UML Profile for Wireless Networks: the case of 802.15.4 standard

April 30, 2007

Abstract

Wireless networks based on IEEE 802.15.4 standard [7] have very specific requirements in terms of complexity and power consumption, since these networks are composed mainly by very small battery-powered devices. Such kind of network fits the concept of Wireless Personal Area Networks (WPANs) that involve little or no infrastructure, allowing small, power efficient, inexpensive solutions to be implemented for a wide range of devices. It defines only the bottom two layers of the International Standard Organization (ISO) Open System Interconnection (OSI) protocol reference model [4]: the Physical (PHY) and the Medium Access Control (MAC) [16].
In the present work the possibility - based on state of art works - to create a UML profile for the IEEE 802.15.4 standard is illustrated. Profiles enable reusability as the system becomes a collection of self-contained modules or components that can be grouped in many different ways resulting in a great number of possible network solutions without the need of a complete redesign. We also show the application of Model Driven Architecture (MDA) concepts that leads to a platform independent model, providing interoperability and portability to the developed profile.

1 Introduction

A wireless sensor network (WSN) is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions at different locations. Each node in a sensor network is equipped with a wireless communication device, a small microcontroller, and a battery. The size of a single sensor node can vary from shoebox-sized nodes down to devices the size of grain of dust [1]. Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed and bandwidth.
The IEEE 802.15.4 standard was chartered to investigate a low data rate solution with multi-month to multi-year battery life and very low complexity. It is operating in unlicensed, international frequency bands [1] and specifies physical and MAC layers for low power and low throughput wireless personal area networks that is extensively used for wireless sensor networks.
Currently, each sensor vendor implements the standard independently from the other, leading to serious interoperability problems [10]. Furthermore, the text specification was not written with the aim of increasing development productivity, so it is not possible to easily identify blocks that can be implemented in parallel.
With UML 2[22] it is possible to develop very sophisticated models and profiles for nearly any type of system (hardware, software, middleware). Using UML it is possible to follow some design techniques such as Service Oriented Architectures (SOA) [20] or MDA [9] in order to design models and/or profiles with a good level of interoperability, reusability and portablity.
The aim of this work is to develop an UML profile for the IEEE 802.15.4 standard in the form of a platform independent model that can be reused and customized by different vendors and at the same time guarantee interoperability among all devices that follow this specification.

2 Background Review

2.1 Protocol Modeling

2.1.1 SDL

SDL[18] is a Specification and Description Language standardized as ITU (International Telecommunication Union) Recommendation Z.100. It has been widely used in the telecommunications field because of its ability to be used as a wide spectrum language from requirements to implementation, suitability for real-time, stimulus-response systems and extended finite state machines features.
UML 1.X has grown popular to depict many aspects of specifications, but still it lacked well defined semantics, for this reason ITU's Z.109 recommendation was created. It consists of a UML 1.X profile mapping a subset of SDL to a subset of UML. The combination of SDL and UML 1.X provides means to express detailed bahavior with formal semantics and to formally define the internal structure of composite entities.

2.1.2 Patterns for Protocol System Architecture

In [13] design patterns for communication systems - consisting of users, communication protocols and communication media - are discussed. The main focus is the protocol controlling communication between user and media, but it can be generalized to be used with other kinds of specifications (see e.g. [11]).
Three patterns - namely Protocol System Pattern, Protocol Entry Pattern and Protocol Behavior Pattern - are presented and illustrated taking as a test case a

[1] 2.4 GHz and 912MHz or 868MHz depending on geographical areas.

TCP/IP stack design using Conduits+ [3] and SDL. UML is mentioned as a possible modeling language to be used for protocol specifications. The proposed methodology uses a top down approach starting from a high level representation with the major components of the system which are then detailed and finally it ends with a behavioral description that integrates all components.

2.1.3 Using UML Models for Performance Analysis of Network Systems

The models presented in [11] are based on [13] and ITU's Z.109 [5] specification. Basically it applies the patterns of [13] using UML 2.0 features to obtain a methodology to design and test a system.
It is possible to define a modeling methodology by following Z109 mapping, taking advantage of pre-existing SDL models and then applying the patterns of [13]. Such methodology leads to a complete definition of a protocol or a network system and is composed by five phases:

1. Requirements Definition: Traditional requirements definition with definition of all the non functional requirements for the communication component (like bandwidth for example).
2. Architecture Specification: UML class and architecture diagrams applying the patterns. It is necessary to identify the active classes and show all communication interfaces. The most important result of this phase is the architecture diagram typically represented as a class or component diagram, or even both of them.
3. Behavioral Specification: During this phase, the overall behavior of the system is modeled through UML 2.0 StateCharts (compatible to a SDL flow chart).
4. Simulation Scenario Specification: uses collaboration diagrams to specify how components communicate to each other. This diagram shows information about the links that allow this communication and also information about workload on each node and network protocols to be used.
5. Results: Calculate performance statistics from traces generated during simulation. This requires the use of specialized tool, like Simmcast for example [19].

2.2 UML 2.0

The new release of UML language adds new features and fills some gaps left by the previous version. UML 1.X did not have adequate modeling capabilities for business and large scale systems and it did not allow modeling of non functional aspects, leaving space for misuse and misinterpretations.
UML 2.0 has been created with the intention of adding new capabilities for large scale systems and achieving great semantic accuracy and concepts consolidation. The new version is more precise, eliminates semantic overlaps and presents improved extension mechanisms. It also brings the concept of structured classes with internal and external structures, where the external structure has multiple interaction points called *ports*. Each port is dedicated to a specific purpose and presents the proper interfaces for that.
This new concept, together with the UML collaboration concept, makes UML 2.0 appropriate for modeling protocols. Using UML 2.0, protocols are modeled as a set of interconnected interfaces whose features are invoked according to a formal behavior specification and ports assume protocol roles according to their provided and required interfaces.

2.3 Model Driven Architecture

Model Driven Architecture (MDA)[9] is a design approach launched in 2001 by OMG[12]. It focuses primarly on the functionality and behavior of a distributed application or system rather than the technology in which it will be implemented. It divorces implementation details from business functions and hence, it is not necessary to repeat the modeling process for each technology that may be used.
UML is the language used to build platform independent models in order to achieve the three MDA's primary goals: *interoperability, reusability and portability.* Since UML is an universally accepted modeling standard, it is possible to create systems and applications that are portable and interoperate across a large set of systems, from embedded to desktop, servers and mainframes. MDA provides technique to :

- Specify a system independently of the platform that supports it (PIM);
- Specify platforms;
- Choose a particular platform for the system;
- Transform the system specification into one for a particular platform (PSM).

2.4 Service Oriented Architecture

An architecture can be seen as a specification of elements and connectors of a system and the rules of interaction among them. SOA is a design approach that defines the interaction among architecrural elements in terms of services that can be accessed without knowledge of the underlying platform implementation.
In a SOA, the components expose their interfaces as services in order to provide services sharing and increase reusability. In this way, the service interface does not depend on the implementation and, thus, changes in the internal behavior of an element does not imply changes in the the components using its services.
SOA and MDA are related since both of them target reusability and interoperability and the combination of both is comonly used as we can see in [15, 6].

3 Methodology

The goal of this work is to obtain a platform independent model which is reusable and guarantees interoperability between devices. The process followed to build the UML profile for IEEE 802.15.4 we will show is based on [11], where the following three steps are

suggested: requirements definition, architecture specification and behaviour specification.
This work focuses on the development of a PIM using component-based approach to facilitate the insertion of reusability points [21, 8]. UML 2.0 defines a component as a modular unit of a system that encapsulates its contents behind the interfaces and it is replaceable within its environment.
Therefore, each protocol layer fits perfectly the UML 2.0 definition of a component and will be represented as one with well known interfaces to interact with other components as defined by the protocol specification. This provides a modular design that allows different combinations of designed components.
Moreover it is possible to have a lot of different implementations for the same layer, each one prioritizing a different aspect of the protocol (e.g. optimizing power consumption, bandwidth or cost).
There are two different strategies for developing a component model, either top-down or bottom-up. The first case provides a good mechanism to obtain a complete overview of the system in a very fast way, which is particularly important for parallel modeling. The bottom-up approach is useful when there is a collection of classes that have been already developed. The system description must start from a component-based design in order to enable or increase reusability or even a co-design approach.
Since this work has not started from a set of already designed classes and our goal is to generate a reusable design, it has been decided to use the top-down approach, as described by the *Whole-Part* design pattern [2]. Therefore, we start the design from a high level of abstraction and then decrease it, increasing the details of each described component as we decrease the abstraction level.

4 UML Profile for 802.15.4

Figure 1 shows how MAC and PHY layers of the IEEE 802.15.4 standard are represented using a component based approach. Each layer is represented as a component and both components present provided and required interfaces for communication with upper and lower layers. It is clear from the figure that each layer is modeled as a component that offers and requires services through its ports and interfaces, therefore each layer can be designed and implemented independently as long as the designer follows the interaction rules established by the ports and interfaces of each component.

4.1 Physical layer

The PHY layer offers two set of services: PHY Data services and PHY management services, each one of them accessed through its own Service Access Point (SAP).Following SOA principles, each SAP was designed as UML port with the necessary interfaces.
Reducing the level of abstraction, the PHY layer was divided in two subcomponents: the Physical Layer

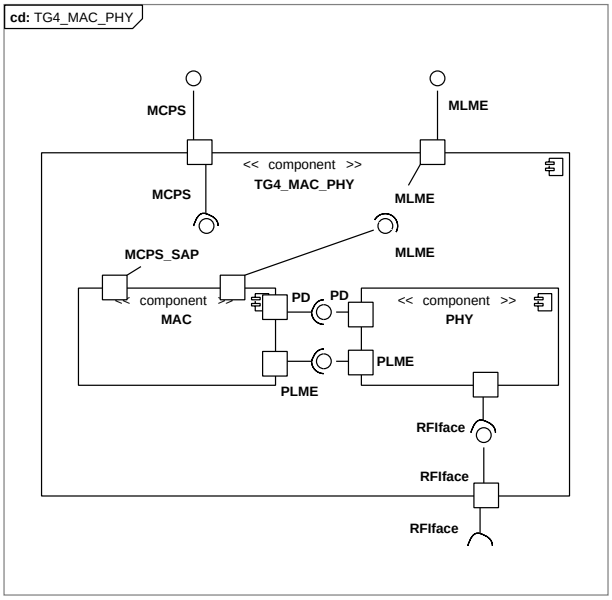

Figure 1: IEEE 802.15.4 high level description using UML 2.0

Management Entity (PLME) and the Physical Data (PD). The result is the component diagram in Figure 2.

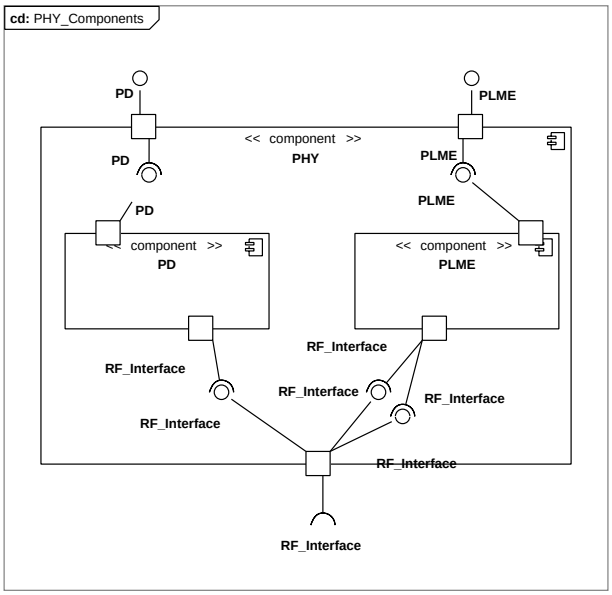

Figure 2: Physical layer with its subcomponents

4.1.1 PD component

The PD service supports *Mac Protocol Data Unit*s (*MPDU*s) transfers between peer MAC sub layer entities. This subcomponent offers three services:

- PD-DATA.request: requests the transfer of an MPDU from the MAC sub layer to the local PHY entity
- PD-DATA.confirm: confirms the end of the transmission of an MPDU from a local MAC sub layer entity to a peer MAC sub layer entity

- PD-DATA.indication: indicates the transfer of an MPDU from the PHY to the local MAC sub layer entity.

PD-DATA.confirm primitive is not actually a service provided by PD component, instead, it is a confirmation message that PD must send, when the data transfer is complete, in response to a PD-DATA.request primitive and thus it was modeled as type returned by a PD-DATA.response call.
PD-DATA.indication is generated by the PHY entity and issued to its MAC sub layer entity to sign the transfer of a received packet. As for PD-DATA.confirm, PD-DATA.indication can not be classified as service, and therefore it was left in the control part.
Figure 3 shows the UML component diagram representing the PD component with all classes and methods related to the services.

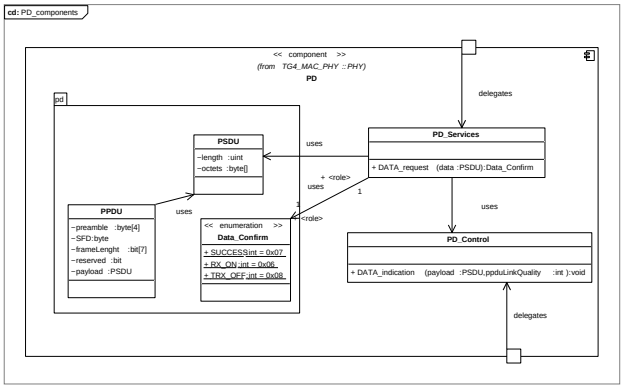

Figure 3: Internal view of the Physical Data component

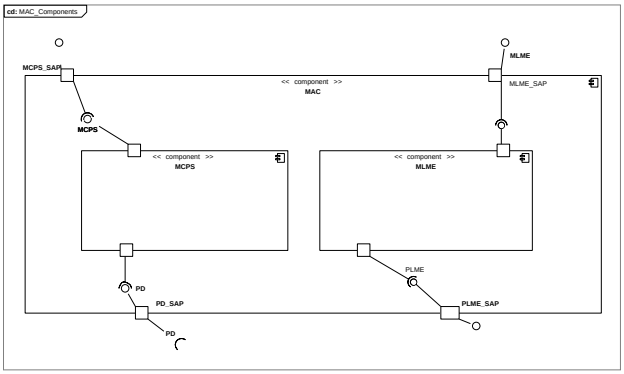

Figure 6: Internal view of the PD component

4.1.2 PLME component

The Physical Layer Management Entity (PLME) component is the UML representation of the management service of the PHY layer. The PLME-SAP, represented in the component diagram by the PLME-SAP port, allows the transport of management commands between the MAC Layer Management Entity (MLME) and in the PLME.
The PLME_SAP port delegates messages to the *PLMEServicesImpl* class which actually implements the management primitives. Many data classes were required to meet the specification, specially enumerations of possible status values returned by functions in the provided interface. Figure 4 shows the component diagram containing also the data classes.

4.2 MAC Layer

The MAC layer handles all access to the physical radio channel and provides an interface between the next higher layer (not specified by IEEE 802.15.4 standard) and the physical layer (PHY). Conceptually it is composed by a MAC Common Part Sublayer (MCPS) and the MLME that provides the service interfaces through which management functions are invoked.
The MAC layer provides two services, accessed through two different SAP's:

- MAC data service, accessed through the MCPS data SAP (MCPS-SAP), and
- MAC management service, accessed through the MLME-SAP

Similarly to the PHY model, the MCPS and MLME entities were modeled as subcomponents and their SAPs were modeled as UML 2.0 ports. Figure 6 shows the component the subcomponents and the ports.

4.2.1 MCPS component

MCPS supports the transfer of protocol data units (PDUs) between service specific convergence sublayer (the higher layers in the stack, not specified by IEEE 805.4.15) entities.
Because the IEEE specification allows the existence of full function devices (FFD) and reduced function devices (RFD), there are some MCPS primitives that are optional for FFDs. To model this, we took advantage of the powerful UML 2.0 stereotyping mechanism and such primitives became stereotyped functions with the *optional* tag.
Figure 7 depicts the UML component representing the MCPS entity. For the sake of simplicity, the diagram in Figure 7 shows only the interface and control classes, the data classes were kept in a separate diagram not shown here beacuse of limited space.
The component diagram shows two ports: the PD

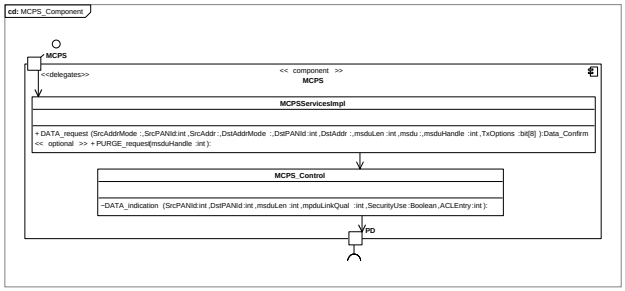

Figure 7: MCPS component representation

port that communicates MCPS with the corresponding

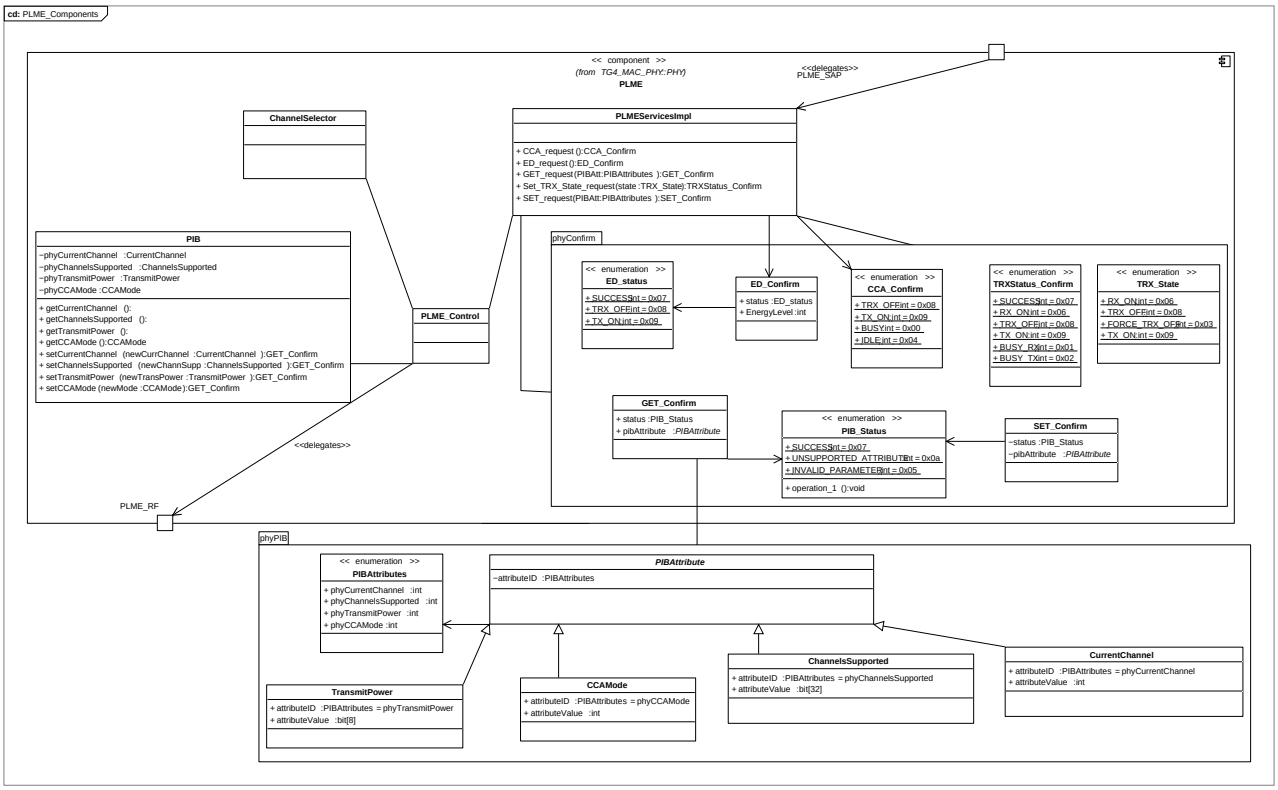

Figure 4: PLME component and its data classes

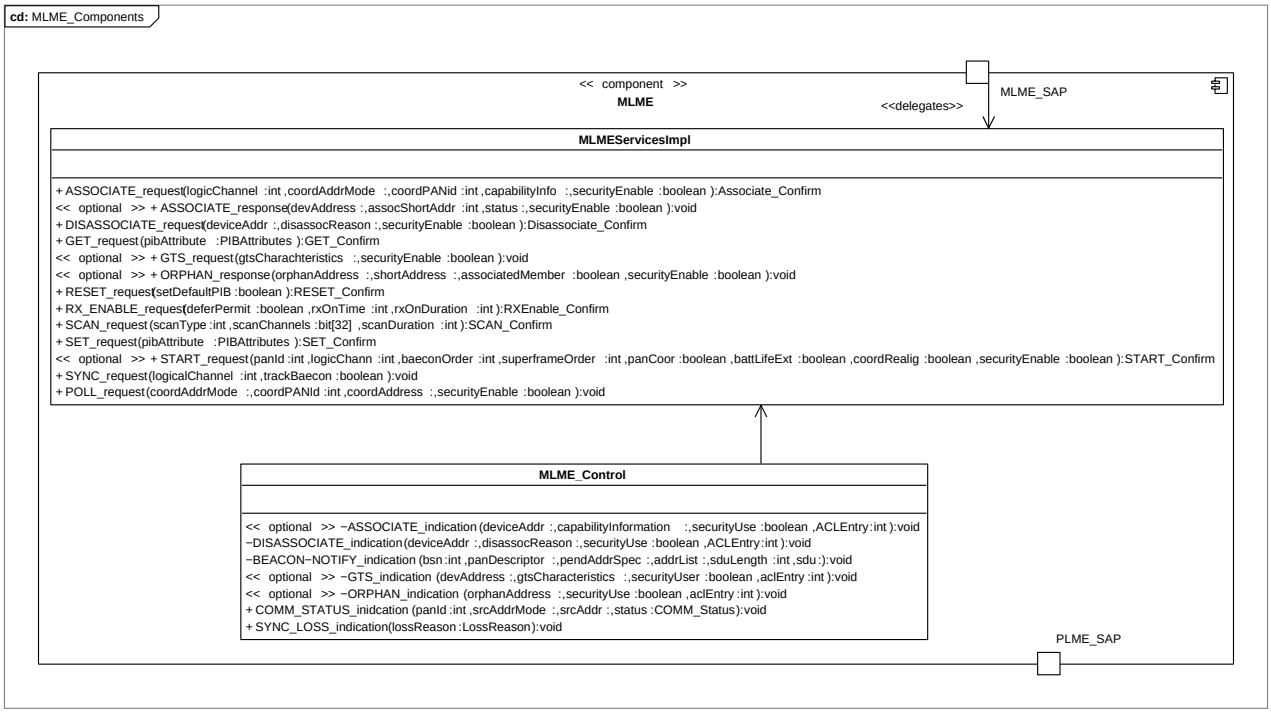

Figure 5: MLME component diagram

service in the PHY layer and the MCPS port that communicates with the higher layers and delegates messages arriving to it to the *MCPSServicesImpl* class that actually implements the services provided by the MCPS.

4.2.2 MLME component

The MLME component represents the management service of the PHY layer. It provides services that allow the transport of management commands between the next higher layer and the MLME. The services are

provided via MLME_SAP, represented in the UML 2.0 component diagram as a port with the same name. Figure 5 shows the component diagram.
Likewise the MCPS component, there are some optional functions signaled with the *optional* stereotype and to keep the diagram readable, the data classes were separated from the control and boundary classes (also not shown here due to limited space).

5 Conclusions

In this work we have shown that it is possible to build a UML profile for the IEEE 802.15.4 standard. Moreover, we demonstrated that such profile can be developed in a way to enable reusability in order to guarantee interoperability between devices and increase productivity during the implementation phase, since such kind of profile allow parallel design and implementation.
UML 2.0 has proven to be a worthy modeling language for systems specification and its combination with MDA concepts can lead to great results in terms of productivity during the design and implementation of systems.
The profile proposed in this work depicts the static structure of the IEEE 802.15.4 standard and offers an excellent level of freedom for the behavioral design. It is posible to build several different bahavioral models (such as Sequence Diagrams or State Machines) that fit the architecture represented in our profile and, hence, are comapatible with the IEEE standard. Furthermore, the methodology described here was carefully developed in order to be useful for designing other wireless protocols that can be divided in layers or components.
In order to represent precisely the IEEE standard, making use of UML 2.0 features we used two different tools: Rhapsody 6.1 [17] and Poseidon UML (versions 4.2 and 5.0) [14]. The former was used for its distinguished ability to represent signals and the latter for its good component representation features. Rhapsody 6.1 does not support component representation with the desired semantics and Poseidon does not give any special representation for physical signals, so using them combined was the best solution in the current tools scenario, resulting in a very precise UML 2.0 model.

References

[1] http://robotics.eecs.berkeley.edu/ pister/smartdust/. website.

[2] Buschmann F., Meunier R., Rohnert H., Sommerlad P., and Stal M. *Pattern-Oriented Software Architecture - A System of Patterns*. J. Wiley and Sons Ltd., 1996.

[3] Huni H., Johnson R., and Engel R. A framework for network protocol software. *ACM*, 1995.

[4] Zimmermann H. Osi reference model - the iso model of architecture for open systems interconnection. *IEEE Trans.Commun*, April 1980.

[5] Itu-t z.109, 1999.

[6] Boomberg J. The role of the service-oriented architect. *The Rational Edge*, May 2003.

[7] Gutièrrez J., Callaway E., and Barrett R. *Low-Rate Personal Area Networks - Enabling Wireless Sensors with IEEE 802.15.4*. IEEE Press, 2003.

[8] DaCruz A. Lewis D., Malbon C. Modelling management components for reuse using uml. *Proceedings of the 6th International Conference on Intelligence in Services and Networks*, 1999.

[9] Mda guide version 1.0.1, June 2003.

[10] Golmie N., Cypher D., and Rebala O. Performance evaluation of low rate wpans for medical applications. *Military Communications Conference*, 2004.

[11] Wet N. and Kritzinger P. Using uml models for the performance analysis of network systems. *Computer Networks*, December 2005.

[12] www.omg.org. website.

[13] Turunen M. Parssinen J. Patterns for protocol architecture. *Proceedings of the 7th Conference on Pattern Languages of Programs*, August 2000.

[14] http://www.gentleware.com/. website.

[15] Radhakrishnan R. Model driven architecture enabling service oriented architectures. march 2004.

[16] Marks R.B., Gifford I.C., and O'Hara B. Standards in ieee 802 unleash the wireless internet. *IEEE Microwave*, June 2001.

[17] http://www.ilogix.com/sublevel.aspx?id=302. website.

[18] http://www.sdl-forum.org/sdl/index.htm. website.

[19] http://inf.unisinos.br/ simmcast/. website.

[20] http://www.service-architecture.com/web-services/articles/service-oriented_architecture_soa_definition.html. website.

[21] Mens T., Steyaert P., and Lucas C. Giving precise semantics to reuse and evolution in uml. *ICSE98 International Workshop on Principles of Software Evolution*, 1998.

[22] www.uml.org. website.