Foreword
Quality in Model Driven Engineering

Vasco Amaral
FCT-UNL, Portugal
vasco.amaral@di.fct.unl.pt

MODEL-DRIVEN Engineering (MDE), or its formal and foundational approach Model-Driven Development (MDD), aims at supporting Software Engineering by using models (and models about models – metamodels), and (automatic) transformations among models as first class entities in the process to derive Software Products. It makes use of Models as abstractions to represent the systems and their different concerns or views, specified in adequate languages (formalisms), either Domain Specific Modeling (DSMLs) or General Purpose Modeling (GPMLs), at appropriate levels of abstraction.

On the one hand GPMLs such as UML, have made significant impact in the software industry, as it is a kind of "Lingua Franca", where Software Engineers are able to express and easily interchange information about their software products and processes. However, with the growing complexity experienced within certain Domains, GPMLs are becoming hard to handle specially due to their lack of expressive power. Namely, domain specific notations that use powerful and intuitive declarative concepts (such as justice, fairness, safety, etc.), otherwise difficult (or impossible) to express with the notations provided by traditional GPMLs which usually concentrate on notions from the domain of the solution (such as object orientation, data structures within the corresponding imperative programming languages).

On the other hand DSMLs are meant to narrow down the conceptual distance between the expressive power within HCI, and the way the users think about their work Domain, and how they specify and analyse their systems. This is achieved by providing suitable expressivity based on notations that use concepts used on the domain of the problem, while abstracting away concepts that are used in the domain of the solution which are usually influenced by a given computational platform or framework.

Once specified, these models can be transformed into other models by means of transformation models and/or automating code generation. The MDD approach allows for the possibility to soundly integrate into the same development framework: Code generation; Verification tools such as Model-Checkers or SAT solvers; Optimizers; and Simulation tools.

The common sense says that Quality is achieved by: a) reducing the complexity of Software development; b) involving the stakeholders into the software development process; c) speeding-up the development lifecycle, not only by increasing productivity (together with rapid prototyping) but also allowing for a sound and agile validation of requirements; d) promoting non-oportunistic re-use of components, by introducing the concept of software factories, or target platforms, that once tested can be used over and over reducing the risk of deriving defect software products, among others.

However, there is a wide range of problems related to the Quality of the process and the tools, that still remain to be solved so that this can be a solid framework. For instance, it is necessary to find adequate approaches, and find real life application examples, to assess the quality during the life-cycle and validate claims of increased quality and productivity by separating business and application logic from underlying platform technology. We should understand not only how to measure the quality of the MDD process (and determine if it is better than other approaches), but also to understand the quality of the models themselves (design patterns, and anti-patterns). Also, the transformation models need to be verifiable so that we know, with a certain degree of confidence, that the derived models are trustworthy representations of the originals. Besides, as verification tools like Model-Checkers have the well known scalability problem, not being usable for all situations, we have to complement with Model-Based Testing techniques.

This years' QUATIC thematic track we have contributions that approach some of the referred problems and that we will summarize next:

i) Achieving Quality by Model-Based Testing techniques: In "Structuring and Verifying Requirements Specifications through Activity Diagrams to Support the Semi-automated Generation of Functional Test Procedures", Jobson Massollar, Rafael de Mello and Guilherme Travassos present a methodology (with associated tools) for the specification of use cases with activity diagrams, with semantic check and test generation. In "Test Generation from UML Sequence Diagrams" by João Faria, Ana Paiva and Zhuanli Yang, the authors propose a lightweight method, and a plugin for Enterprise Architect, for generating test cases from sequence diagrams.

ii) Quality Assessment and MDD: In "A Multimodel for Integrating Quality Assesment in Model-Driven Engineering", Javier Gonzalez, Emilio Insfran and Silvia Abrahao present presents a multimodel approach for explicitly representing the relations or inter-relationships between models that might be expressed in different languages or not. The authors provide a syntactic model of the multimodel language, and a process for defining multimodels.

iii) Cases Studies to enhance Quality of Models and Product:

a) Comparison analysis: In "Survey on Cross-Platforms and Languages for Mobile Apps"

André Ribeiro and Alberto Rodrigues Da Silva, a brief comparison of some cross-platform development environments for mobile applications is presented.

b) Building automation: Finally in "Towards a robust solution in Building Automation Systems: supporting rapid prototyping and analysis", Vasco Amaral, Paulo Carreira and Bruno Barroca discuss the benefits of applying MDD is the domain of Building Automation as a way to fight heterogeneity and contribute to the integration of complex multi disciplinar information, being the technological enabler for BA Software Systems (simulation, monitoring, control, among others) systems' certification both at design time and runtime.

This said, as we can confirm, this year papers are over hot topics of quality in MDD and it is expected an exciting discussion during the presentations session.