Design Methodologies for Embedded Systems: Where is the Super-Glue?

Fabrice Kordon
Université Pierre & Marie Curie, LIP6-CNRS UMR 7606,
4 place Jussieu, 75252 Paris Cedex 05,
France
Fabrice.Kordon@lip6.fr

1 Introduction

Embedded systems is an area that grows rapidly with new communication media such as smart-phones, house automation applications (that might finally come) and all other hidden systems we use everyday in cars, airplanes, etc. Moreover, in these last domains, there is a need for very safe development because such systems are often life-critical.

2 Building Embedded Systems

So, the community is working hard to deal with some major problems raised by such developments.

Numerous embedded systems do have to deal with time constraints. So, a lot of top world-researchers are working hard on this. How to handle and design systems with time like in [1, 5] ? How to check timing constrains like in [3, 2] ? How to model time like in [7] ?

One problem when such embedded systems are distributed is the underlying execution environnement that must be handled by middleware. So, people are building middleware or protocols that should increase safety of such systems like in [11, 9]. And now we start being able to consider middleware as Components off-the-shelf. Some of them are even formally verified for some points like in [4].

And we could scan as well numerous areas in embedded systems... All this work allow to build nice case studies that comes from realistic application such as the BART system [6].

3 So What's Wrong?

However, embedded systems stil cost a lot more than expected, probably due to the high number of so-called "non functional requirements"... actually, the ones that require the most important development effort. So, what's wrong?

In French, there is an expression that can be translated as follows: "be aware of the tree that hides the forest". In Embedded systems research, there are several trees hiding the forest that should be considered too: when we pay attention to "local" problems (that are also important ones) such as modeling, building, analyzing. We forget the forest behind these trees : the required super-glue gathering all these techniques together in an orchestrated way to help engineers to build their systems.

One could say: watch UML [8] ? Watch AADL [10] ? However, what do you get with these ? A huge standard-book with very detailed explanations telling you how to describe this or that. But no real information about how to use the notation in a comprehensive way to build systems... this is not yet methodologies than can be operated "on the ground", where people have to "fight with bugs" to make reliable embedded systems on time.

The recent model driven engineering approach could raise some hope in that direction. However, I feel that there are still some areas missing since model transformation and code generation, if they are of precious usage in such projects, do not solve all problems of embedded systems as stated in [12]. In particular, constraints such as resource usage or time are quite difficult to capture in early requirements and Another issue exists with education in computer engineering that should consider methodological aspects as well as the technical ones.

4 Conclusion

So, I feel this is important that researchers **also** put their attention to the definition of "good practices" in order to sketch some appropriate engineering rules. This is the super-glue that creates the global coherence of the "big picture" in embedded systems development.

Of course, the underlying techniques, as the ones outlined in section 2, should be defined (methodology to build

978-0-7695-3132-8/08 $25.00 © 2008 IEEE
DOI 10.1109/ISORC.2008.88

an house will not help if you do not even know how to elaborate concrete). However we should **also** pay more attention to methods that are the "hidden part" of development, as non functional requirements are for embedded systems : the part you forget and "consider later" (when it is too late).

References

[1] B. Bérard, P. Gastin, and A. Petit. Intersection of regular signal-event (timed) languages. In E. Asarin and P. Bouyer, editors, *Proceedings of the 4th International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS'06)*, volume 4202 of *Lecture Notes in Computer Science*, pages 52–66, Paris, France, Sept. 2006. Springer.

[2] B. Berthomieu and F. Vernadat. Time petri nets analysis with tina. In *Third International Conference on the Quantitative Evaluation of Systems (QEST 2006), 11-14 September 2006, Riverside, California, USA*, pages 123–124. IEEE Computer Society, 2006.

[3] W. Deng, M. Dwyer, J. Hatcliff, G. Jung, Robby, and G. Singh. Model-checking middleware-based event-driven real-time embedded software. In *Proceedings of the First International Symposium on Formal Methods for Components and Objects (FMCO 2002)*, 2003.

[4] J. Hugues, Y. Thierry-Mieg, F. Kordon, L. Pautet, S. Baarir, and T. Vergnaud. On the Formal Verification of Middleware Behavioral Properties. In *Proceedings of the 9th International Workshop on Formal Methods for Industrial Critical Systems (FMICS'04)*, Linz, Austria, Sept. 2004. TO BE PUBLISHED.

[5] H. Kopetz. Event-triggered versus time-triggered real-time systems. In A. I. Karshmer and J. Nehmer, editors, *Operating Systems of the 90s and Beyond, International Workshop, Dagstuhl Castle, Germany, July 8-12, 1991, Proceedings*, volume 563 of *Lecture Notes in Computer Science*, pages 87–101. Springer, 1991.

[6] F. Kordon and M. Lemoine, editors. *Formal Methods for Embedded Distributed Systems: How to Master the Complexity*. Kluwer Academic, 2004. ISBN:1-4020-7997-4.

[7] E. A. Lee and Y. Zhao. Reinventing Computing for Real Time. In F. Kordon and J. Sztipanovits, editors, *Reliable Systems on Unreliable Networked Platforms, 12th Monterey Workshop 2005, Laguna Beach, CA, USA, September 22-24, 2005. Revised Selected Papers*, volume 4322 of *LNCS*, pages 1–25. Springer Verlag, January 2007.

[8] OMG. Unified Modeling Language: Superstructure, v2.0. Technical report, Object Management group, 2005.

[9] B. Ravindran, E. Curley, J. S. Anderson, and E. D. Jensen. Assured-timeliness integrity protocols for distributable real-time threads with in dynamic distributed systems. In *Emerging Directions in Embedded and Ubiquitous Computing, EUC 2007 Workshops: TRUST, WSOC, NCUS, UUWSN, USN, ESO, and SECUBIQ, Taipei, Taiwan, December 17-20, 2007, Proceedings*, volume 4809 of *Lecture Notes in Computer Science*, pages 660–673. Springer, 2007.

[10] SAE. Architecture Analysis & Design Language (AS5506). available at `http://www.sae.org`, sep 2004.

[11] D. Schmidt and F. Buschmann. Patterns frameworks and middleware: Their synergistic relationships. In *Proceedings of the 25th International Conference on Software Engineering*, 2003.

[12] B. Selic. From model-driven development to model-driven engineering. In *19th Euromicro Conference on Real-Time Systems, ECRTS'07, 4-6 July 2007, Pisa, Italy, Proceedings*, page 3. IEEE Computer Society, 2007.