Tackling Automotive Challenges with an Integrated RE & Design Artifact Model

Birgit Penzenstadler

Technische Universität München
Fakultät für Informatik
Lehrstuhl für Software und Systems Engineering
Boltzmannstr. 3, 85748 Garching, Germany
{penzenst}@in.tum.de

Abstract. The automotive industry faces the need for large and complex embedded systems. The original equipment manufacturers (OEMs) assign the development of subsystems to suppliers. Therefore they are confronted with many challenges concerning specification, documentation, and integration until start of production (SOP).
A major part of these challenges can be tackled with continuous and integrated model-based requirements engineering (RE) and design. Following the analysis of the challenges, we present current work on a supporting artifact model for embedded systems development and especially focus on the design part in more detail.[1]

Key words: Architecture, Design, Documentation, Automotive, Embedded Systems

1 Introduction

A well-known fact is that the complexity of embedded systems is increasing, especially in the automotive domain, as for example the number of ECUs has increased from less than 10 in 1995 to more than 60 today [11]. Strong crosslinking between them makes designing an overall system architecture even more challenging.

The need for appropriate architectural specification and documentation is generally accepted [12]. In the automotive domain, this is complicated by the state of practice distributed development within an association of suppliers.

Contribution This paper discusses common challenges in software system development in the automotive domain and introduces work in progress on an integrated artifact model for RE and design that satisfies the special needs of the automotive domain. The focus lies on the design within the different abstraction layers.

Outline In Sec. 2 we briefly sketch the state of practice development process for automotive software and describe the arising challenges and related work.

[1] This work was partially funded by the German Federal Ministry of Education and Research (BMBF) in the framework of the REMsES project.

Sec. 3 explains the concepts and structure of the artifact model. Sec. 4 details the design part of the model and shows how to satisfy the mentioned specific needs of the automotive domain. Finally, Sec. 5 proposes possible next steps for the still unsolved challenges.

2 Automotive Software Development Process

The general automotive development process is organized according to the V-model [10]. During the conceptual phase (RE & design), the requirements are elicitated, and the logical architecture is designed. Then the technical system architecture, where the components are the electronic control units (ECUs), and networking (i.e., the layout of the wiring harness) are defined, and the software components are specified. The components are either developed in house or assigned to suppliers.

During the realization phase (implementation and integration), the components are implemented and tested, then during integration follow the integration tests, system tests and and acceptance tests. The strict deadline is SOP. This whole development cycle entails certain challenges:

Challenge of Architecture Specification The main aspects for the decomposition or modularization of the system during the conceptual phase are cost-optimization, exchangeability, reliability, and supplier structures. [4] state that "modeling of architectural designs (...) lives on whiteboards, in Microsoft PowerPoint slides, or in the developers' heads". Therefore, specification and documentation of an overall system's architecture is an important challenge.

Challenge of Distributed Development The highly distributed development in the automotive domain implies the need for thorough requirements specification with adequate interfaces, constraints, and context specifications. A number of constraints arises from different aspects of the system environment and all potentially relevant impact factors have to be specified in the tender documents for the suppliers.

Challenge of Integration The timespan for coordination and integration until start of production is one to two years, as the distributed development leads to late integration. The plan includes 5 levels of integration from basic physics to serial maturity. There is a strict process for error categorizing, tracing, and solving, with many participants and high costs. Especially for the safety-critical systems the aspect of liability is particularly important.

Related Work The challenges in the automotive domain have been discussed in depth by [5], while this paper only mentions the challenges that are faced by the artifact model.

Some of these challenges have already been adressed in different ways, as well in requirements engineering, e.g. [8], as in design, e.g. [7, 1]. In contrast, our artifact model covers different layers of abstraction.

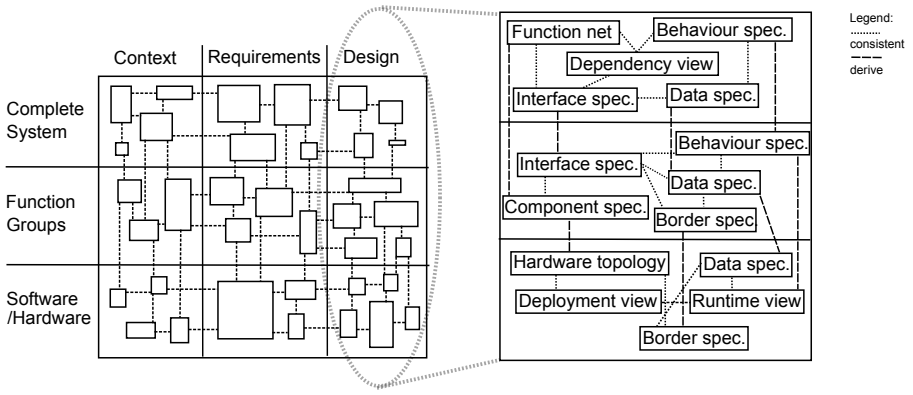

Fig. 1. Dimensions of the Artifact Model and Design Artifacts

3 Artifact Model for RE&Design of Embedded Systems

The REMsES project develops a guidebook with a building set of RE and design techniques tailored for the automotive domain. The key to consistency within the artifacts over the whole development process with distributed suppliers and demanding integration is seamless model-based requirements engineering and design.

The REMsES artifact model [2] is structured in three content categories and three abstraction layers (Fig 1).

Abstraction Layers The abstraction layers are (top-down) the complete system, the function groups, and the software/hardware (SW/HW), each layer specifying in more detail and adding certain views. On the complete system layer, the system is seen as black box that provides functionality to a user. The function groups layer represents the logical architecture of software subsystems. The SW/HW layer details the technical architecture with little abstraction from the implementation.

Content Categories The content categories are the context, requirements, and design. The context structures the interrelated conditions of the environment, inter alia with a system vision, business goals, stakeholders, and different types of constraints [13]. The requirements category encompasses documentation and refinement of system goals, use cases, scenarios, and functional requirements [9, 3].

The design category captures the early blueprints of the architecture on each layer. Due to space limitations we present only this last category in detail in Sec. 4, as for this workshop the design is the most interesting part of the model.

4 Integrated Design with the Artifact Model

The artifacts for the content category "design" were developed on the basis of [6]. These artifacts (see right side of Fig. 1) are views that are based on

one underlying system model per abstraction layer, and they serve as basis for deriving artifacts on lower layers by systematic refinement.

On the *complete system layer*, the design is captured in terms of user services or functions, and the views on those services are a function net, a dependency view, a behaviour view, and interface and data specifications. The function net gives an overview of the structure and interaction of the user functions. The dependency view provides a formal analysis of function interactions and system states, while the system behaviour is modelled with state automata. The data dictionary details on the representation and semantics of the input- and output-data of the functions and the interface captures the communication boundary.

On the *function group layer* the dominant design concept are logical components and their relations. Different modelling aspects are represented by structural component, interface, data and behaviour views, and an optional border specification. The structure is represented in terms of components that are connected through channels and ports. The interface, data and behaviour views refine the information of the complete system layer per component.

The border specification is a special new artifact that allows for the extraction of a subsystem – its intended use cases are either distributed development by suppliers or reuse within a new surrounding system specification. It encapsulates a short abstract of the functionality and usage of the subsystem and the particularly relevant information from the complete system context for the specific subsystem.

Finally, on the *software/hardware layer* there are hardware topology, runtime view, deployment view, and optional data and border specifications. The hardware topology describes the hardware units of the platform for the technical realization. The runtime view details on the cooperation of hardware clusters and application clusters (= software units) in terms of tasks, events, and buffers. The deployment view maps the application clusters to hardware units. The data dictionary can be refined in case of relevant hardware characteristis, e.g. sensor specifics. In case of reuse or development assignment on this abstraction layer, the border specification is completed with the additional information about technical constraints.

Facing the Challenges

Architecture specification challenges are met by the strong integration of RE and design in the proposed artifact model. The tailoring of the artifact model to the needs of the embedded systems domain becomes more obvious, the lower the regarded abstraction layer is, as the software/hardware layer contains all the specific details from sensor granularity to CAN message codes.

Distributed development is facilitated by the modular structure of the model. The complete system specification has to be maintained only once as the subsystem specifications are completely integrated. Then the tender documents can be extracted from the overall specification for the assignment to suppliers with the help of the border specification and a guiding process.

Integration is also supported by having one overall systems model, as a major impact on the integrational efforts originates from the quality and adequateness

of the system's architecture. The introduced design model facilitates integration in two ways: the artifacts can be used for simluation (verification) and support the process of the V-Model, as the abstraction layers match its steps.

5 Conclusion

In this paper, we have presented a number of challenges to current automotive software systems development and introduced the REMsES artifact model with special emphasis on the design part. The challenges of architecture specification and documentation, development by suppliers, and integration can effectively be met with this artifact model. There will be support for product lines that faces the problems of configurability, which is work in progress from our project partners.

We have evaluated the first two abstraction layers in a student project and we are currently evaluating the whole artifact model and the guidebook within a pilot scheme in industry.

Acknowledgement We would like to thank Doris Wild for helpful comments and feedback.

References

1. Balarin et al. A formal approach to system level design: metamodels and unified design environments. *MEMOCODE '05*, pages 155–163.
2. Bramsiepe et al. REMsES D2.2: Grobes Produktmodell inklusive Abstraktionsebenen. Project Deliverable, 2007.
3. Bramsiepe et al. Ableitung von Systemfunktionen aus Zielen und Szenarien. *Softwaretechnik-Trends*, 2008.
4. A. Brown and J. McDermid. The art and science of software architecture. In *ECSA Conference Proceedings*, 2007.
5. M. Broy. Challenges in automotive software engineering. In *ICSE Conference Proceedings*, pages 33–42. ACM, 2006.
6. Broy et al. Umfassendes Architekturmodell für das Engineering eingebetteter software-intensiver Systeme. Technical report, Technical University of Munich, 2008.
7. Lonn et al. FAR EAST: Modeling an automotive software architecture using the EAST ADL. *IEE Seminar Digests*, 2004.
8. Paech et al. An Experience-Based Approach for Integrating Architecture and Requirements Engineering. In *STRAW Proceedings '03*.
9. K. Pohl and E. Sikora. COSMOD-RE: Supporting the Co-Design of Requirements and Architectural Artifacts. In *RE Conference Proceedings*, pages 258–261, 2007.
10. A. Rausch and M. Broy. *Das V-Modell XT*. Springer, 2008.
11. K. Steinhauser and G. Bauer. Hitech im Antriebsstrang: Vernetzte Funktionen und Entwicklung. *Hanser Automotive*, 10:28–32, 2007.
12. J. Tyree and A. Akerman. Architecture decisions: demystifying architecture. *Software, IEEE*, 22(2):19–27, March-April 2005.
13. T. Weyer and K. Pohl. Eine Referenzstrukturierung zur modellbasierten Kontextanalyse im Requirements Engineering softwareintensiver eingebetteter Systeme. In *Modellierung Proceedings*, pages 181–197, 2008.