"Model-Driven Engineering: authoring and exploiting models"

9th MODPROD Workshop on Model-Based Product Development, February 3-4, 2015

Sébastien Gérard (sebastien.gerard@cea.fr)

CEA Saclay Nano-INNOV, Institut CARNOT CEA LIST, DILS/Laboratoire d'Ingénierie dirigée par les modèles pour les Systèmes Embarqués (LISE)

Abstract:

Overcome the current limits of the systems development approaches in their ability to manage the essential complexity requires new means, more powerful and more efficient. Abstraction is the most used universal principle to tackle this kind of issue: through the simplification it brings, it eases, even fosters, reasoning and hence enables to manage complexity of systems under design. In this area, models happen to be the undisputed champions. Indeed, when abstraction lefts the human mind to become tangible, it becomes a model. UML - "Unified Modeling Language" standardized by the OMG in January 1997 - is certainly the modeling language the most widespread in the world: it is taught in most schools and universities around the world, many tools exist, and a considerable amount of information is freely available on the internet through articles, tutorials and application examples. However, because of its generic character and its software connotation, its adoption may be difficult, deemed sometimes inadequate, insufficient or too complicated. Domain-specific modeling language are then preferred, that is to say languages specifically built to fit a particular application area or a specific concern.

The first part of this presentation will provide insight on how to deal with this paradox, standard versus specific. In particular, we will show how to benefit from the advantages of the UML, a standard modeling language internationally widespread for over 15 years while retaining the opportunity to profit from special advantages and properties of domain-specific modeling languages.

Abstraction and thus thereby modeling are not new paradigms. We practice them for a long time in all areas of "classical" engineering such as mechanics or thermodynamics. What is new with those approaches defined to model-driven or even model-based, and whose abstraction is the first principle, is the place and role of models in the development process. They are now considered as first-citizens and then put in the center of concerns. With a concrete description adapted to its problem, each party involved in the development of a system can reason and thus more effectively perform its specific tasks (for example, to describe the functional architecture of a system or a safety analysis). However, even if models are abstract, their complexity may remain high. Human cognitive abilities can then be insufficient to mentally use information provided by the model and thus be able to reason. One solution to this problem is to

raise the abstraction level that is to say to “abstract the abstraction” in order to come up with something humanly manageable. But it is not always possible. Indeed, to abstract consists in omitting details, and some details are sometimes significant and therefore should not be hidden. An alternative to this method is based on the second principle of model-driven engineering: automation. This means assisting the stakeholders concerned by a model to exploit its content. This can be done in different ways, either by generating documentation, code or else by "automating" the implementation of analytical technique (e.g., performance and schedulability analysis). Thanks to model transformations, it is possible to attend the cognitive activity and thus push the limits of human management complexity.
After having discussed ways to create models, the second part of this presentation will focus on the exploitation means via computer-aided design. Indeed, the use of models that can be done through model transformations allows either to automate simple, repetitive tasks or to attend the implementation of analysis techniques sometimes sophisticated and hence difficult to access. Exploiting models with a computer requires the ability to automatically understand the semantics of a model, which means to formalize it. This section will therefore also cover the different initiatives underway at the OMG ("Object Management Group") to formalize its work (e.g., fUML and PSCS respectively "Foundational UML" and "Precise Semantics of Composite Structure”).

Short bio:

Sébastien Gérard is leading the LISE laboratory (Laboratory of Model Driven Engineering for Embedded Systems) at CEA LIST (http://www-list.cea.fr/en). Working on research issues related to complex and critical system and software design for more than 15 years, his research interests include correct-by-construction specification and design of complex systems, model-based engineering of RT/E systems and modeling language engineering. He is the CEA representative at OMG for more than 15 years. In particular, he is the chair of the MARTE standardization task force. He is also leading the open-source project, Papyrus (www.eclipse.org/papyrus), the UML modeling tools of Eclipse. In 1995, he has a diploma in mechanics and aeronautics from the ENSMA high-school, in 2000 he obtained a PhD diploma in Computer Science from the Evry University and in 2013 he got his “habilitation à diriger des recherches” diploma in the domain of computer science from the Orsay university.