Communication Stack Customization for Wireless Sensor Networks

Fadila Khadar, David Simplot-Ryl
IRCICA/LIFL, Univ. Lille 1
CNRS UMR 8022, INRIA Lille-Nord Europe, France
Email: {fadila.khadar,david.simplot}@lifl.fr

ABSTRACT

Communication in Wireless Sensor Networks (WSN) is a big issue. Designing an energy-efficient networking stack requires expertise, as the programmer must carefully design each layer of the stack and their interactions. This paper describes our on-going research on networking stack customization for WSN. We present an architecture which aims at simplifying the process of building an application by providing an energy-efficient and application-specific protocol stack.

1. INTRODUCTION

A Wireless Sensor Network (WSN) is a set of tiny autonomous objects with limited resources which cooperate in order to perform a common task. Once deployed, these sensors need no infrastructure to maintain the network: they are self-organized. WSNs cover a wide range of applications: from intrusion detection in a military field to bird observation or health monitoring [1]. In order to perform these tasks, communication between nodes is mandatory. When designing the application, the programmer cannot only concentrate on the application behavior. They also have to develop a protocol stack, which will handle all the communications between nodes. Communication interfaces, such as *send()* or *receive()* are implemented in this networking stack. As the communication unit is the most energy consuming component of a sensor, the network stack has to be carefully design in order to use it as seldom as possible.

We propose a programming model *i.e.,* a model that eases wireless sensor network application design by allowing the developper to focus only on how the application acts.

This paper is organized as follows. The next Section discusses related work. In Section 3 we outline our solution and give the architecture overview. In the last section, we conclude and discuss future work.

2. RELATED WORK

Offering an easier way of programming sensor networks is a very active research area. A survey of programming models for sensor networks can be found in [5]. The authors of [4] propose an application development framework (ADF) for programming sensors. A programming model, a program representation, a template of the runtime system and a compiler are provided in order to translate the application behavior into a program for sensors. Yet, the ADF does not customize the network stack to the application. It offers ways of optimizing the protocol stack by for instance changing the MAC protocol, but it is neither automatically done nor application specific.

A framework for modeling sensor networks is proposed in [3]. It is based on an ontology to describe general features of the network such as topology, network setting, hardware, and data flow. Each protocol is assigned a score for each feature (1,0,-1 for favorable, neutral or unfavorable behavior). The protocol with the highest score is chosen. The problem is that this protocol does not always fit the network features. A protocol that requires location-awareness can be chosen even if it is not available in the network. The authors propose to modify the protocol to incorporate the missing mechanisms. Besides, the score assigned to protocols is rather subjective.

We propose a protocol choice based on their analytical analysis of their energy consumption. To make sure that the chosen protocols meet the requirements of the network we use a constructive approach *i.e.,* a protocol is chosen only if all the features it requires are provided by the network or by another protocol. For this purpose we first need to model protocols so that the cost functions can be easily deduced when given the topology model.

3. ARCHITECTURE OVERVIEW

The architecture (Figure 1) consists of a set of protocol cost functions, an application model and the hardware specification. With these components the selection engine can generate the set of protocols that will be implemented in the protocol stack.

Application model.

The application model describes the behavior of the application at the node level. In order to enable the programmer to describe the application we use an API (Application Programming Interface) written in Java. From this description we can characterize the packet traffic model.

Hardware specification.

The hardware specification depicts the characteristics of the sensors used in the network. It gives the energy consumption of a node for the different operations it can achieve. For instance, it should specify the amount of energy used for sending one bit or one packet, when in idle and active modes, or when switching the radio on.

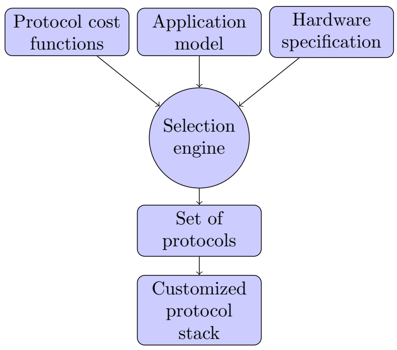

Figure 1: System Architecture

Protocol cost functions.

The protocol cost functions indicates how much energy is spent while using a protocol. It takes into account the topology control cost, the energy spent to send a unit of data (bit, byte, packet, etc... according to the unit used by the protocol). For instance the cost of a routing protocol which maintains route tables is composed of the costs of maintaining these tables and of sending messages.

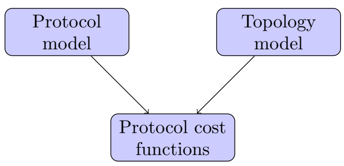

Figure 2: Protocol cost functions

Protocol cost functions are obtained thanks to the protocol model and the topology model as shown on Figure 2. The protocol model describes the protocol model using a specific description language. The topology model represents the layout of the network using density functions which gives, for example, the size of the one hop neighborhood (number of neighbors of a node).

Generation of the customized protocol stack.

The customized protocol stack is represented by a set of Java classes which has been selected by the engine. These classes implement the API given to the programmer. We use the Java-In-The-Small (JITS) framework [2] in order to generate the embedded system including communication stack. JITS is a framework which aims at allowing the easy deployment of standard Java applications on small devices (*e.g.*, smartcards or motes) with hardware constraints (memory, CPU, energy, etc.).

4. CONCLUSION AND FUTURE WORK

In this paper we described a possible architecture for a programming environment for WSN. This environment allows the programmer to only focus on the application behavior while having an energy-efficient networking stack. The protocols to be implemented in this stack are chosen according to their energy consumption cost. The API for the application model has already been designed. We are currently working on the protocol cost functions. We will also test the API and the general architecture on various test applications.

5. ACKNOWLEDGEMENTS

This work is supported by the WASP (Wirelessly Accessible Sensor Populations) and the SVP (SurVeiller et Prévenir) projects.

WASP [7] is an integrated project funded by the European Union. It aims at rectifying the imbalance between research at the application level and the node and the network level. The objectives are to develop an autonomous, intelligent and cost-efficient infrastructure which incorporates a WSN and encourages application driven optimisations.

The french project SVP [6] is proposing the study, the development and the experimentation of an integrated ambient architecture to make easier tasks such as conception, deployment and optimal operating of monitoring and prevention services for different kinds of dynamic networks

6. REFERENCES

[1] Th. Arampatzis, J. Lygeros, and S. Manesis. A survey of applications of wireless sensors and wireless sensor networks. In *Intelligent Control, 2005. Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation*, pages 719–724, 2005.

[2] Java In The Small (JITS). http://jits.gforge.inria.fr/.

[3] Raja Jurdak, Cristina Videira Lopes, and Pierre Baldiy. A framework for modeling sensor networks. In *Proceedings of the Workshop on Building Software for Pervasive Computing at OOPSLA'04*, 2004.

[4] Animesh Pathak and Viktor K. Prasann. Issues in designing a compilation framework for macroprogrammed networked sensor systems. In *InterSense '06: Proceedings of the first international conference on Integrated internet ad hoc and sensor networks*, page 7, New York, NY, USA, 2006. ACM Press.

[5] Ryo Sugihara and Rajesh K. Gupta. Programming models for sensor networks: A survey. *ACM Transactions on Sensor Networks (to appear)*, 2007.

[6] SVP project website. http://svp.irisa.fr/.

[7] WASP project. http://www.wasp-project.org/.