Software Evolution in Model-Driven Product Line Engineering

by Silvia Abrahão, Javier González-Huerta, Emilio Insfran and Isidro Ramos

New requirements and technology changes lead to continuous changes of the assets comprising a software product line. Since the product line represents a large number of potential products (or already deployed products) in a given domain, managing these changes becomes a key issue when dealing with evolution. We present a framework to support the development and evolution of high-quality software product lines. The framework is based on several interrelated models or system views (eg, functionality, variability, quality) and a production plan defined by model transformations that generate a software system that meets both functional and quality requirements. We used our framework to develop a software system for the automotive domain.

Software Product Lines (SPLs) are families of products that share common functionality but also have variations tailored for different customer needs. Many of the benefits expected from SPLs are based on the assumption that the additional investment in setting up a product line pays off later when products are created. However, product line assets have to evolve continuously in order to keep the economic benefits of a product line at an optimal level. Managing this evolution therefore becomes a key issue when maintaining a product line.

In the MULTIPLE project, we defined a framework to support the development and evolution of high-quality SPLs. This framework is based on the definition of a multimodel that represents the different system views and the relationships among them. The variability management involves the manipulation of features, represented as cardinality-based feature models, and the support of such variability in the so-called product line core assets. Specifications of the system variability, functionality, and quality can be dealt with models that are independent from each other but where their inter-consistency is assured by means of the relationships defined in this multimodel.

The SPL production plan is parameterized by means of the multimodel which specifies the corresponding model transformations that are needed to obtain a specific product with the desired features, functions, and quality.

Figure 1 shows the multimodel playing a pivotal role in the SPL production plan. In the domain engineering phase it is used to express the impact and constraints among features, functional components and quality attributes, describing in this way, the extension of the product line. In the application engineering phase, it guides the product configuration, allowing the selection of features and functional components that meets the quality attributes selected by the application engineer. This approach leads us to re-consider the problems related to the intra (eg internal consistency of the feature model) and inter-model consistency (eg correspondences among elements of the feature and quality models) in a broader and realistic context.

Model-based evolution of software systems implies the evolution by using models, eg applying model-driven techniques to support product evolution, or the evolution of models, ie the evolution of the models/metamodels that describe the product. In our framework these evolutions are done by means of reification of software artifacts at the metalevel. When the evolution affects the types then the metamodels need to be reified and evolved. A reflection process translates to the source level the evolved artifact. At a model level, the process is similar. The models are reified at the metalevel, evolved, and then reflected. In the MOF hierarchy, the evolution behaves in a similar way in each level and the consequences over the instances of the evolved artifact can be computed automatically.

Figure 2 shows the main steps of a production plan to develop a software system by applying model transformations for the automotive domain. The quality attributes that the product must fulfill are used as the decision factor when alternative transformations appear. In this way, the application engineer can experiment with different quality-driven model transformations (which represent design alternatives) and choose the one that better satisfies the functional and quality requirements.

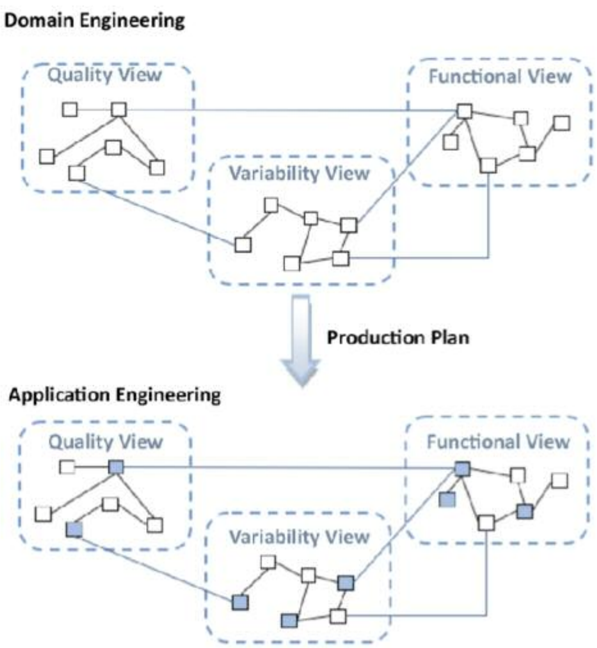

Figure 1: The Multimodel in the Software Product Line development process

The output of the process is the product represented by means of its architectural models. These architectural models can be completed in successive model transformation processes to generate the fully functional version of the product. In addition, in each step of the process, the quality model can be used to determine whether the artifacts obtained (eg product architecture) meet the expected quality levels by automatically applying metrics defined on the quality model. This assessment provides evidence about the correctness of the individual models as well as the multimodel relationships.

Finally, the multimodel helps to establish the traceability and to ensure the consistency among the system views. Furthermore, the model transformations are guided by the structural constraints established by the multimodel and the quality view contributes to the quality assurance and improvement of the product line artifacts. Currently, we are building a tool to give support to this approach.

Links:
ISSI Research Group (Universitat Politècnica de València)
http://issi.dsic.upv.es/projects

FMCL: Feature Modeling Constraint Language
http://issi.dsic.upv.es/~agomez/feature-modeling

Please contact:
Silvia Abrahão
Universitat Politècnica de València, Spain
Tel: + 34 963877000
E-mail: sabrahao@dsic.upv.es

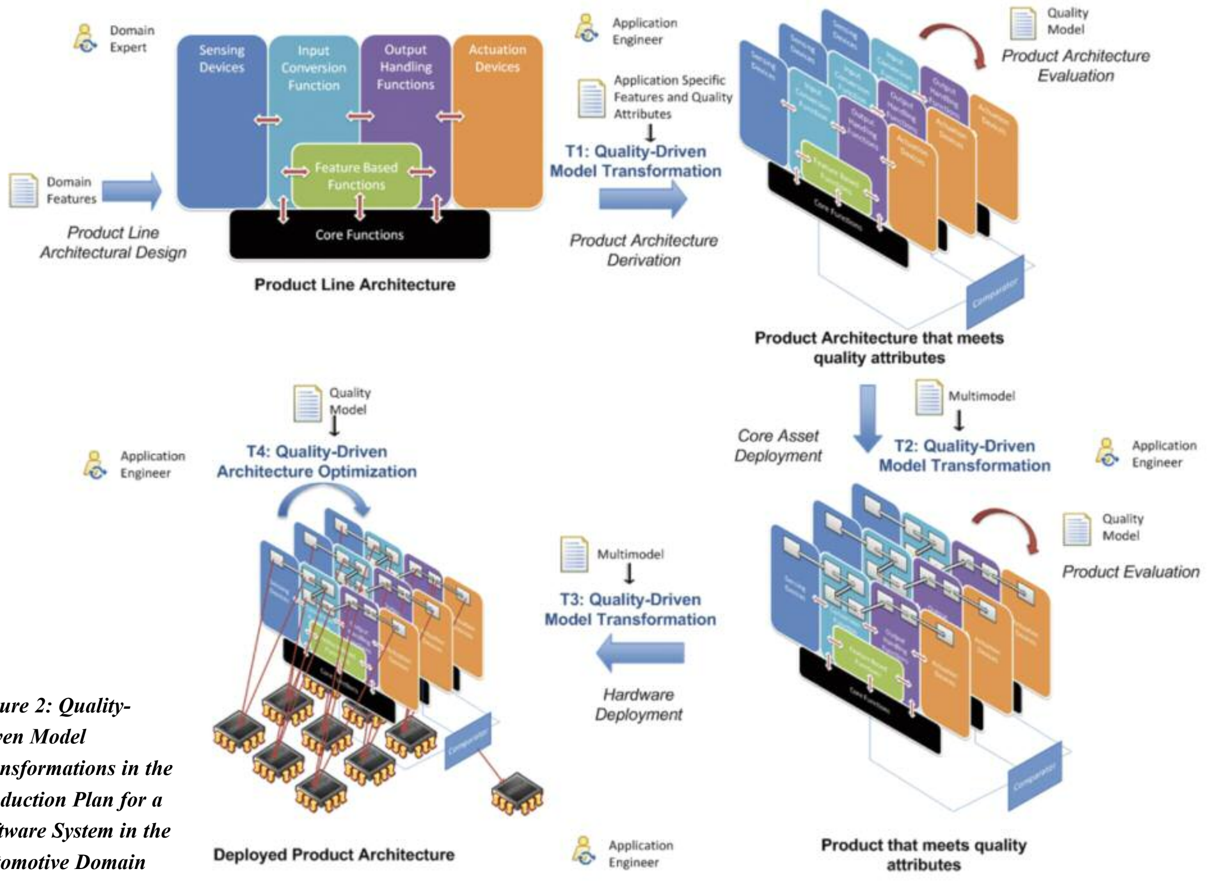
Domain Expert
Sensing Devices
Input Conversion Function
Output Handling Functions
Actuation Devices
Feature Based Functions
Core Functions
Domain Features
Product Line Architectural Design
Product Line Architecture
Application Engineer
Application Specific Features and Quality Attributes
T1: Quality-Driven Model Transformation
Product Architecture Derivation
Quality Model
Product Architecture Evaluation
Product Architecture that meets quality attributes
Multimodel
Core Asset Deployment
T2: Quality-Driven Model Transformation
Application Engineer
Quality Model
Product Evaluation
Product that meets quality attributes
Multimodel
T3: Quality-Driven Model Transformation
Hardware Deployment
Application Engineer
Quality Model
Application Engineer
T4: Quality-Driven Architecture Optimization
Deployed Product Architecture

Figure 2: Quality-driven Model Transformations in the Production Plan for a Software System in the Automotive Domain