SPES 2020 Deliverable D1.1.A-3

Literature Survey
Identification of Relevant Literature for SPES 2020

Author: Martin Feilkas
        Alexander Harhurin
        Judith Hartmann
        Markus Hermannsdörfer
        Florian Hölzl
        Daniel Ratiu
        Wolfgang Schwitzer

Version: 1.0
Date: 31.05.2009
Status: Released
## Contents

1 Introduction 3

2 Relevant Literature 3
   2.1 Seamless Model-based Development: from Isolated Tools to Integrated Model Engineering Environments 3
   2.2 Challenges in Automotive Software Engineering 3
   2.3 Model-driven architecture-centric engineering of (embedded) software intensive systems: modeling theories and architectural milestones 3
   2.4 The ’Grand Challenge’ in Informatics: Engineering Software-Intensive Systems 4
   2.5 Umfassendes Architekturmodell für das Engineering eingebetteter Software-intensiver Systeme 4
   2.6 An Architecture-Centric Approach towards the Construction of Dependable Automotive Software 4
   2.7 A formal model of services 4
   2.8 Modulare hierarchische Modellierung als Grundlage der Software- und Systementwicklung 5
   2.9 COLA - The COmponent LAnguage 5
   2.10 Specification and Development of Interactive Systems 5
   2.11 Case tools for embedded systems 6

References 6
1 Introduction

This paper presents a survey of the literature relevant to the project “Software Plattform Embedded Systems 2020” (SPES). It focuses on literature written at the chair on Software & Systems Engineering at the Technische Universität München.

2 Relevant Literature

2.1 Seamless Model-based Development: from Isolated Tools to Integrated Model Engineering Environments

In [BFH+09] the authors envision a seamless development of models throughout the development process. The authors state that in the current practice model-based development is done in isolated tools using models are not built on a common semantic fundament. Furthermore the paper presents a vision what future modeling languages and techniques should look like. The authors call for a deep integration of the model-based development tools based in a common model repository. The different models are regarded as views onto a (potentially huge) comprehensive meta-model. To enable a very controlled execution of the development process, the workflows that are defined by the development process applied in a project should also be integrated into such a tooling environment. A mechanism to compose modeling languages is proposed to enable a domain appropriate modeling by integrating domain-specific and general-purpose modeling languages based on a building blocks principle. To implement such an approach the architecture of a common tooling platform is described. Additionally, technical as well as political barriers to the development of a common tooling platform are presented.

2.2 Challenges in Automotive Software Engineering

In [Bro06a] the author presents an overview of current and future challenges of automotive software engineering. The deficiencies of a traditionally bottom up oriented automotive software development process are pointed out. The author gives motivation to employ comprehensive model-based software development in general and specifically in the automotive domain. The characteristics of the automotive domain are described with respect to the specific demands of software production. The necessity of a seamless model-driven development process is emphasized and an appropriate comprehensive architecture that covers all relevant levels of the system under design is introduced. The potentials of the presented approach are pointed out in contrast to possible weaknesses that modeling in the automotive sector still shows today.

2.3 Model-driven architecture-centric engineering of (embedded) software intensive systems: modeling theories and architectural milestones

[Bro07] presents a motivational overview of the Focus-like theory, the architecture, and the model based development process. This paper is quite simple and understandable by the industry.
2.4 The 'Grand Challenge' in Informatics: Engineering Software-Intensive Systems

[Bro06b] is an overview paper that presents the motivation for model-based development and its influence on more areas of software engineering. One of the key statements is that 'Developing a methodology for specifying and verifying software-intensive systems poses a Grand Challenge that a broad stream of research must address'. Furthermore the author states that architecture plays a dominant role and 'The architecture provides the blueprint for the development process and for integration'. The author concludes that while 'Modeling is the heart of engineering, yet the theory of modeling for software-intensive systems remains incomplete'.

2.5 Umfassendes Architekturmodell für das Engineering eingebetteter Software-intensiver Systeme

[BFG+08] introduces a framework of abstraction layers for the specification of software intense, embedded systems. The basic idea is to specify a system on consecutive abstraction layers, each one giving a more detailed model of the system, where the highest layer reflects a very abstract, informal description of a system, while the lowest one represents a concrete deployable implementation. The introduced framework provides the necessary basis for an systematic software engineering process and future tool support. The paper motivates the general use of different abstraction layers and includes for each layer a motivation and a short description of its basic modeling constructs.

Though describing the layered framework only from a high-level perspective, the paper represents a good starting basis for the definition of a system of abstraction layers in SPES (AP 1.2).

An english version of the paper is in preparation.

2.6 An Architecture-Centric Approach towards the Construction of Dependable Automotive Software

[WFH+06] contains an introduction to the automotive-specific modeling language CAR-DL (Combined Architecture Description Language) developed within the project “mobilsoft”. The CAR-DL is based on a system of automotive-specific abstraction levels. This system as well as, for each abstraction level, the corresponding modeling constructs of the CAR-DL and their semantics are described in the paper.

The framework described in [BFG+08] was heavily inspired by this work. The paper is relevant for AP 1.2.

2.7 A formal model of services

In [BKM07], based on the FOCUS theory, a theory and formal model of services is introduced. In contrast to FOCUS components, which define total behavior, services allow for partial behavioral specifications. The paper gives a theoretical foundation of the notion of services and
shows how services can be structured and how software architectures can be composed of services and components.

The paper might be interesting for AP 1.1.

2.8 Modulare hierarchische Modellierung als Grundlage der Software- und Systementwicklung

[BR07] contains the motivation and introduction of the formal foundations a FOCUS-like modelling theory.

The paper is relevant for AP 1.1.

2.9 COLA - The COmponent LAnguage

The project Base.XT developed an integrated modeling language which covers all abstraction levels identified earlier: the feature architecture, the logical architecture, and the technical architecture. The modeling theory underlying the modeling language is based on perfect synchrony (also known as weak causality), i.e. computation and communication take no time [KTB+07].

The feature architecture structures the features of a system in a hierarchy, and makes the dependencies between them explicit (feature interaction) [Rit08]. The transition from the feature architecture to the logical architecture is far from clear, although it is currently assumed that it is a combination of refactoring and refinement.

The logical architecture decomposes the system according to control flow (automata) and according to data flow (networks) into modular components that can be realized independently of each other [KTB+07].

The cluster architecture clusters the logical architecture into distributable entities (clustering), and maps them onto a hardware architecture (allocation). Both clustering and allocation can be performed without any user interaction. Finally, code can be generated from the technical architecture which can be deployed on real hardware [HTB08].

Unfortunately, the integrated modeling language is not fully integrated yet. However, the experiences gained through Base.XT might be interesting for AP 1.2.

2.10 Specification and Development of Interactive Systems

This book [BS01] provides a mathematical and logical foundation for the specification and development of interactive systems based on a model that describes systems in terms of their input- and output-behavior. Based on the model, the authors build a basic method, called FOCUS, that enables interactive systems to be described by characterizing their histories of message interaction. The book progresses from an introduction and guided tour of FOCUS through streams, specifications and their properties, and behavioral, interface, and conditional refinements.
2.11 Case tools for embedded systems

This case study [SHH+03] shows what modeling concepts are reasonable and can be useful in the domain of embedded systems. It also shows that general purpose object-oriented modeling is not desirable in this domain. The authors state that suitable tools have to address issues like describing structures more abstractly than on the level of class/object diagrams and the introduction of communication mechanisms more suitable than method or procedure. Furthermore, it is shown what models and description techniques are commonly accepted as essential, giving a snapshot of the CASE-based specification of embedded software. Additionally, the authors present what kind of tool support is available for the development of such specifications and what the resulting development process looks like with a focus on the design phase by comparing eight tools from different domains. A sketch of what should be expected from future tools for embedded software is given.

References


