishers

Budinsky, F. et al. 2003. *Eclipse Modeling Framework: A Developer's Guide*, Addison Wesley

Camilo, T. et al. 2007. GENSEN: A Topology Generator for Real Wireless Sensor Networks Deployment, SEUS, volume 4761 of *Lecture Notes in Computer Science, Springer*, pp. 436-445

Delin, K.A. 2005. Sensor Webs in the Wild, *Wireless Sensor Networks: A Systems Perspective*, Artech House

Di, L. 2007. Geospatial Sensor Web and Self-adaptive Earth Predictive Systems (SEPS), *Proceedings of the Earth Science Technology Office (ESTO)/Advanced Information System Technology (AIST) Sensor Web Principal Investigator (PI) Meeting*, San Diego, USA

Fajar, M., Hisazumi, K., Nakanishi T., and Fukuda, A. 2011. Applying Domain Specific Modelling for Enviromental Sensing Using Wireless Sensor Network, *Asian Journal of Information Technology* 10(7): 296-305

France, R. and Rumpe, B. 2007. Model-driven Development of Complex Software: A Research Roadmap, *Future of Software Engineering*, pp. 37-54

Jersey JAX-RS framework, Available: https://jersey.java.net/

Liddle, S.W. 2010. Model-Driven Software Development', Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.172.5995&rep=rep1&type=pdf

Ludovici, A. and Calveras, A. 2010. Integration of Wireless Sensor Networks in IP-based networks trough Web Services, Proceedings *of 4th Symposium of Ubiquitous Computing and Ambient Intelligence*, Spain

Martinez, A.P. 2011. Implementation and documentation of Sensor Web Enablement at KNMI, De Bilt, Stageverslag

NTC thermistors for temperature measurement, Type B57164, 2006, Available: http://eecs.oregonstate.edu/education/docs/datasheets/10kThermistor.pdf

Paternostro, M. and Hussey, K. 2009. Building RESTful Java™ Applications with EMF, *EclipseCon*

Pautasso, C., Zimmermann and O., Leymann, F. 2008, RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision, *17th International World Wide Web Conference (WWW2008)*, pp. 805-814

Potti P.K. et al. 2012. Comparing Performance of Web Service Interaction Styles: SOAP vs. REST, *Proceedings of the Conference on Information Systems Applied Research*, New Orleans Louisiana, USA

Ramadan, R. and El-Rewini, H. 2009. Deployment of Sensing Devices: A Survey, Available: http://rabieramadan.org/papers/deployment%20survey.pdf

Resistance table for 10kNTC Thermistor, Available: http://coldtears.lin3.siteonlinetest.com/files/10kNTC.pdf

Rouached, M., Baccar, S. and Abid, M. 2012. RESTful Sensor Web Enablement Services for Wireless Sensor Networks, *IEEE Eighth World Congress on Services*, pp. 65-72

Sadilek, D.A. 2008. Domain-Specific Languages for Wireless Sensor Networks, *Doctoral symposium of the Modellierung conference*

Sadilek, D.A. 2007. Prototyping Domain-Specific Languages for Wireless Sensor Networks, *ATEM'07: 4th International Workshop on Software Language Engineering*

Schreier, S. 2011. Modeling RESTful applications', *WS-REST ACM*, pp. 15-21

Stanley-Marbell, P. et al. 2008. *System Models in Wireless Sensor Networks*, ES Reports Eindhoven University of Technology Available: http://www.es.ele.tue.nl/esreports

Vujović, V. et al. 2013. Web Integration of REST Enabled Wireless Sensor Networks for Fire Detection, *International Conference on Applied Internet and Information Technologies*, Zrenjanin, Serbia, pp. 30-35

Yazar, D. 2009. *RESTful Wireless Sensor Networks Domain*, Master Thesis, Uppsala Universitet