Poster Abstract: Prototyping a Software Factory for Wireless Sensor Networks

Tomasz Naumowicz
Freie Universität Berlin
Institute of Computer Science,
Takustr. 9, 14195 Berlin, Germany
naumowic@inf.fu-berlin.de

Benjamin Schröter
Microsoft Research
7 JJ Thomson Ave CB3 0FB
Cambridge, UK
v-besch@microsoft.com

Jochen Schiller
Freie Universität Berlin
Institute of Computer Science,
Takustr. 9, 14195 Berlin, Germany
schiller@inf.fu-berlin.de

Abstract

Wireless sensor networks (WSNs) are often advertised with high sensing accuracy, long lifetime, and easy deployment. However, they are still not widely used in environmental research due to of poor tool support and high complexity. A wider use of WSNs in field science would enable researchers to address scientific questions that are infeasible today.

To address this issue, we designed and prototyped a Software Factory for WSNs that hides the complexity of software development for embedded systems. It exposes a visual domain-specific modeling language and supports code generation for resource constrained devices. The proposed Software Factory simplifies the integration of domain experts into the development process, making WSNs more attractive as a tool for researchers from outside of the computer science field. This could lead to a wider adoption of WSNs in field sciences.

Categories and Subject Descriptors

D.1.7 [**Programming Techniques**]: Visual Programming; D.3.4 [**Programming Languages**]: Processors—*Code generation*

General Terms

Management, Design, Experimentation, Human Factors

Keywords

Wireless Sensor Networks, Software Factory, Domain-Specific Languages

1 Introduction

Research in the area of WSNs has previously focused on hardware design, self-organization, various routing algorithms, or energy saving patterns. This trend is already changing [2], but the available tools typically target experienced software developers rather than researchers from outside the computer science field.

In recent cooperation with domain experts during real world WSN deployments [4], we observed that our expertise as computer scientists in software development for embedded systems was required in every phase of the project such as planning, deploying, and managing of the WSNs. The research goals shifted often with changing field conditions and we had to update the deployed applications each time. High complexity of required changes prevented domain experts from modifying the application themselves. The dependency on us was a big disadvantage - it resulted not only in delays but also distracted researchers from the core goals of their work. This highlights the main barriers to the wide adoption of WSNs in real-world scenarios: high complexity, poor tool support, and continuous dependency on computer scientists.

We have designed and developed a Software Factory to address this problem. Software Factories are model-driven development environments that support automatic code generation. Our solution is based on a data-centric programming model where data flow models are used to describe how information should be processed. Domain experts use a simple visual Domain-Specific Language (DSL) to define data flows and, as a result, to specify the behavior of the WSN. Data flows are automatically transformed into native application code and compiled for the selected hardware platform, encapsulating the complex process of writing and compiling WSN programs. We refer to the proposed Software Factory as Flow.

Flow was prototyped for the resource constrained ScatterWeb WSN platform MSB430H [1] with 55kB flash memory and 5kB RAM. The prototype of Flow is available for download[1] and evaluation.

2 The Software Factory in Brief

Flow provides a visual editor for modeling of applications for WSNs, a set of DSLs, and a set of code generators. Flow focuses on the visual representation of data flows at a very high abstraction level (Figure 1). The modeled data flows are validated for syntactic correctness at design time and instant feedback is provided.

The Software Factory does not rely on a virtual machine and generates native code for the target platform. This approach significantly reduces the amount of work when adding support for new hardware platforms or new sensors. Native code generation also enables easy integration of code blocks which can be required to handle advanced tasks, e.g. domain specific mathematical computations.

Flow was built on top of Microsoft DSL Tools and leverages MS Visual Studio 2008 as the Integrated Development Environment.

SenSys'09, November 4–6, 2009, Berkeley, CA, USA.
ACM 978-1-60558-748-6

[1] http://cst.imp.fu-berlin.de/projects/flow

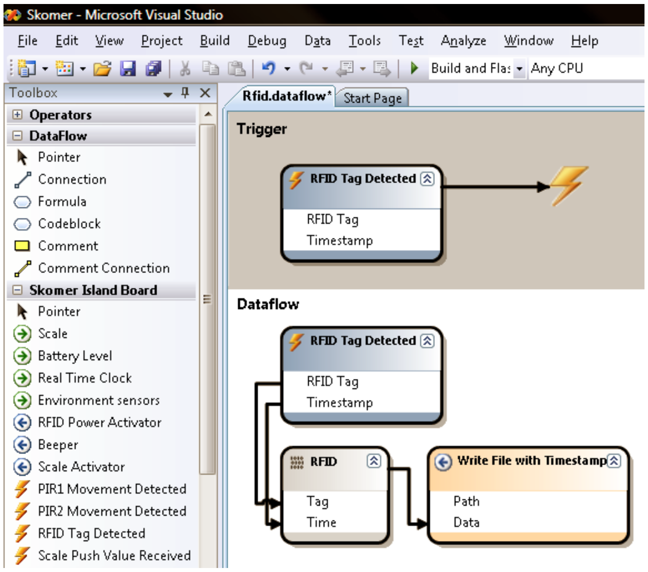

Figure 1. A sample data flow model.

2.1 Domain-Specific Languages

Flow uses three custom visual DSLs: the Hardware Description DSL, the Data Structures DSL and the dynamic Data Flow DSL.

The Hardware Description DSL is used to specify the underlying hardware platform. The specification of the platform and custom hardware extensions are created by a hardware engineer. The hardware description contains definitions of reusable software components (shapes) that represent real or virtual hardware. For example a temperature sensor or a FAT file system could be defined here. The shapes are automatically generated from annotated native firmware sources. The hardware description and the corresponding firmware sources are distributed as a hardware setup package.

The Data Structure DSL is used by domain experts to define project specific data records (data containers) and variables.

The Data Flow DSL is used by domain experts to define the data flows. The actual set of available components depends on the selected project specific hardware description and definitions of the data structures. Domain experts use the Data Flow DSL to combine and configure the available shapes and thus to describe the behavior of the nodes in the WSN.

3 Future Work

Flow needs to be evaluated by domain experts in real world scenarios. Flow will be deployed on the WSN installed on the Skomer Island [4]. Another upcoming evaluation is a deployment of a WSN with 40 sensor nodes at Wytham Woods near Oxford, UK. The WSN will be used to support the research on dispersal, environment and spatial heterogeneity of avian malaria in Great Tits. In both experiments researchers will use Flow to program their WSNs.

4 Related Work

The research community acknowledged the need of DSL support in WSNs. Several approaches were proposed that allow to model WSN behavior for simulation purposes, see e.g. [6]. Actual support for generation of executable code is still rare:

Mozumdar et al. prototyped a framework for modeling, simulation and automatic code generation of applications for sensor networks [3]. Their advanced solution targets professional software engineers.

Sadilek proposed a well-designed concept of a software factory for embedded systems and prototyped a DSL for describing of earthquake detection algorithms [5]. They use Scheme as an intermediate language which has to be compiled for the selected target platform. Currently their code generator supports embedded Linux platforms. Support for resource constrained devices is envisioned as a virtual machine that can be extended with native code.

5 Conclusions

The proposed solution has the potential to shorten the development cycles dramatically during WSNs deployments in field sciences, to reduce the dependency on hardware and software engineers, and to lead to a wider adoption of WSNs outside the computer science field.

6 Acknowledgments

This work has been partially sponsored and supported by the Computational Science Group of Microsoft Research.

7 References

[1] M. Baar, E. Köppe, A. Liers, and J. Schiller. Poster and abstract: The ScatterWeb MSB-430 platform for wireless sensor networks. In *SICS Contiki Hands-On Workshop*, page 3, Kista, Sweden, 03 2007.

[2] M. Ceriotti, L. Mottola, G. P. Picco, A. L. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon. Monitoring heritage buildings with wireless sensor networks: The Torre Aquila deployment. In *The 8th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, San Francisco, CA, US, 04 2009.

[3] M. M. R. Mozumdar, F. Gregoretti, L. Lavagno, L. Vanzago, and S. Olivieri. A framework for modeling, simulation and automatic code generation of sensor network application. In *SECON*, pages 515–522. IEEE, 2008.

[4] T. Naumowicz, R. Freeman, A. Heil, M. Calsyn, E. Hellmich, A. Brändle, T. Guilford, and J. Schiller. Autonomous monitoring of vulnerable habitats using a wireless sensor network. In *REALWSN '08: Proceedings of the workshop on Real-world wireless sensor networks*, pages 51–55, New York, NY, USA, 2008. ACM.

[5] D. A. Sadilek. Domain-specific languages for wireless sensor networks. In T. Kühne, W. Reisig, and F. Steimann, editors, *Modellierung*, volume 127 of *LNI*, pages 237–241. GI, 2008.

[6] S. Wielens, M. Galetzka, and P. Schneider. Design support for wireless sensor networks based on the IEEE 802.15.4 standard. In *PIMRC 2008: Proceedings of the IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–5. IEEE, 2008.