WDS'10 Proceedings of Contributed Papers, Part I, 76–81, 2010. ISBN 978-80-7378-139-2 © MATFYZPRESS

Component-Based Development for Real-time Embedded Devices: Current Obstacles and Next Steps Forward

T. Pop

Charles University, Faculty of Mathematics and Physics, Department of Distributed and Dependable Systems, Prague, Czech Republic.

Abstract. As the embedded control systems became a part of an every day equipment, the importance of methods and tools for cost efficient development rises enormously. These new technologies are still typically missing important features making system engineering markedly more efficient – some of them (as system simulations or debugging on the code level) are well known from non-component development, other (as debugging on the meta-model level) are representing completely new approach. One of promising solution lies in the component-based engineering. This article presents the vision of future extensions of the SOFA-HI component framework and related tools needed to overcome this significant obstacle in using the component-based technologies for production development.

Introduction

Real-time embedded devices became in the last years an important part of our lives. These small systems could be found in consumer electronic, automotive industry and telecommunication devices as well as in various kinds of control systems all around us.

As the complexity of these systems increases exponentially H. Fennel et al. [2006], the importance of more and more efficient methodologies and tools for cost efficient and short time-to-market engineering and re-engineering of embedded systems grows rapidly.

Traditional development approach to the development of embedded and real-time devices is based on monolithic software programming. With the growing of complexity, this fact becomes a significant obstacle. More over, programmers of these systems have to deal with specific hardware, various operating systems and limited resources and designed systems often have to be tested or verified for safety critical or long-time-running deployment

Mentioned issues are well addressed in a component-based system engineering (CBSE), which provides methods for behavior specification Allen [1997], systems verification and fragmentation of software complexity. In addition, this approach could be also easily adapted to provide techniques addressing heterogeneity of a target environment – hardware or specific operating system could be modeled easily by specific components.

Significant obstacle in using today's mature component-based technologies is the fact, that the they are typically either proprietary and closely associated with particular target domain or they are experimental and suffers from lack of crucial features needed by developers.

This article presents proposed future improvements and extensions of the SOFA-HI component framework, which is designed for building reliable software for embedded and real-time devices. This framework builds upon years of research of the mature and feature rich SOFA 2 component framework.

First, the basic introduction to component-based system development is given and typical problems, that have to be addressed in the domain of embedded real-time systems are discussed. In the following part, SOFA 2 component framework is presented and the level on which the previously declared requirement are met in existing component-based technologies (including SOFA HI) for this target domain is discussed. Finally, our planed extensions and improvements are summarized end we conclude in the last section.

Background

Component-Based Development Process

One of the biggest problems of today's software systems is a growing complexity. The maintainability of such systems becomes problematic, resource consuming and hence expensive.

This problem could be solved by dividing software into small independent software units called components. These units have well defined interfaces, their implementation is hidden and could be possibly changed without any observable impact to a rest of constructed system. Another significant advantage is an easy software reuse – components can be stored in a repository and systems can be built by assembling these (already developed, analyzed and tested) units with completely separate life cycle. These software units could be pre-compiled and delivered by the hardware manufacturer. Described process can lower required level of the programming skills, hide hardware details and simplify the development process.

Development for Real-time and Embedded Systems

Software development for embedded device differs in many aspects from development of enterprise systems. Programmers have to deal with specific hardware, various operating systems and limited resources.

More over, embedded devices often realize safety critical or long time running systems leads to requirements for advanced methods for system specification, testing, and verification.

The most important aspects of design and development process real-time and embedded systems are discussed in Buttazo [2005], Cooling [2000] ,Barr and Massa [2006] and development for real-time software using the component-based approach is discussed in Crnkovic and Larsson [2002] and they are summarized in the following list:

- Support for periodic and aperiodic tasks (R1a)
- Support for modeling of real-time attributes (R1b)
- Support for real-time task scheduling at run-time (R1c)
- Scheduling analysis and verification support (R2a)
- General property sets analysis and checking support (R2b)
- Support for application operating modes (R3a)
- Small or memory footprint, verification of memory image size (R3b)
- Platform independence and portability (R4)
- Support for distributed and inter-process communication (R5)
- Support for distributed real-time computation (R6)
- Mature development tools and development environment (R7a)
- Existence of development methodology or guidelines for development process (R7b)
- Debugger and simulation tools. (R7c)

SOFA-HI Component System

SOFA HI Prochazka et al. [2009] is an extension of the SOFA 2 component framework Bures et al. [2006] targeted at the high-integrity real-time embedded systems.

SOFA 2 is a mature component system, it supports hierarchically composed components and moreover, SOFA 2 describes and supports all phases of system development and deployment life-cycle.

Its component model is defined by its meta-model. SOFA 2 components specifies provided and required interfaces and also component behavior. Each component internally contains set of micro-components which realize control part of component. As a profile of SOFA 2 targeted at real-time embedded systems, SOFA HI imposes various restrictions on the component model in order to make it more predictable and lightweight. Most important differences to SOFA 2 are support for specification of extra-functional (e.g. timing) properties by its meta-model and restrictions of dynamic reconfiguration at run-time. SOFA HI run-time as well as SOFA HI primitive components are implemented in the C programming language. Development tools and infrastructure are build upon SOFA 2 tools.

Related Work

There are many systems that could be used for development of embedded and real-time devices from both, academical as well as from industrial sphere. Industrially used systems as Robocop H. [2005] or BlueArxKim et al. [2009] are in many cases proprietary and the documentation or development timeline is not available. Academical and experimental component frameworks (as Fractal Bruneton et al. or PECOS Nierstrasz et al. [2002] or MyCCM-HI Vergnaud et al. [2005]) are typically summarized in an article discussing also future work (i. e. what should be finished), but on the other hand, there is typically available no information describing long term future visions of progress and improvements. Some future goals are presented in few existing project proposals, as for example in the Fractal MIND project Mind Team or the Eclipse RTSCThe Eclipse Foundation [2010] project.

Covering Development Process Needs

In the previous section we have listed most important criteria, that are affecting applicability of any technology intended to be used for development of real-time and embedded devices. In this section, the level on which this needs are covered in existing component based systems are discussed with special emphasis on SOFA-HI.

Because the major part of existing systems is using some general purpose real-time operating system as underlied technology, support for periodic and aperiodic tasks (R1a), support for scheduling on run-time (R1c) and portability issues (R4) are covered automatically. All the systems targeted to the domain of real-time end embedded systems support in some way modeling of real-time features Petr Hošek [2010]. On the other hand, schedulability analysis (R2b) and re-usability of timing properties is in the world of component systems still open question Medina Pasaje et al. [2001]. For example PECOS Nierstrasz et al. [2002] proposed method using timing Petri nets for real-time properties verification, but the method was never implemented. SOFA-HI timing verification has not been implemented yet. Verification of other system properties is very general question. This could mean system behavior specification and verification (e.g. behavior protocols Frantisek Plasil [2001]) or general checking of the system over a set of properties implemented for example in PECOS Nierstrasz et al. [2002] by generating the Prolog fact sets during system construction and then by querying this database.

Similar to timing properties issue is the question of a small memory footprint and checking of its size during deployment phase (R3b). Even if this is not complicated task, it is very important and typically not supported.

Because embedded systems are very sensitive to performance issues and run-time reconfiguration mean too big overhead, the variability is not supported or it is restricted to operational modes, which are implemented for example in myCCM-HI Vergnaud et al. [2005] or BlueArX Kim et al. [2009] and they are planed to be implemented also in SOFA HI. In the same manner, feature variability is supported in SOFA HI, but only at design time. This means when feature is not needed, its code is not included in the produced executable binary file.

Support for real-time communication and real-time distributed computation is implemented in different systems on different levels. It presumes using of real-time middleware and hardware communication technologies and on a sufficient level is supported only in few systems as AUTOSAR H. Fennel et al. [2006], MyCCM-HI Vergnaud et al. [2005] or ROBOCOP H. [2005].

Development tools and methodologies (R7a, R7b) are important not only for the developers comfort but also for the system maintainability. Many systems support at least basic development tools and methodologies.

Embedded device software is hard to debug and test and more over, the number of rewrites of device memory containing software is limited. Thus, simulation and debuggers are important parts of a development platform. Simulation can be performed not only by simulating system behavior as it is in systems like Scade Esterel Technologies [2010] but also on the level function calls or interfaces (as shown in Figure 3), on the meta-model level or by simulation of device behavior in virtual environment. This could give to developers valuable imagination what is happening inside the constructed system. Another important tool is a debugger, by which one could mean integration with well known code level debugger and embedded device debugger (JTAG) or introducing completely new approaches like debuggers on meta-model component level (further described in the next section). Debuggers and simulators are typically still not integrated with the existing component based technologies.

Planed Extensions

System Verification

As was already discussed in previous sections, software for embedded devices is often used in control and safety critical systems and thus there is a strong need for verification against various aspects. SOFA HI aims at very general model of verification properties shown and described in Figure 1 including checking of following aspects:

Timing properties: A very important aspect of real-time software is schedulability analysis. Even the fact, that SOFA HI is using real-time operating system as underlied technology does not guarantee, that all the timing properties will be met. Actually, checking of real-time properties in distributed or multiprocessor systems is complicated even in case of monolithic systems, and methods for specifying timing properties to reusable units brings to methods new dimension of complexity. This is contrasting with the fact, that practically usable implementation of timing constraints checking is extremely

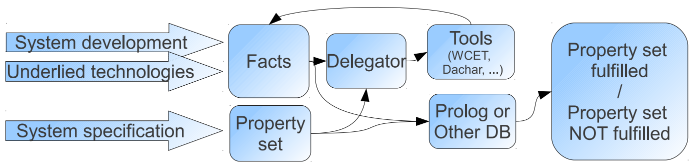


Figure 1. The idea is based on checking against set of properties produced by requirements on produced software and target platform. Some of properties could be checked just by querying database of facts generated during design and deployment phase, checking of others will be delegated to other tools.

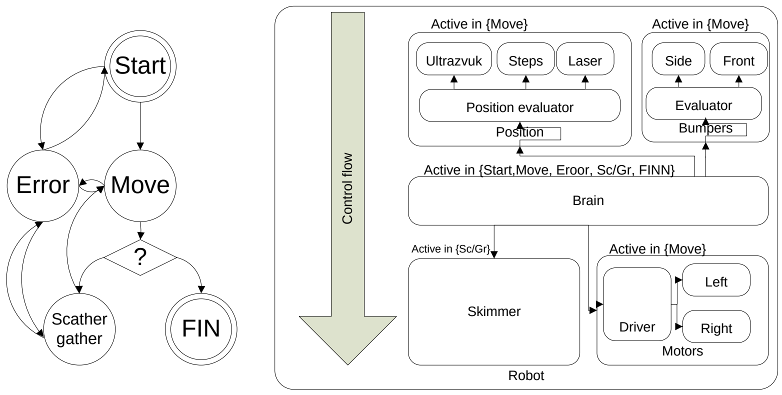


Figure 2. Example of application modes. The figure shows possible modes and their realization by component architecture in software system driving robot on *Eurobot* competition few years ago. At first, robot initialize itself in Start mode, than the robot is going to the given position (Move mode), where it gather balls (Scather/Gather mode). After this, robot goes to other position (Move mode) where it shots balls to a bin (Scather/Gather mode)

important for developers. We are planning to implement two methods, both covering local as well as distributed systems. First one is based on generating of timing Petri nets Felder et al. [1994] and than using existing tools as Romeo Gardey et al. [2005] or TINA Berthomieu and Vernadat [2006], and the second will produce the AADL Hugues et al. [2007] model and use the Chedar Singhoff et al. [2004] framework for further analysis.

Memory image footprint: Checking generated image footprint is very simple issue, but it is very important. SOFA HI is planing to check image size during the deployment phase, when both, generated image as well as target device memory size is known.

Formal behaviour checking will use behavior protocol Frantisek Plasil [2001] checkers developed for the SOFA 2 Bures et al. [2006] component framework.

Statical code analysis will be used for checking typical erroneous patterns and issues as stack overflow. We plane to use static code analysis tools for C as for example frama-C Claude March, INRIA [2010]

Mode Support

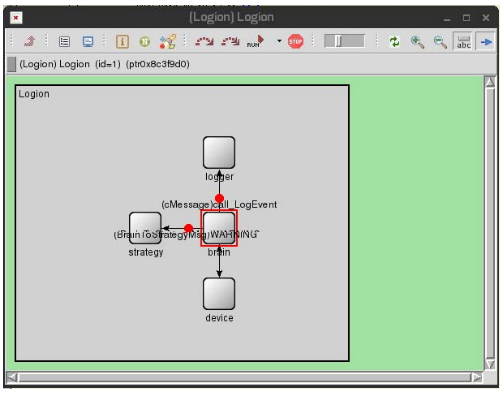
Real-time systems are typically passing several phases during application run time. General reconfigurations of application architecture are typically restricted due to overhead which is unacceptable on embedded devices and thus this phases are often represented by software architecture modes Hirsch et al. [2006]. In Figure 2 is shown an example from mobile robotics.

SOFA HI will implement direct support for application modes (run-time will take care about switching components, changing recursively hierarchical structure if needed and changing scheduling rules)

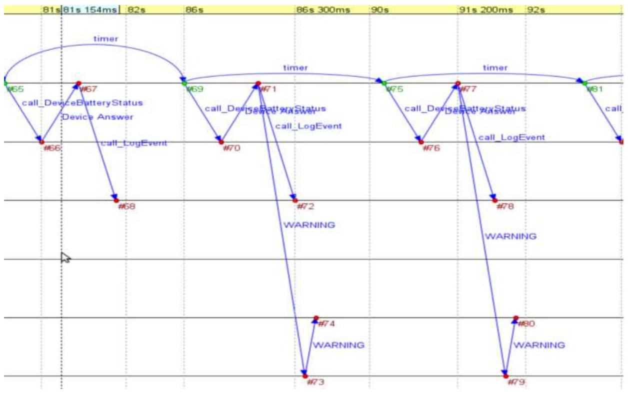
Simulation

Simulation of a system can significantly lower the time needed for system development, and we plan support not only deployment to a target device (as it is implemented now) but also deployment to simulation tools like Simulink or Scade.

More over, we are planing generating of the system models. Early experiments results are shown in Figure 3. This kind of simulation can help with debugging of the software on the design level. Quick and illustrative visualization can significantly lower time needed by the developer for orientation and thus lower development costs.



(a) Component interraction



(b) Control flow visualization in component system

Figure 3. First experiments with system model simulation and visualization produced by simulation tool Omnet++. In Figure (a) are simulated components and it's interaction during system execution, in Figure (b) is shown the control flow inside the simulated system.

Meta-model Debugger

Debugging of the system is important part of development process. Developers are inhabited to use debuggers, but today debuggers are operating on the level of the source code, but in component systems where the major part of executable code is generated and thus the existence of meta-model debugger, enabling observation of interfaces and system behavior on the component level would be great contribution because it would enable debugging on the system design level.

Distributed Real-time Communication

Distributed communication is very important in modern systems, where the computation is divided to many computational nodes. Good example is the automotive domain, for example Volvo CX90 is using approximately 40 computational nodes Tomáš et al. [2008]. SOFA-HI will use generated connectors Galik and Bures [2005] and real-time middleware (e.g. PolyORB Thomas Vergnaud and Kordon [2004] or RT-CORBA) as underlied technology.

Conclusion

In this paper we have summarized aspects that should be considered when designing a platform for the development of real-time and embedded devices. We have discussed how this aspects are met in current systems and we have introduced our vision of future improvements and extensions intended to replenish found inadequacies in the SOFA HI component system targeted to domain of embedded and real-time devices. We believe, that perspective of future development is important for potential users as well as that this early visions will be inspirative for developers of similar systems.

Acknowledgments. This work was partially supported by the Grant Agency of the Czech Republic projects 201/08/0266 and 201/09/H057.

References

Allen, R., *A Formal Approach to Software Architecture*, Ph.D. thesis, Carnegie Mellon, School of Computer Science, issued as CMU Technical Report CMU-CS-97-144., 1997.
Barr, M. and Massa, A., *Software Engineering for Real-Time Systems*, O'Reilly Media, 2006.
Berthomieu, B. and Vernadat, F., Time petri nets analysis with tina, *Quantitative Evaluation of Systems, International Conference on*, *0*, 123–124, 2006.
Bruneton, E., Coupaye, T., and Stefani, J.-B., The Fractal Component Model, http://fractal.ow2.org/specification/.
Bures, T., Hnetynka, P., and Plasil, F., SOFA 2.0: Balancing Advanced Features in a Hierarchical Component Model, in *Proceedings of SERA 2006*, pp. 40–48, Seattle, USA, 2006.
Buttazo, G. C., *Hard Real-time Computing Systems Predictable Scheduling Algorithms and Applications, 2nd edition*, Springer Verlag, 2005.

Claude March, INRIA, Frama-c, Web - electronic, URL `http://frama-c.com/`, 2010.

Cooling, J., *Software Engineering for Real-Time Systems*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000.

Crnkovic, I. and Larsson, M., *Building Reliable Component-based Software Systems*, Artech House, INC, Norwood, MA, 2002.

Esterel Technologies, Scade suite modeler :: Scade suite, Web - electronic, URL `http://www.esterel-technologies.com/products/scade-suite/modeler`, 2010.

Felder, M., Mandrioli, D., and Morzenti, A., Proving properties of real-time systems through logical specifications and petri net models, *IEEE Transactions on Software Engineering*, *20*, 127–141, 1994.

Frantisek Plasil, Stanislav Visnovsky, M. B., Behavior protocols, Technical Report, URL `d3s.mff.cuni.cz/publications/tr2000-7.pdf`, 2001.

Galik, O. and Bures, T., Generating connectors for heterogeneous deployment, in *SEM '05: Proceedings of the 5th international workshop on Software engineering and middleware*, pp. 54–61, ACM, NY, USA, 2005.

Gardey, G., Lime, D., Magnin, M., and (h. Roux, O., Romo: A tool for analyzing time petri nets, in *In Proc. CAV05, vol. 3576 of LNCS*, pp. 418–423, Springer, 2005.

H., M., A Robust Component Model For Consumer Electronic Products, in *Dynamic and Robust Streaming in and between Connected Consumer-Electronic Devices*, vol. 3, pp. 167–192, Springer Netherlands, 2005.

H. Fennel et al., Achievements and exploitation of the autosar development partnership, Technical Report, AUTOSAR GbR, URL `http://www.autosar.org/`, 2006.

Hirsch, D., Kramer, J., and Magee, J.and Uchitel, S., Modes for software architectures, in *Proceedings of EWSA 2006*, 2006.

Hugues, J., Zalila, B., Pautet, L., and Kordon, F., Rapid Prototyping of Distributed Real-Time Embedded Systems Using the AADL and Ocarina, in *IEEE International Workshop on Rapid System Prototyping*, 2007.

Kim, J. E., Rogalla, O., Kramer, S., and Hamann, A., Extracting, specifying and predicting software system properties in component based real-time embedded software development, in *Software Engineering - Companion Volume, 2009. ICSE-Companion 2009. 31st International Conference*, pp. 28–38, 2009.

Medina Pasaje, J. L., González Harbour, M., and Drake, J. M., Mast real-time view: A graphic uml tool for modeling object-oriented real-time systems, in *RTSS '01: Proceedings of the 22nd IEEE Real-Time Systems Symposium*, p. 245, IEEE Computer Society, Washington, DC, USA, 2001.

Mind Team, The MIND project, URL `http://www.minalogic.com/en/posters/Mind-eng-web.pdf`.

Nierstrasz, O., Arévalo, G., Ducasse, S., Wuyts, R., Black, A. P., Müller, P. O., Zeidler, C., Genssler, T., and Born, R., A component model for field devices, in *CD '02: Proceedings of the IFIP/ACM Working Conference on Component Deployment*, pp. 200–209, Springer-Verlag, London, UK, 2002.

Petr Hošek, Tomáš Pop, T. B. P. H. M. M., Comparison of component frameworks for real-time embedded systems, 2010.

Prochazka, M., Ward, R., Tuma, P., Hnetynka, P., and Adamek, J., A Component-Oriented Framework for Spacecraft On-Board Software, in *Proceedings of DASIA 2008, DAta Systems In Aerospace*, Palma de Mallorca, 2009.

Singhoff, F., Legrand, J., Nana, L., and Marcé, L., Cheddar: a flexible real time scheduling framework, *Ada Lett.*, *XXIV*, 1–8, 2004.

The Eclipse Foundation, Real-time software components, Technical Report, The Eclipse Foundation, URL `http://wiki.eclipse.org/DSDP/RTSC`, 2010.

Thomas Vergnaud, Jerome Hugues, L. P. and Kordon, F., Polyorb: a schizophrenic middleware to build versatile reliable distributed fapplications, in *Reliable Software Technologies - Ada-Europe 2004*, Springer Berlin / Heidelberg, Berlin / Heidelberg, 2004.

Tomáš, B., Carlson, J., Sentilles, S., and Vulgarakis, A., A component model family for vehicular embedded systems, in *ICSEA '08: Proceedings of the 2008 The Third International Conference on Software Engineering Advances*, pp. 437–444, IEEE Computer Society, Washington, DC, USA, 2008.

Vergnaud, T., Pautet, L., and Kordon, F., Using the aadl to describe distributed applications from middleware to software components, in *Ada-Europe*, pp. 67–78, 2005.