Towards a Principled and Evolvable Approach to Software Development for Future Wireless Sensor Networks

Michael R. Poppleton, Geoff V. Merrett
Electronics and Computer Science
University of Southampton, Southampton SO17 1BJ, UK
{*mrp,gvm*}*@ecs.soton.ac.uk*

***Abstract*—Due to the operational demands and requirements diversity in wireless sensor networks (WSNs), great improvements in the software engineering process are required. As WSNs increasingly become essential, even critical, components in systems-of-systems (SoSs), the case for verification in the development process is strong. In this position paper we present our vision for a principled formal software development and verification process for WSNs within SoSs.**

***Keywords*-complexity, formal, refinement, simulation, software engineering, wireless sensor network**

I. Introduction and Motivation

WSN deployments and applications have many dimensions of variability, for example heterogeneity of purpose and functionality, communication patterns, fault tolerance and safety, and performance requirements. These challenges - along with the well-known constraints of node software - lead to the fact that software development is currently a "code-and-fix process" [1] using APIs provided by simple WSN OSs and middleware (although, from our own experiences in low-power WSN design, these are often avoided, in favour of coding monolithic programs to obtain the desired effect). OS and middleware platforms have been proposed [2] but so far only exploited in isolated ways. In his position paper, Picco [1] stresses the need for collaboration between SE and WSN communities to develop principled SE development processes for WSNs. The current lack of progress in this area threatens to hinder further progress, and certainly the much-prophesised mass commercial exploitation, of WSNs. [1] identifies the need - with which we agree - for improved development practices and confidence in correctness and reliability, and advocates addressing these through engineering processes for middleware abstractions.

In this position paper we argue that a need for structured development, which encourages code reuse and ensures correctness and reliability, cannot be solved by middleware or programming abstractions alone. Instead, we propose a principled SE process driven by formal modeling and design with built-in simulation, test and verification support.

Compounding all the above is the vision of truly pervasive computing, in which individual WSNs are a part of a larger system-of-systems (SoS), interacting with other proprietary and heterogeneous systems [3]. Such a SoS may exhibit complex and emergent behaviours, and its components may need to adapt, reconfigure, or retask accordingly. Design, simulation, operation and maintenance of the SoS will rely on various infrastructural assumptions on its components; it might impose assumptions on the WSN such as latencies between sensing and actuation, bandwidth, reliability, or power consumption. Thus the gathering, modelling and verification of these assumptions represents an evolving requirements engineering process for the WSN through time.

II. Formal Software Development for WSNs with built-in Simulation, Test and Verification

Our vision for a principled software engineering process is one based on formal software technology; Picco [1] makes a succinct case for the increasing importance of "provably-correct behaviour" of WSNs, and thus the integration of Formal Methods technology in the SE process. Event-B [4] is a leading formal modelling language by J.-R. Abrial and its RODIN toolkit [5] provides formal modelling, animation, model-checking, verification, and proof. A rich set of tools serves as foundation for current FP7 project DEPLOY[1] which is scaling methodology and tooling for Event-B. In software development with Event-B, *refinement* is the central method by which initially small, abstract models of high-level requirements are elaborated through the addition of lower-level requirements during architectural and detailed design, down to code generation. Refinement removes implementation-freedom and adds data and algorithmic structure. Event-B allows us to formally encode requirements as correctness properties for models and their refinements, and to prove these properties formally.

Our visionary concept for this new software development process is depicted graphically in Figure 1. Initial requirements are drawn from the SoS (1), dictated by the WSN application being delivered, and formally modelled and refined at network level (2). This network modelling drives the automated production of conventional WSN network-level simulation models (3). This interaction is central to

[1]DEPLOY - Industrial deployment of system engineering methods providing high dependability and productivity: FP7 Project 214158 http://www.event-b.org

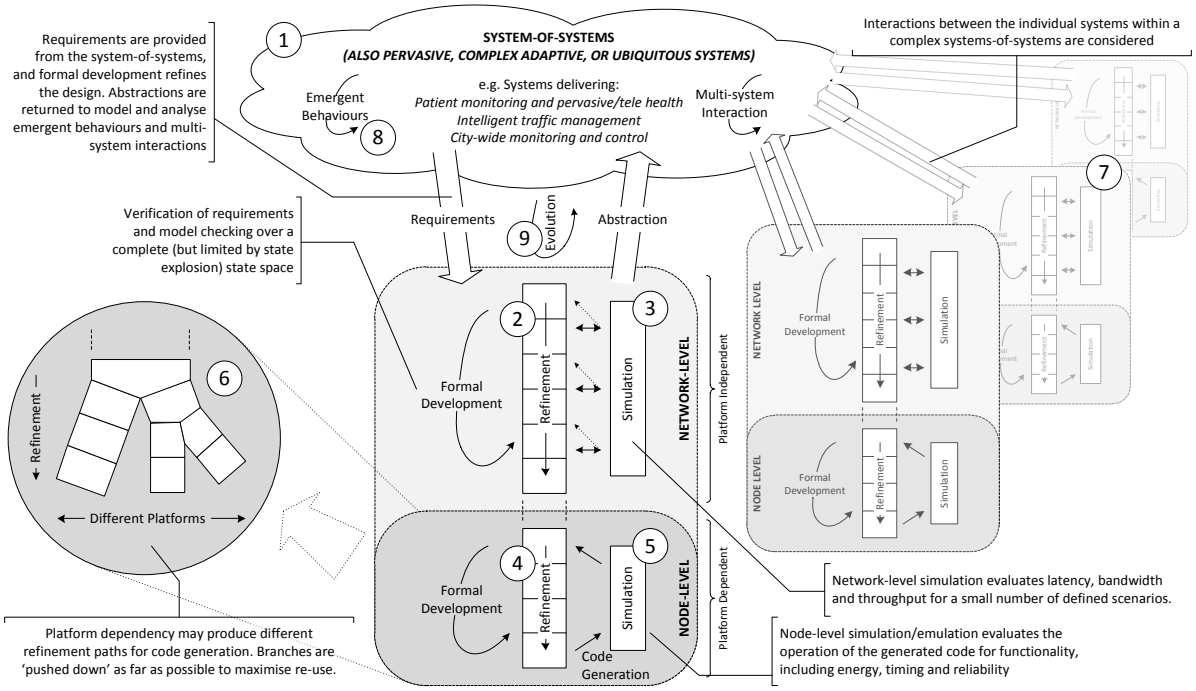

Figure 1. Our Vision for Formal Software Development for WSNs

any principled software engineering process; while functional properties of the WSN can be proved, nonfunctional, timing and performance properties in general cannot. Formal animation (manual symbolic execution of single test cases) and model-checking (exhaustive state-space evaluation via constraint-checking of required properties) can validate behaviour, and verify deadlock and simple temporal properties within the usual state explosion limits. A long-running WSN, with ranges of configuration data and sensed data, requires conventional simulation, i.e. efficient large-scale test scenarios to evaluate behaviour and performance. As formal verification and simulation confirm satisfaction of requirements, (2 - 3) will be an iterative cycle.

Similarly, the formal design output from the network phase, is input to an iterative cycle of formal design/refinement (4) and simulation (5) at the node level, where microcontroller code is generated from the end of the refinement chain and emulated. The network-level models impose requirements at the node level - on top of intrinsic node requirements such as energy-awareness, power efficiency - which are verified in this cycle.

Requirements to implement on different hardware or code-generate to different middleware platforms, will generate forks in the refinement tree (6), which will be "pushed-down" to maximise reuse. This will, effectively, represent product-line working for WSN.

Ultimately, our vision includes interaction through the SoS with other WSNs (7), existing or developed as above. Also, the complexity of the SoS may produce emergent behaviours (8) - perhaps discovered through SoS simulation - necessitating evolution of the high-level requirements (9) of the SoS.

III. Conclusion

In this position paper we propose a principled SE process for WSNs, driven by formal modeling and design with built-in simulation, test and verification support. This principled process combines best practice in formal, middleware and broader SE technology. Our vision is timely, as recognition of the need for efficient SE processes becomes widespread in the WSN community, and as WSNs are increasingly seen as components of SoSs of the future.

References

[1] G. Picco, "Software engineering and wireless sensor networks: Happy marriage or consensual divorce," in *FoSER 2010*, 2010.

[2] L. Mottola and G. Picco, "Programming wireless sensor networks: Fundamental concepts and state of the art," *ACM Computing Surveys*, vol. 43, no. 3, April 2011.

[3] J. A. Stankovic, "Wireless sensor networks," *IEEE Computer*, vol. 41, no. 10, pp. 92–95, 2008.

[4] J.-R. Abrial, *Modeling in Event-B - System and Software Engineering*. Cambridge University Press, 2010.

[5] J. R. Abrial, M. Butler, S. Hallerstede, and L. Voisin, "An open extensible tool environment for Event-B," in *ICFEM 2006*, ser. LNCS, Z. Liu and J. He, Eds., vol. 4260, Macau, 2006.