First Experiment on Modeling Safety LifeCycle Process in Railway Systems

Brahim Hamid, IRIT, University of Toulouse, France
Yulin (Huaxi) Zhang, IRIT, University of Toulouse, France
Jacob Geisel, IRIT, University of Toulouse, France
David Gonzalez, IKERLAN-IK4, Spain

ABSTRACT

The conception and design of Resource Constrained Embedded Systems (RCES) is an inherently complex endeavor. Non-functional requirements from security and dependability are exacerbated this complexity. Model-Driven Engineering (MDE) is a promising approach for the design of trusted systems, as it bridges the gap between design issues and implementation concerns. The purpose of process models is to document and communicate processes, as well as reuse them. Thus, processes can be better taught and executed. However, most useful metamodels are activity-oriented, and the required concepts of safety lifecycle, such as validation, cannot be easily modeled. In this paper, the authors propose a safety-oriented process metamodel that extends existing framework to support all safety control requirements. A new safety lifecycle development processes technique has been built to ease its use in a building process of system/software applications with safety support. As a proof of concept, the feasibility of the approach has been evaluated with an example. The example is an engineering process for building industry control systems with safety requirements for software and hardware resources. A prototype implementation of the approach is provided and applied to the example of industry control systems in the railway domain.

Keywords: Development Process, Embedded Systems, Modeling, Process Metamodel, Safety Lifecycle

INTRODUCTION

Motivation and Background

Embedded systems can be defined as information processing systems integrating hardware and software embedded into enclosing products to fulfill a specific function such as mobile phone or fabrication equipment, lift control, defibrillator, etc. These systems come with a large number of common characteristics, including real-time and temperature constraints, dependability as well as efficiency requirements (Zurawski, 2007; Henzinger, 2006). Specifically, resource constrained embedded systems (RCES) refer to systems which have memory and/or com-
putational processing power constraints. They can be found in many application sectors such as automotive, aerospace, and home control (Hamid, Desnos, Grepet, & Jouvray, 2010). They are in many types of devices, like sensors, automotive electronic control units, intelligent switches, and home appliances such as washing machines and smart-meters. In addition, they have different form factors (e.g., standalone systems, peripheral subsystems, and main computing systems). Computing resources of RCES, e.g., memory, tasks, and buffers, are generally statically determined. The generation of RCES therefore involves specific software building processes. These processes are often error-prone because they are not fully automated, even if some level of automatic code generation or even model driven engineering support is applied. Furthermore, many RCES also have assurance requirements, ranging from very strong levels involving certification (e.g., DO178 and IEC-61508 for safety-relevant embedded systems development) to lighter levels based on industry practices.

The development of these systems by traditional processes is no longer applicable and the need for new methods is necessary to design and provide software for resources constrained embedded systems (RCES) correct by construction. We seek a mechanism which allows a safe, easier and fast RCES development process. To cope with the growing complexity of embedded system design, several development approaches have been proposed. The most popular are those using models as the main artifacts to be constructed and maintained. In these processes, software development consists of model transformations.

Recent times have seen a paradigm shift in terms of design through the combination of multiple software engineering paradigms together, namely Model Driven Engineering and Software Product Line Engineering. Such a paradigm shift is changing the way systems are developed nowadays, reducing development time significantly. Embedded systems are a case in which a range of products for assorted domains such as energy, transportation, automotive, and so on are conceived as a family. However, most of the work so far has been focused on functional parts. MDE supports the designer to specify in a separate way non-functional requirements such as security and/or dependability issues at a higher level of abstraction. MDE has the potential to greatly ease daily activities of S&D experts. Previous work has shown the potential of Safety life cycle modeling solutions. They are created by experts, then validated, and finally stored in a repository that makes them available.

Development process cycles, including analysis and design phases, are mainly iterative, resulting in different refinement levels of models. Several model-based architectures (e.g., MDA standard) exist and address these issues in specific contexts. Unfortunately, in most of these cases, the links between processed and refined models are still solved in an ad hoc manner, without adequate support. For instance, references between two models are not automated (i.e., checking non-existing and non-referenced elements, managing new elements). The required solutions involve generally more reliable processes, which essentially ensure consistency and traceability of the produced models. We are still waiting for widely-applicable technologies that solve these issues in RCES environment. Modeling software and system process is fundamental in order to improve the quality of the application. The main goal of these processes is to provide organizations with the means to define a conceptual framework. For this reason, several tentatives (i.e., including those developed by the OMG, an organization which normalizes the UML language) have been proposed to model software process. For instance, the SPEM specification is used for describing a concrete software development process or a family of related software development processes. It’s conformed to the OMG’s MOF metametamodel and it defines a UML profile.

Meta-process modeling supports the effort of creating flexible process models. The purpose of process models is to document and communicate processes and to enhance the reuse