

MPLM – MaTeLo Product Line Manager

Hamza Samih, Ralf Bogusch

▶ **To cite this version:**

Hamza Samih, Ralf Bogusch. MPLM – MaTeLo Product Line Manager. 18th International Software Product Line Conference (2014), Sep 2014, Florence, Italy. hal-01025159

HAL Id: hal-01025159
https://inria.hal.science/hal-01025159

Submitted on 17 Jul 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

MPLM – MaTeLo Product Line Manager

[Relating Variability Modelling and Model-based Testing]

Hamza Samih
ALL4TEC/INRIA Rennes
France
hamza.samih@all4tec.net
hamza.samih@inria.fr

Ralf Bogusch
Airbus Defence & Space
Germany
ralf.bogusch@cassidian.com

ABSTRACT

The diversity of requirements elicited from different customers leads to the development of many variants. Furthermore, compliance with safety standards as mandated for safety-critical systems requires high test efforts for each variant. Model-based testing aims to reduce test efforts by automatically generating test cases from test models.

In this paper, we introduce variability management to usage models, a widely used model-based testing formalism. We present an approach that allows to derive usage model variants from a desired set of features and thus generate test cases for each variant. The approach is integrated in the industrial model-based testing tool chain MaTeLo and exemplified using an industrial case study from the aerospace domain.

Categories and Subject Descriptors

D.2.1 [**Software Engineering**]: Requirements—*Languages*; D.2.5 [**Software Engineering**]: Testing and debugging—*Testing tools*; D.2.13 [**Software Engineering**]: Reusable Software—*Domain engineering*; G.3 [**Probability and Statistics**]: Markov processes

General Terms

Feature, model-based testing, product line engineering, product line requirements, product line usage model, usage model variant, variability, variability model, variant

Keywords

MaTeLo, Product Line Manager, Model-based Testing tool, OVM.

1. INTRODUCTION

Safety-relevant avionic systems are becoming more and more complex. Each system must satisfy specific customer requirements. However, those variants often share commonalities and hence may constitute a product line (PL). Additionally, safety standards like RTCA/DO-178C require rigorous verification activities to demonstrate compliance with applicable airworthiness requirements. Consequently, substantial efforts are spent to test each single variant. Due to the strong cost pressure in the aerospace market, industry is forced to optimise their life-cycle processes and move from single-product testing towards more efficient product-line testing.

In order to address these challenges, we propose an approach that relates variability models with so-called usage models [1, 2]. A usage model is a Model-based Testing (MBT) formalism, represented as a probabilistic state - transition system, which describes the possible use of a product. MBT is used to automatically generate test cases [3, 4, 5, 6]. During test case generation the traceability between requirements and test cases is automatically established. This is important when proving compliance of the tested product against its requirements.

The main reason for introducing variability modelling in MBT is the reduction of cost when test artefacts such as usage models can be strategically reused for different variants. Other reasons are the improvement of quality, since test artefacts are reviewed and used in several variants, and reduction of time to market due to the reuse of test artefacts for each new variant.

In this paper, we present the MaTeLo Product Line Manager (MPLM) tool that implements our approach [1, 2]. MPLM is an Eclipse RCP application that extends the industrial MaTeLo[1] tool chain. MPLM offers the possibility to relate a product line usage model with a variability model by establishing formal correspondences between features, requirements and a usage model. The relationships are then exploited to automatically synthesize usage model variants for valid configurations given by a desired set of features. Then, the derived usage model variants are used to generate test cases for desired variants. We apply the approach in a case study to a situational awareness suite which supports helicopter pilots in degraded visual environments.

The remainder of the paper is organized as follows. Section 2 describes the MPLM approach. Section 3 introduces the industrial case study, which exemplifies the MPLM approach. Section 4 concludes the paper and provides an outlook to future work.

2. MPLM IN A NUTSHELL

MPLM [2] is a perspective integrated in the new platform of MaTeLo tool based on Eclipse RCP (Figure 1), with around 14,000 lines of code in its current version. MPLM aims to adapt the MaTeLo tool chain to support product line testing. MaTeLo is an industrial MBT solution, which allows to automatically generate test cases for single products [5, 7] by using a usage model. With the help of MPLM, usage

[1]Markov Test Logic, http://www.all4tec.net/index.php/en/model-based-testing
[2]http://people.irisa.fr/Hamza.Samih/mplm

model variants can be derived for MaTeLo in order to generate specific test cases for variants of a product line. The MPLM approach can be summarized as follows:

- Develop a variability model with the Orthogonal Variability Modelling (OVM) approach [8]. OVM provides an explicit documentation of the variability related to functions, quality characteristics, and interfaces.
- Develop a product line usage model with MaTeLo [1, 2]. The usage model comprises traceability to the product line requirements [7].
- Associate product line requirements with features of the OVM model.
- Configure different variants by selecting the desired set of features.
- Derive usage model variants covering the relevant features and product requirements [1, 2].
- Generate test cases with MaTeLo for each variant based on the derived usage model variants [5] .

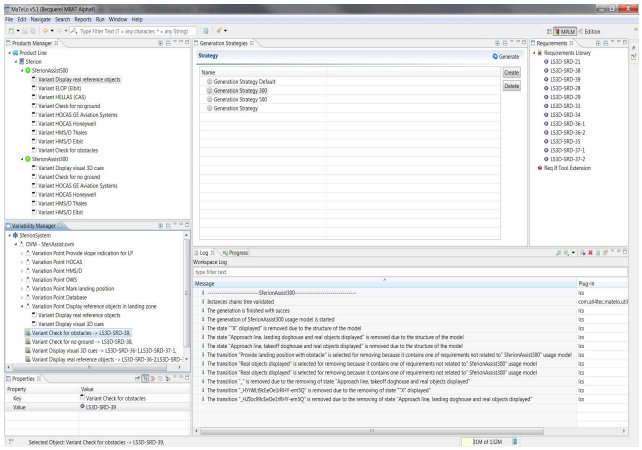


Figure 1: MPLM tool

3. INDUSTRIAL CASE STUDY

An experimental case study was performed together with the industrial partner Airbus Defence & Space in the frame of the ARTEMIS Joint Undertaking research project MBAT[3] in order to validate the approach from an industrial point of view [2].

3.1 Case Study Description

Airbus Defence & Space develops avionic systems that support helicopter pilots in degraded visual environments which can be caused by e.g. rain, fog, sand, dust and snow. Many accidents can directly be attributed to such degraded visual environments where pilots often loose spatial and environmental orientation. In this case study we employ the landing symbology function which is part of the situational awareness suite Sferion™. The landing symbology function supports helicopter pilots during the landing approach. It enables the pilot to mark the intended landing position on ground using a head-tracked HMS/D (Helmet Mounted Sight and Display) and HOCAS (Hands on Collective and Stick). During the final landing approach the landing symbology function enhances the spatial awareness of flying crews by displaying 3D conformal visual cues on the HMS/D. Additionally, obstacles residing in the landing zone can be detected and classified using a real-time OWS (Obstacle Warning System). The situational awareness suite Sferion™ constitutes a product line. Different features can be selected for the landing symbology function depending on the customer and the helicopter platform to which the solution shall be deployed.

The following sections summarize the steps performed in the case study and illustrate the use of the MPLM tool and the related MaTeLo tool chain.

3.2 Product Line Engineering

Product Line Engineering (PLE) [9] covers all activities that enable systematic reuse. This includes understanding the domain, identification of the scope, development of common assets, and managing variabilities.

Define product line scope. The product line scope provides the starting point of all subsequent activities. During scoping, we identify relevant product features and potential product configurations based on business and market information in order to bound the scope of the product line. This involves the following steps:

- Identify features which could provide value to at least one customer.
- Classify features according to the categories mandatory, optional, or range of alternatives.
- Assess the relevance of each feature in terms of value (ability to contribute to customer satisfaction), risk (maturity of development) and cost (effort required for development).
- Define the scope of the product line by including only features with high relevance.

Develop variability model. The result of the scoping is then formalized using the OVM approach [8]. The REMiDEMMI[4] tool is employed for developing variability models in the OVM language. The variable features identified during scoping are represented as variation points in OVM. For example, the OVM depicted in Figure 2 shows the variation point "Mark landing position" with dependencies to the mandatory feature "Check for no ground" and the optional feature "Check for obstacles". In case the optional feature is selected, a <requires> constraint assures that also an OWS is selected during the variant configuration.

[3]Combined Model-based Analysis and Testing of Embedded Systems, http://www.mbat-artemis.eu

[4]/http://remidemmi.cdhq.de/

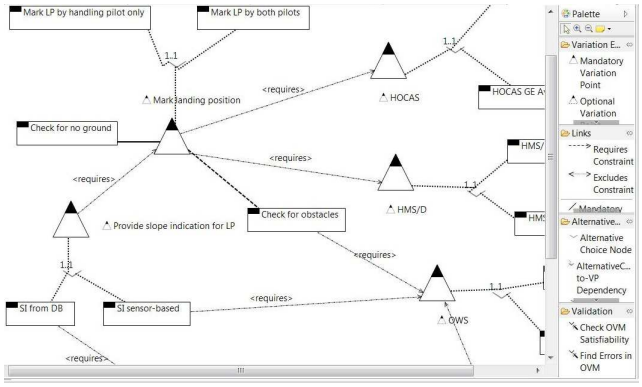


Figure 2: Create OVM in Eclipse environment

Develop product line requirements. Both common and variable features of the landing symbology function are refined into a set of system requirements. The requirements are created in the requirements management tool IBM Rational DOORS. In this case study only functional requirements are regarded. An example is given below:

> LS3S-SRD-38: The LS3D function shall visualize "NO GROUND" on reception of the mark-landing-position trigger, if the landing symbology has been activated and there is no intersection between LoS of the tracker of the HMS/D and the ground surface for the marked landing position.

> LS3S-SRD-39: The LS3D function shall visualize "X" on reception of the mark-landing-position trigger, whenever a sensor-classified obstacle is detected within the doghouse square centred at marked landing position.

Develop product line usage model. In this activity the functional system requirements for the product line are imported into the MaTeLo editor perspective and a product line usage model is developed. Figure 5 depicts the created usage model that covers all features and requirements of the product line that we want to test. The imported system requirements are assigned to the respective transitions of the usage model, as illustrated in Figure 3. In addition, test stimuli and expected behaviours are defined in the usage model and assigned to the equivalent transitions. Each transition has a probability for a profile. Profiles qualify the usage model to describe how the system under test will be used statistically. The usage model comprises several hierarchical levels which are composed of multiple extended Markov chains.

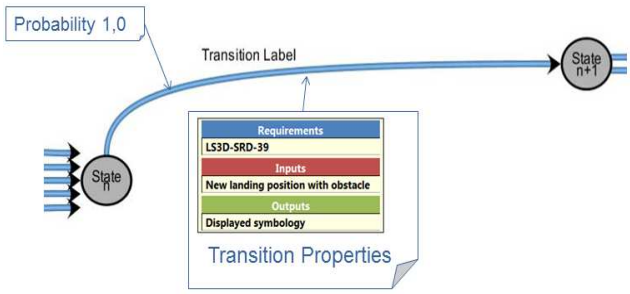


Figure 3: Data associated to transition

Associate product line requirements with features. The MPLM tool reads as input a product line usage model and an OVM model. As illustrated in Figure 4, MPLM lists all requirements covered by the usage model as well as the variable features provided in the OVM model. In this activity associations between requirements and features are defined. For example, the feature "Check for obstacles" is associated with the requirement "LS3D-SRD-39".

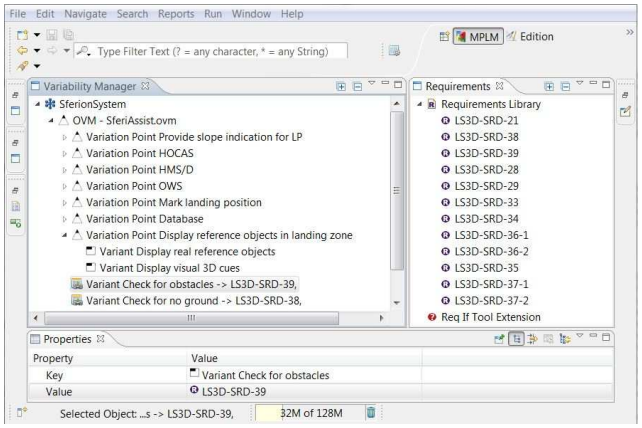


Figure 4: Associate requirements with features in MPLM

3.3 Product Engineering

A further major activity is the product engineering. Based on customer needs, business opportunities or customer contracts a decision is made to realize a product. The main goal of product engineering is to derive a dedicated variant by reusing as many assets as possible. In this case study, we select features desired for a variant and derive a product-specific usage model that allows to generate the test cases for a variant.

Configure variants. Using the MPLM tool, valid configurations can be specified by selecting the variable features which should be provided by the variant. During the configuration, the relationships between variation points and features (i.e. mandatory, optional, or alternative choices) as well as constraints between OVM model elements (e.g. requires or excludes) are respected. According to Figure 6, two different variants have been defined: SferiAssist500 (high-end product) and SferiAssist300 (low-end product). The feature "Check for obstacles" is only present in the high-end product.

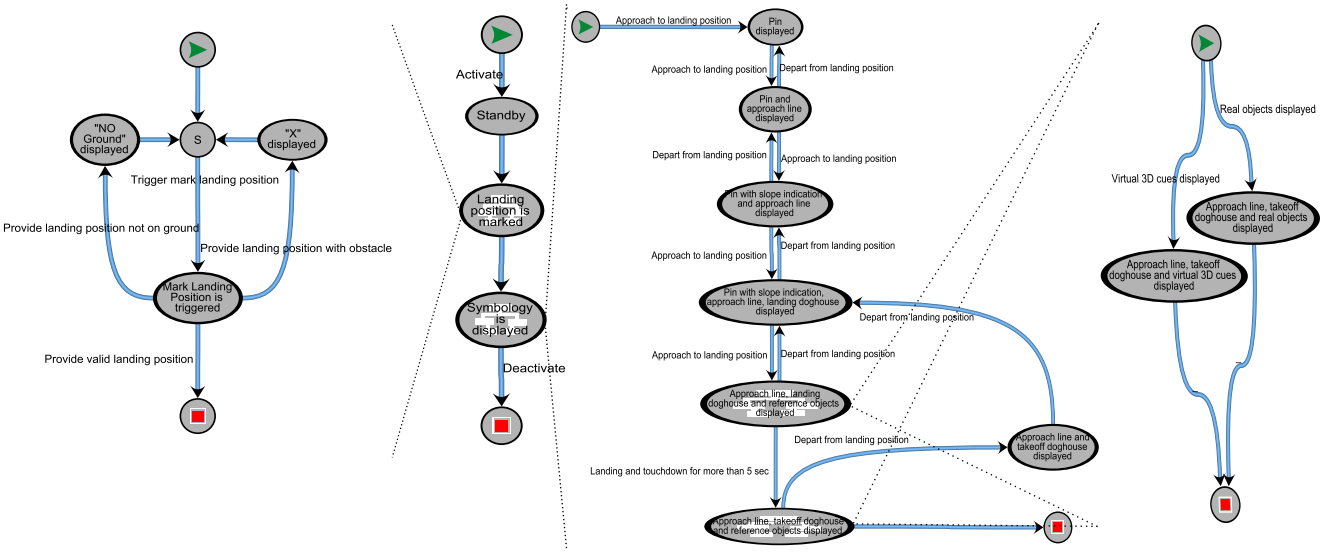


Figure 5: The product line usage model

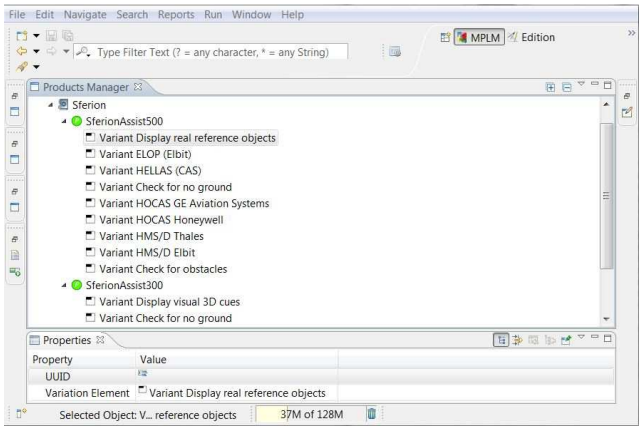


Figure 6: Configure variants in MPLM

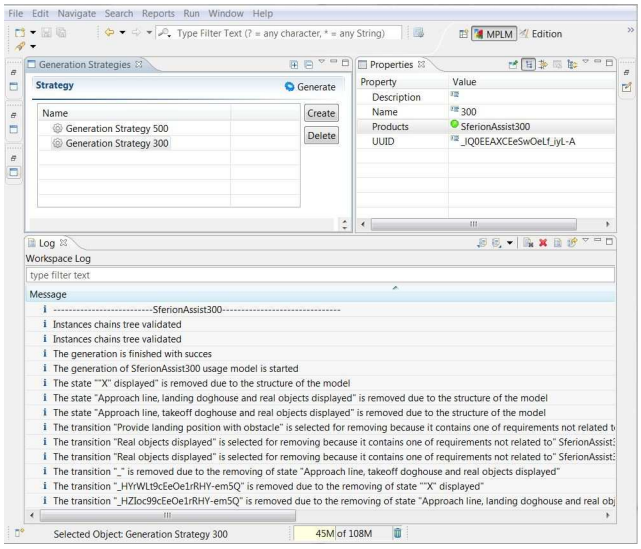


Figure 7: Derive usage model variant in MPLM

Derive usage model variant. In this activity a usage model variant that covers all selected features and associated requirements for the selected variant is derived from the product line usage model. The automated derivation consists of projecting a set of features composing a variant onto the PL usage model. The bindings between features, PL requirements and the transitions of the PL usage model enable reusing the PL usage model to extract a usage model variant with only the transitions and requirements of the selected variant. For instance, the derived usage model variant represents the test cases needed for the high-end product [2]. The derivation process ensure that the derived model is valid, by computing if there are any unreachable states in the model, if yes all broken branches will be removed, and the probabilities of the removed transitions will be distributed proportionally on adjacent transitions. Then, all associated profiles will be automatically updated [1]. Generation details, such as states and transitions that have been removed in the product line usage model, are displayed in a run log as shown in Figure 7.

Generate test cases. Product-specific test cases can automatically be generated based on the derived usage model variant using the MaTeLo generation perspective tool. The test case generation is based on defining a test strategy by choosing a usage profile and generation algorithm. Thereafter, the test case generation consists in selecting transitions according to their probabilities and the chosen algorithm.

3.4 Discussions

Our goal is to show that mixing MBT and PLE can reduce test costs and modelling effort when testing several product variants with a products line perspective. We observe that our approach does not impact existing MBT solutions (such as MaTeLo), since we extend the approach effective for single systems to a suitable PL solution. The variability is managed through an external variability model. Changes in the PL may also not impact the PL usage model, and con-

sequently, do not impact the existing test models. Several formalisms exist to describe PL [10], [11], [12], [13]. The variability model helps in identifying commonalities, differences and allows automated analyses. For instance, Hervieu et al. [14] propose to determine all possible variants of a PL that are valid for testing. Acher and al. [15] present *FAMILIAR*[5] a dedicated framework to the large scale management of variability models, that complements existing tool support. We have chosen OVM for the documentation of the variability, in order to allow deriving automatically new usage model for each new product variant from the PL usage model.

In our case, Eclipse RCP brings to MPLM the ability to be interconnected with formal analysis tools used in PLE to deal with combinatorial explosion. Furthermore, these parameters can help testers to adopt PLE in the testing process for large projects.

3.5 Results

An industrial experience with the situational awareness suite Sferion™ performed with MPLM and MaTeLo shows that practitioners can reduce the cost for test case development while raising the level of abstraction. We also report in [2] on the impacts of the approach in terms of adoption and testing methodology.

The experimental study comprises 12 system requirements and 16 features. The product line usage model provides 25 states, 45 transitions, and 5 chain instances. MPLM generates valid usage model variants with an average generation time of less than one second on a standard PC equipped with a 2.8 GHz processor and 8 GB RAM.

4. CONCLUSION

MPLM is a novel approach that extends model-based testing to product line engineering. The approach was applied in an industrial study from the aerospace domain. One major outcome of this case study is that the presented approach allows to significantly reduce the test efforts for variants of a product line. The product line usage model covers all features of the product line and hence allows to effectively reuse the test artefacts (e.g. generated test cases) across different variants. In the past, separate projects were setup for different customers and consequently requirements analysis and test case design was done independently for all variants without systematic reuse.

Future work. We are currently working to extend MPLM to support several variability models.

Acknowledgements. The research leading to these results has received funding from the ARTEMIS Joint Undertaking under grant agreement no. 269335 (ARTEMIS project MBAT[6]), French DGCIS and German BMBF.

5. REFERENCES

[1] H. Samih, M. Acher, R. Bogusch, H. Le Guen, and B. Baudry, "An approach to derive usage models variants for model-based testing," in *26th IFIP International Conference on Testing Software and Systems*. 26th IFIP International Conference on Testing Software and Systems, 2014, to appear.

[2] ——, "Deriving Usage Model Variants for Model-based Testing: An Industrial Case Study," Aug. 2014. [Online]. Available: http://hal.inria.fr/hal-01002099

[3] M. Utting and B. Legeard, *Practical Model-based Testing*. Morgan-Kaufmann, 2007.

[4] G. Perrouin, S. Sen, J. Klein, B. Baudry, and Y. Le Traon, "Automated and scalable t-wise test case generation strategies for software product lines," 2010, pp. 459–468.

[5] A. Feliachi and H. Le Guen, "Generating transition probabilities for automatic model-based test generation," in *ICST*, 2010, pp. 99–102.

[6] X. Devroey, M. Cordy, G. Perrouin, E.-Y. Kang, P.-Y. Schobbens, P. Heymans, A. Legay, and B. Baudry, "A vision for behavioural model-driven validation of software product lines," ser. ISoLA'12, 2012, pp. 208–222.

[7] H. Le Guen and T. Thelin, "Practical experiences with statistical usage testing," in *Proceedings of the Eleventh Annual International Workshop on Software Technology and Engineering Practice*, ser. STEP '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 87–93. [Online]. Available: http://dl.acm.org/citation.cfm?id=1032663.1034395

[8] K. Pohl, G. Böckle, and F. van der Linden, *Software Product Line Engineering: Foundations, Principles and Techniques*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[9] P. Clements and L. Northrop, *Software Product Lines: Practices and Patterns*, ser. The SEI series in software engineering. Addison Wesley Professional, 2002.

[10] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson, "Feature-oriented domain analysis (foda) feasibility study," Carnegie-Mellon University Software Engineering Institute, Tech. Rep., November 1990.

[11] B. Combemale, O. Barais, O. Alam, and J. Kienzle, "Using CVL to Operationalize Product Line Development with Reusable Aspect Models," 2012.

[12] Q. Boucher, A. Classen, P. Faber, and P. Heymans, "Introducing tvl, a text-based feature modelling," in *VaMoS*, 2010, pp. 159–162.

[13] T. Berger, R. Rublack, D. Nair, J. M. Atlee, M. Becker, K. Czarnecki, and A. Wąsowski, "A survey of variability modeling in industrial practice," in *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*, ser. VaMoS '13. New York, NY, USA: ACM, 2013, pp. 7:1–7:8.

[14] A. Hervieu, B. Baudry, and A. Gotlieb, "Pacogen: Automatic generation of pairwise test configurations from feature models," ser. ISSRE '11, 2011, pp. 120–129.

[15] M. Acher, P. Collet, P. Lahire, and R. France, "FAMILIAR: A Domain-Specific Language for Large Scale Management of Feature Models," *Science of Computer Programming*, Dec. 2012. [Online]. Available: http://hal.inria.fr/hal-00767175

[5] http://familiar-project.github.io/
[6] http://www.mbat-artemis.eu