

Modeling and Analysing Ubiquitous Systems Using MDE Approach

Amara Touil*, Jean Vareille*, Fred Lherminier†, and Philippe Le Parc*

*Université Européenne de Bretagne, France

Université de Brest - EA3883 LISyC

Laboratoire d'Informatique des Systèmes Complexes

20 av. Victor Le Gorgeu, BP 809, F-29285 Brest

Email: {amara.touil, jean.vareille, philippe.le-parc}@univ-brest.fr

†Terra Nova Energy

Z.I. de Kergaradec, 28 rue Victor Grignard, F-29490 Guipavas

Email: fredl@terranov.com

Abstract—The growth of industrial activities during the last decades and the diversity of industrial products require standards and common methodologies for building and integrating different parts. It is also required that working groups use the same terminologies and concepts needed for each domain. The Model Driven Engineering approach aims to give an answer while using a high level method based on models and transformations. In this paper, we use this approach to model ubiquitous systems. Those systems are composed of devices interconnected through various kinds of network and offer to get and send information. We present a model for this class of system and its use in the field of energy while studying real cases from our industrial partner Terra Nova Energy. This company aims to give solutions to monitor energy use and to reduce consumption. First results, where the use of a Model Driven Engineering approach makes it possible for our partner to improve and to get other points of view of his systems, are presented.

Keywords - Domain Specific Language, Model Driven Engineering, Analysis, Telecontrol Systems.

I. INTRODUCTION

Ubiquity is often define as being able to be in several places at the same time. In the field of Information Technologies, this definition can be refined in, at least, two ways that may be apparently opposites. The first approach is to considered as in [1] that users are surrounded with "intelligent" systems, that may delivers them needed information: in this case, computing facilities are used to locate users, to understand their environment, to anticipate their needs and to make it possible that all information they require is available anywhere at anytime. The second approach, is to offer people to be able to get information from a system located somewhere and to be able to act safely on it: in this case computing facilities are used to make it possible to be "virtually" present in several places at the same time. We place our work in the second approach in order to model and analyze telecontrol application as a kind of ubiquitous systems.

Terra Nova Energy (TNE) is an innovative company that provides solutions for data mining in the fields of electricity, hydraulic and pulse-energy. It integrates sensing, acting and communicating devices in telecontrol systems, for collecting data from different sites using various technologies. TNE's

systems allow real-time operating and provide processes for handling different data treatments.

In order to improve its development, this company is facing two problems: how to conceive remote monitoring systems as automatically as possible and how to fasten their on-site deployment ?

To answer these questions, we propose to use a model based approach, relying on specific components for telecontrol domain, and libraries to integrate industrial parts coming from various providers. Our idea is to build a framework [2] following the Model Driven Engineering (MDE) methodology [3][4] to create a telecontrol ontology and proper transformations for building, generating, analyzing and deploying such ubiquitous telecontrol systems.

Our aim is to define a generic meta-model and to use it for various systems, as case study examples, one of them being the TNE's system.

This paper is organized as follows: first of all, the MDE approach, Domain Specific Languages (DSL) and transformations that may be conducted are briefly introduced. Second, we introduce the generic meta-model proposal. Third, this general meta-model is applied to our case study, while defining an instance for TNE. Then, we explain how to carry a static analysis on a given instance and show first results. Finally we discuss our method and we finish by further work.

II. RELATED WORKS

In this section, we introduce some basic notions about the MDE approach and DSL, and how to conduct transformations on models.

A. The MDE approach

A model is an abstracted view of a system that expresses related knowledge and information. It is defined by a set of rules, used to interpret the meaning of its components [5][6]. A model is defined according to a modeling language that can give a formal or a semi formal meaning description of a system depending on modeler's intention. Modeling languages can be textual or graphical.

The model paradigm has gained in importance in the field of systems engineering since the nineties. Its breakthrough was favoured by working groups like the Object Management Group (OMG) [7] that has normalized modeling languages such as Unified Modeling Language (UML) and its profiles (such as SysML and MARTE for real-time systems). This group also provides the Model-driven architecture (MDA) software design standard. The MDA is the main initiative for Model Driven Engineering (MDE) approach.

According to OMG, four abstraction levels have to be considered: a meta-meta-model that represents the modeling language, which is able to describe itself; a meta-model level that represents an abstraction of a domain, which is defined through the meta-meta-model; a model level that gives an abstraction of a system as an instance of the meta-model; finally, the last abstraction level is the concrete system.

Tools have been defined to implement OMG standards. Some have general and wide purpose like those for UML language, others have been defined for specific and reduced class of applications like in [8] that aims to specify wireless sensor network systems in order to generate code. Another approach is to use Domain Specific Language (DSL) to specify a special semantic and/or syntactic rules for a class of application. One of DSLs definition is given by [9]: "A domain-specific language (DSL) is a programming language or executable specification language that offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain". In our case we define our own DSL for telecontrol, according to Eclipse Core (Ecore) meta-model [10].

B. Model transformation

In order to breath life in models [11], model transformations aims to exceed the contemplative model view to a more productive approach, towards code generation, analysis and test, simulation, etc. Models are transformed into other models that may be manipulated by specific tools or that may be transformed again into other models. Two kinds of transformation are used, exogenous if source and target transformation do not have the same meta-model and endogenous if source and target have the same meta-model. We use the latter here in order to perform static analysis of the built models.

Generally, static analysis deals with observing system and checking some properties before real system creation, production or code implementation. As example, formal verification and model checking techniques [12][13] are used for verifying models. Behavioural analysis are not beyond in the scope of this article. Static analysis are only conducted in simple cases, but relevant ones for TNE. In this paper we will focus on placement but others studies have been conducted (cost, bandwidth,...). As soon as the model is created, it may give answers to the engineer: Are my sensors, repeaters, routers placed in a proper location? Are there better configuration? etc..

Concerning the first question, there are several works treating these points like [14], which study Wireless Network

Sensors (WSN) placement for smart highways; it gives a solution based on geometric resolution algorithms in outdoor and open space. In our case, indoor deployment environments, including various fix and mobile obstacles, have to be studied. Component's placement depends on monitored zone, on the chosen topology, on the network capacity, etc. The second question can be adressed by optimizing the placement configuration.

Next section describes the meta-model proposed for telecontrol and its derivation for TNE's remote monitoring needs.

III. PROPOSED META-MODEL

In this section, we, describe the meta-model we are proposing to specify systems based on communicating objects. Only a high level view and the main concepts are presented. Starting from this meta-model, we derive a specific instance for TNE.

A. Generic meta-model

At a first abstraction level, we built our domain around systems containing entities that communicate with each other as stated in [15]: "A system is a construct or collection of different elements that together produce results not obtainable by the elements alone". In the meta-model proposal (Figure 1), a system, modeled by *System*, can contain other systems according to the *ownedSystem* reference, in order to provide the "system of the system" notion. Regarding entities, they are modeled by *Entity* and its system containment is referenced by the *itsEntity* relationship. Entity paradigm in our meta-model aims to be a generic and general concept formed by extracting common features from our specific telecontrol domain. It can be a logical or physical and can be also composed of other entities (*ownedEntity*).

Moreover, entities are described using several related concepts trying to keep information about their physical properties, communication facilities or behaviour. These concepts are modeled as follows:

- *Structure* element defines the interrelation or arrangement of different physical parts or the organization of elements that provides coherence, shape and rigidity to an entity. It may describe mechanical, chemical or biological aspects.
- Actions performed by entities are described by the *Task* concept. They can be internal, such as making computation, external such as sending information and also connected to the real world such as sensing or acting on a real device.
- *ContactInterface* element allows connection between entities. It can be a physical contact, or a logical interface depending on the specification level, on the refinement degree or on its own structure.
- An entity may contain *Data* related to itself or to its environment.
- *Message* element provides data exchange between entities. To send (or to receive a message) from entity A to entity B, A and B must be connected through *ContactInterfaces*, directly or, it may exist a path of entities that may forward messages.

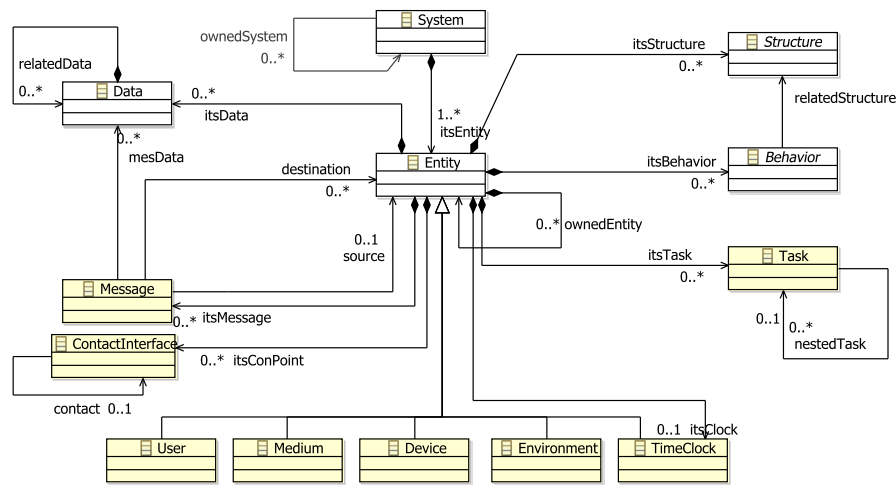


Fig. 1. A view of generic meta-model components (Basic package)

- To each entity, a *Behavior* concept is added, in order to describe the different tasks an entity may realize. From this concept, code generation or simulation may be refined.

As the *Entity* concept is very general, in order to improve our generic meta-model, different sub-entities have been specialized using the concept of heritage:

- 1) *Device* represents a physical component such as a sensor, a router, a computing unit, etc.
- 2) *Medium* element represents an entity deployed and used between other entities to specify communication's medium. In classical approaches, medium is often omitted and considers as perfect links. In our work, medium has its own structure, behavior, contact interfaces, datas, messages and tasks. Different mediums may be described and classified following their own specifications, wired or wireless for example.
- 3) *User* element is used to model a person interacting with other entities, it is a kind of human avatar. Like other entities *User* can have different presentation depending on model view.
- 4) The different entities composing the system are under some physical conditions and constraints that affect and influence the global behaviour. In our meta-model we introduce the *Environment* concept to describe such conditions. Because we are dealing with a complex system, the *Environment* is an entity of this system and not just an external element. In our modeling approach, *Environment* acts on internal elements in order to build a global circumstances of evolving system and gives context for ubiquitous entities and systems that are context-aware.
- 5) The concept of time is crucial for telecontrol systems that is why a *TimeClock* entity is added to our generic meta-model for measuring, recording or indicating time.

The presented generic meta-model intends to specify any

kind of systems containing communicating objects. At this abstraction level, we present just a first facet (also named *Basic package*), without getting into all the details of the meta-model.

B. Terra Nova Energy meta-model

In the previous section, a generic meta-model, at a first level of abstraction with some inheriting elements from *entity* such as *Device* has been presented.

According to the concept of inheritance, the Terra Nova Energy meta-model (Figure 2) is built as an extension and a generalization of the generic meta-model. The first element is the *TNESystem* that inherits from the generic element *System*. Other elements have been spread over two sets: generic devices and off the shelf devices that could be grouped in a specialized library.

1) *Terra Nova Energy's generic devices*: This first set of elements represents generic devices that allow to add a new industrial component to a library or to build a specific system. All these devices inherit from generic *Device*, imported from Basic package as shown in Figure 2. We present them successively as follows:

- *BoxaNova* models the main device of TNE systems. It collects information, sends orders and manages the entire flow of data between different devices and users. A *BoxaNova* device is generally created by some others devices depending on system configuration and deployment.
- *Router* models a device that handles message transfers between different TNE elements. It handles also some other functions like controlling repeaters, sensors, actuators and some other server facilities.
- *Concentrator* is used to model a collecting place of data coming from repeaters, sensors, actuators before sending them to routers depending on requests or preprogrammed sending tasks.
- *Sensor_ActuatorDevice* is used to capture measurement data and to send them or to act pursuant to an order.

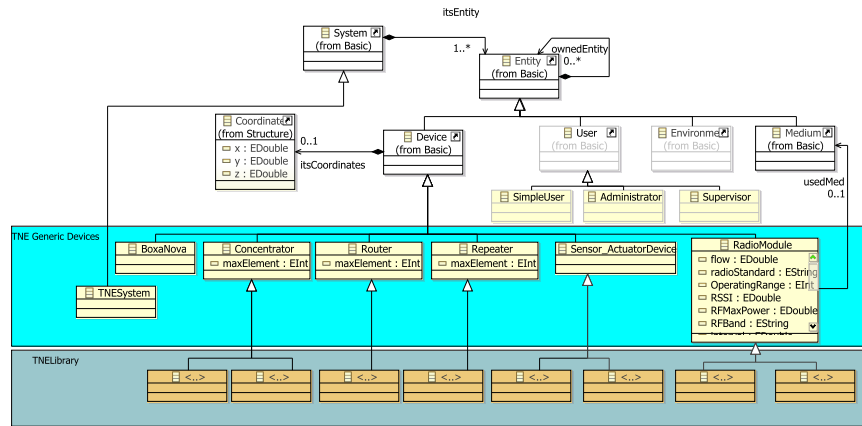


Fig. 2. A view of Terra Nova Energy meta-model

- *Repeater* is used to ensure the link between a *Concentrator* and *Sensor_ActuatorDevice* elements if they are not within reach.
- *RadioModule* models the device that ensures exchanging data messages between all other devices when wireless communication links are used.

Since TNE devices are specialized from the generic element *Devices*, they inherit all properties and references.

2) *Terra Nova Energy's off the shelf devices*: This second set of elements represents real devices ready to use in the framework. They are another level of specialization of the first set. They may be grouped in a kind of library (blue box in Figure 2), composed with industrial devices coming from different manufacturers and in order to be integrated into TNE systems. In this paper we neither describe this library of devices nor the manner of their built because we focus on some other properties and aspects of model analysis.

In the next section, we propose a methodology of analysis showing the usefulness of these presented meta-models, and their examination and validation on real industrial cases.

IV. MODEL ANALYSIS

In this section, we present how to exploit the MDE approach to analyse placement for a given instance, and automatic placement of repeaters and concentrators for an uncompleted system, where only sensor, actuator positions are known.

The first part describes our methodological approach, based on processing properties and applied constraints for model's elements.

A. Presentation

From the TNE domain specific meta-model, presented in the previous section, an instance that describes a real system of telecontrol may be defined. According to the user needs, this instance captures a particular concern for a system and gives the necessary elements and their relevant properties. Such a model requires some processing stages to meet the final requirements in terms of analysis. These processing stages are done in response to expected system constraints applied to

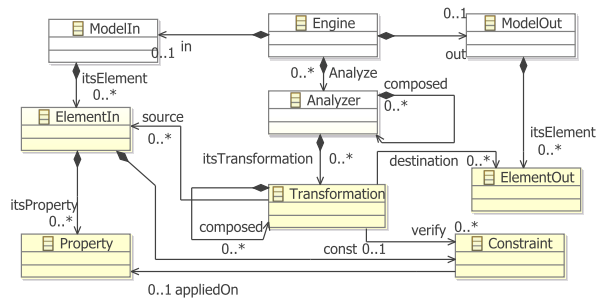


Fig. 3. A view of analyzer's engine model

its various elements. Figure 3 shows a model overview of the various elements used to carry out those treatments for analysis methodology purpose.

The first analysis component (*ModelIn*) loads the instance that describes a telecontrol system according to its meta-model. This instance is composed of elements called *ElementIn*. The *Engine* scans iteratively the input model, element by element, to find different properties and constraints. For each property the *Analyzer* checks if it fits well the associated constraint. When properties are verified, the element will be transformed by *Transformation* and treated by *ElementOut*. The *ModelOut* generates automatically output model formed by those transformed elements. Output model should be consistent to its meta-model that may be the input one or another depending on performed transformation.

Generally, input models have a tree shape structure and their components can be composed of other components. Properties and applied constraints can also have this composition criterion. In order to deal with this kind of structure and having a generic analyzer, possible compositions have been taken into account for analyzing and transforming tasks. In Figure 3, these composition aspects are represented by the *Composed* reference. The next part of this section deals with an application of this methodology to a system. First of all, analysis for

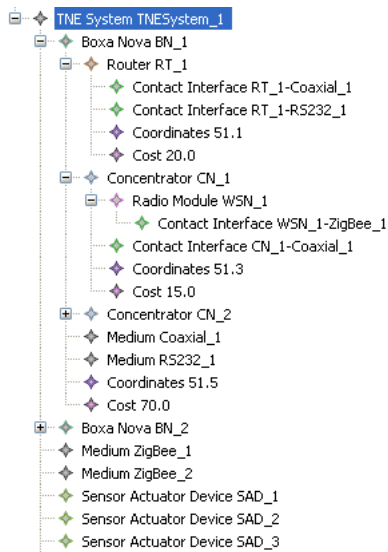


Fig. 4. A screen shot of manual instance creation

a completely described given instance are conducted to verify that elements are well placed and may communicate together properly. Second we tell that the Model Driven Engineering approach used, may also be helpful to build instances, starting with an uncomplete system, where only sensors and actuators are known, to add necessary repeaters and concentrators.

B. First case study with a complete TNE's instance

In this first case study, an instance of a TNE meta-model has been created with a tree editor as shown in Figure 4. It is an abstraction of a *TNESystem* named *Telecontrol-System_1* composed of two *BoxaNova* that manage twenty *Sensor_ActuatorDevices*. Each *BoxaNova* consists of two *Concentrator* elements and a *Router*. Communication between these components uses two *Mediums*; a coaxial cable that connects the first concentrator and the router and a RS232 cable that provides serial communication between the second concentrator and the router.

Sensor and actuator devices use a ZigBee protocol (IEEE802.15.4 standard) [16] to communicate with the *BoxaNova* concentrators. To achieve this task, they use *radioModules*. Concentrators also have their own radio modules. In general, wireless propagation environments in TNE systems may have different attenuations. Thus, each communication between a sensor/actuator and its repeater or concentrator is defined by a medium with relevant properties and necessary (contact) interfaces. In our model, this information is kept by properties such as the attenuation value, speed of transmission, signal length, etc. Connection with the medium and other elements is ensured by the definition of two entities as *ContactInterface*; one for the medium and the other for the associated element.

The model also captures other information like the physical location (x, y coordinates) of the different parts of the system. In this way, *Communication vs. placement* analysis have been

conducted. In fact, component's placement in TNE systems is crucial to ensure good communication and proper information transfer. As the model is fully known, an analyzer can be created to verify that every *Sensor_ActuatorDevice* element according to its coordinates may communicate to its corresponding concentrator. This verification takes the attenuation between communicating elements into account, specified as a property of the medium, which links them. Another analyzer can be defined and used for inspecting the number of sensor/actuator devices for each concentrator and its compliance to specified *maxElement* property.

From information analysis text file obtained (see transformation model to text in [11]), manual corrections may be realized. This task is manageable for small systems but gets difficult and very expensive for large ones with many components. The next part gives an illustration of a methodology that can treat this difficulty.

C. Second case study with a partial TNE's instance

In this second case study, only the number and the position of sensor/actuator devices are given. The instance is partial, and the objective is to automatically find a solution for repeater's and concentrator's placement. This problem is a very complex one and several solutions may be found in literature [14][16][17].

In our case an input model is generated automatically with the required properties. As a first step, an instance of the TNE meta-model is created with a system *TNESystem*, a number of *sensor_ActuatorDevice* and an environment that represents a factory hall. In our case, we suppose that radio measurements were made in the environment and have identified attenuation values.

In a second step, an analysis engine for placement, composed of three analysers, is used:

- 1) The primary analyzer performs an initial refinement of the input model. First, it divides the environment, specified in the input model, into zones where mediums have the same attenuation so that the range of sensor-actuator radio modules, who are currently in, reach the middle. This range is calculated by applying attenuation value imposed in the input model. Then, depending on the number of sensor/actuator devices, it creates the necessary repeaters with their radio modules, and adds them to the input model with coordinates in the vicinity of the centre of the sub-environment. Finally, it creates connections (*ContactInterface*) between repeaters and sensor/actuator elements and adds them to the model.
- 2) A second pass of refinement with the same treatment approach is made by the second analyzer. Its objective is to connect repeaters with concentrators. The input model environment is split up into several zones with new radio modules ranges of repeaters. The number of added concentrators will depend on their capacity and also on the number of repeaters within reach.
- 3) The third analyzer performs an optimization on the obtained model that contains sensor/actuator devices that

can communicate directly with concentrators without the use of a repeater. This analyzer checks for each sensor/actuator device if it reaches any concentrator and changes its connection from a repeater to a concentrator. Repeaters without connected sensor/actuator devices will be removed.

At the end of these three analyzer passes, a new output model is obtained. It is composed of initial sensor/actuator devices and added components (repeaters, concentrators and connections). Figure 5 shows a simple placement layout of TNE system components obtained with the following initial model inputs: 200 sensors/actuator devices (location of these elements has been chosen randomly for the test), an environment that represents a factory with 800 meters length and 600 meters width and an attenuation value of 5 decibels by meter. Red rectangles represent repeaters, blue triangles represent concentrators and green circles represent sensor/actuator devices.

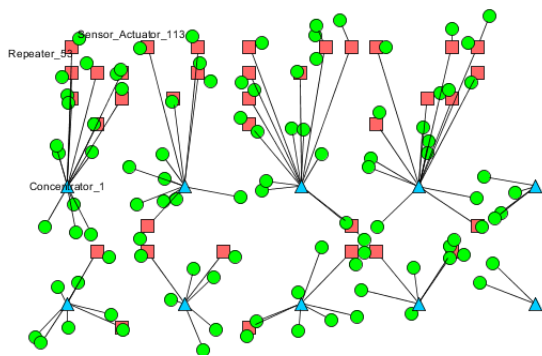


Fig. 5. Placement layout

The generated model allows us to have a first proposal for the positions of the repeaters and of the concentrators in the environment. This obtained placement is of course not optimal, as simple hypothesis have been used. Indeed, the presence of mobile obstacles in the environment and their influence on signal propagation should be considered. To optimize the placement and to ensure the highest data rate optimal algorithms should also be used. The orientation of elements should also be taken into account.

Nevertheless, our objective is to show that using model analysis in ubiquitous systems in MDE approach may support engineers in the design of their systems and verifying some properties in earliest stage without real implementation such as communicating object placement.

V. CONCLUSION AND FUTURE WORKS

In this paper, a generic meta-model for modeling ubiquitous telecontrol systems and, specially telecontrol systems is proposed. A variant of this metamodel is specified to fit the needs of Terra Nova Energy.

The introduction of the Model Driven Engineering approach in this field allows us to exceed the contemplative dimension of models towards productive models. The presented cases

studies highlights this approach by proposing a possible model for placement analysis for an example of the TNE system, using transformations.

In the future, we would like to improve our generic meta-model in order to take the behaviour of the different components into account. We would like to move from static analysis to simulation with a complete integration of a system inside its environment.

ACKNOWLEDGMENT

This work is supported by the city of Brest, "Brest Métropole Océane", and performed in partnership with Terra-Nova Energy. We thank them for their help.

REFERENCES

- [1] M. Weiser, "The computer for the 21st century," *Scientific American Special Issue on Communications, Computers, and Networks*, 1991.
- [2] P. Le Parc, A. Touil, and J. Vareille, "A model-driven approach for building ubiquitous applications," in *The Third International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies - UBICOMM 2009*, Oct. 2009.
- [3] S. Kent, "Model driven engineering," *Lecture notes in computer science*, pp. 286–298, 2002.
- [4] F. Fleurey, J. Steel, and B. Baudry, "Validation in model-driven engineering: testing model transformations," in *Workshop WS5 at the 7th International Conference on the UML, Lisbon, Portugal*, 2004.
- [5] S. Gerard, F. Terrier, and Y. Tanguy, "Using the model paradigm for real-time systems development: Accord/uml," *Lecture notes in computer science*, vol. 2426, pp. 260 – 269, 2002.
- [6] D. Moody, "Graphical Entity Relationship Models: Towards a More User Understandable Representation of Data," *Lecture Notes in Computer Science*, vol. 1157, pp. 227–244, 1996.
- [7] S. Cranefield and M. Purvis, "UML as an ontology modelling language," in *Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, vol. 212, 1999.
- [8] F. Losilla, C. Vecente-Chicote, B. Alvarez, A. Iborra, and P. Sánchez, "Wireless sensor network application development: An architecture-centric mde approach," *Lecture Notes in Computer Science*, vol. 4758, p. 179, 2007.
- [9] A. van Deursen, P. Klint, and J. Visser, "Domain-specific languages: an annotated bibliography," *SIGPLAN Notices*, vol. 35, no. 6, pp. 26–36, June 2000.
- [10] R. Gronback, "Eclipse Modeling Project: A Domain-Specific Language (DSL) Toolkit," *Addison-Wesley Professional*, 2009.
- [11] K. Czarnecki and S. Helsen, "Classification of model transformation approaches," in *Proceedings of the 2nd OOPSLA Workshop on Generative Techniques in the Context of the Model Driven Architecture*, 2003.
- [12] T. Alenljung and B. Lennartson, "Formal Verification of PLC Controlled Systems Using Sensor Graphs," in *Proceedings of the fifth annual IEEE international conference on Automation science and engineering*. The Institute of Electrical and Electronics Engineers Inc., 2009, pp. 164–170.
- [13] A. Voronov and K. Aakesson, "Verification of process operations using model checking," in *CASE'09: Proceedings of the fifth annual IEEE international conference on Automation science and engineering*. The Institute of Electrical and Electronics Engineers Inc., 2009, pp. 415–420.
- [14] S. Ghosh and S. Rao, "Sensor network design for smart highways," in *CASE'09: Proceedings of the fifth annual IEEE international conference on Automation science and engineering*. The Institute of Electrical and Electronics Engineers Inc., 2009, pp. 353–360.
- [15] I. C. Committee, "A consensus of the incose fellows," International Council On Systems Engineering, Tech. Rep., 2006. [Online]. Available: <http://www.incose.org/practice/fellowconsensus.aspx>
- [16] P. Baronti, P. Pillai, V. Chook, S. Chessa, A. Gotta, and Y. Hu, "Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and ZigBee standards," *Computer Communications*, vol. 30, no. 7, pp. 1655–1695, 2007.
- [17] X. Cheng, D. Du, L. Wang, and B. Xu, "Relay sensor placement in wireless sensor networks," *Wireless Networks*, vol. 14, no. 3, pp. 347–355, 2008.