

Usability Evaluation of Variability Modeling by means of Common Variability Language

Jorge Echeverria¹, Jaime Font¹, Carlos Cetina¹, and Oscar Pastor²

¹ Escuela Politécnica Superior, Universidad San Jorge, Spain,
jecheverria@usj.es, jfont@usj.es, ccetina@usj.es

² Centro de Investigación ProS, Universitat Politècnica de València, Valencia, Spain
opastor@dsic.upv.es

Abstract. Common Variability Language (CVL) is a recent proposal for OMG's upcoming Variability Modeling standard. CVL models variability in terms of Model Fragments. Usability is a widely-recognized quality criterion essential to warranty the successful use of tools that put these ideas in practice. Facing the need of evaluating usability of CVL modeling tools, this paper presents a Usability Evaluation of CVL applied to a Modeling Tool for firmware code of Induction Hobs. This evaluation addresses the configuration, scoping and visualization facets. The evaluation involved the end users of the tool whom are engineers of our Induction Hob industrial partner. Effectiveness and efficiency results indicate that model configuration in terms of model fragment substitutions is intuitive enough but both scoping and visualization results suggest that CVL visual notation should be improved.

Keywords: Usability Evaluation, Common Variability Language, Modeling Variability

1 Introduction

Common Variability Language (CVL) has been recently proposed by the architectural board of the OMG as Variability Modeling standard [9]. CVL expresses variability among models in terms of Model Fragments such as Placement Fragments (variation points) and Replacement Fragments (variants). The materialization of product models is performed by means of Fragment Substitutions between a Base Model (Placements) and a Model Library (Replacements).

Usability is a widely-recognized quality criterion essential to warranty the successful use of tools that put the above ideas in practice. This paper presents a usability evaluation of a Modeling Tool augmented with CVL (MT+CVL). The research question addressed by this evaluation is: Are Modeling Tools augmented with CVL intuitive enough to perform the main facets of variability modeling approaches (configuration, scoping and visualization)?

In order to materialize the ideas of CVL, we are going to use our industrial partner Modeling Tool, an induction hobs company that generates their induction hobs' firmwares following a model driven development approach. They used

to follow a clone and own approach (without explicit definition of variability) but we have augmented their modeling tool with CVL in order to model the variability existing among their products.

Our Usability Evaluation comprises both (1) test methods such as Performance Measurement, Satisfaction Questionnaire and Interview and (2) inspection methods such as KeyStroke-Level Model [11]. The human computer interaction research community advises to combine these methods to achieve reliable assessment. The selected Usability Evaluation Methods enable us to (1) assess effectiveness, efficiency and satisfaction and (2) to identify usability problems.

Effectiveness and efficiency results (configuration tasks 85% and 132.2%, scoping tasks 65% and 49.93%, visualization tasks 88% and 64.62%) indicate that model configuration in terms of model fragment substitutions is intuitive enough but both scoping and visualization require to improve the visual notation of CVL.

The remainder of this paper is structured as follows: Section 2 discusses related works. Section 3 summarizes the main concepts of the Common Variability Language. Section 4 presents an experimental study to evaluate the usability of the Modeling Tool with CVL. Then, Section 5 describes the results of evaluation and the set of Usability Problems detected. Finally we conclude the paper.

2 Related Work

There are research efforts in literature towards the visualization of SPL (Software Product Line) related artifacts. For instance in [14] the authors present an approach to visualize Pareto-optimal variants (variants, with respect to a set of objectives where no single quality can be improved without sacrificing other qualities). In addition [6] presents an approach that employs visualization and interaction techniques to support end users in the process of product derivation.

There is a concern in existing literature about the comprehensibility of feature models and possible difficulties for different user groups. For instance, in [16] the authors present an experimental approach in understanding of cross-tree constraints in feature models.

In [8] the authors present a Configurable Product Line tool that enable users of the PL to customize it. The authors abstract the technical issues of these customizations to help the users of the PL to understand the implications of decisions made during customization. Furthermore, in [15] the authors are concerned about the flexibility of their product line. Therefore, they present an end-user oriented tool that can support diverse end-users such as project managers, sales people or engineers in their specific tasks.

There is also a concern about the usability of DSL and the tool used to generate them. For instance, in [2] the authors present a comparison between five different development tool to create DSLs (and their associated editors). In [5] the authors discuss how user-centered design can be adapted to the context of DSLs development. They argue that usability should be fostered from the beginning of the DSL development cycle, enabling real people to use the DSL.

3 Common Variability Language

CVL is a Domain Specific Language (DSL) for modeling variability in any model of any DSL based on Meta-Object Facility (MOF). The CVL proposal [10, 7] is designed to work in conjunction with an existing DSL editor. Fig. 1 shows an overview of the application of CVL to a given DSL editor. Left part shows the DSL editor itself, while right part represents the library of replacements that will be used to define variants of the base model.

By means of the **use replacement operation**, users can perform substitutions, including fragments from the library into the model being edited. By means of the **create replacement operation**, users can create new replacement fragments and incorporate them into the library ¹.

These are the main elements and operations of CVL, and need to be fulfilled to apply CVL for a given DSL. It is necessary to augment the DSL editor in order to enable the operations defined by CVL, but its application is the same for any given DSL. For further details about the inner workings of CVL see [10].

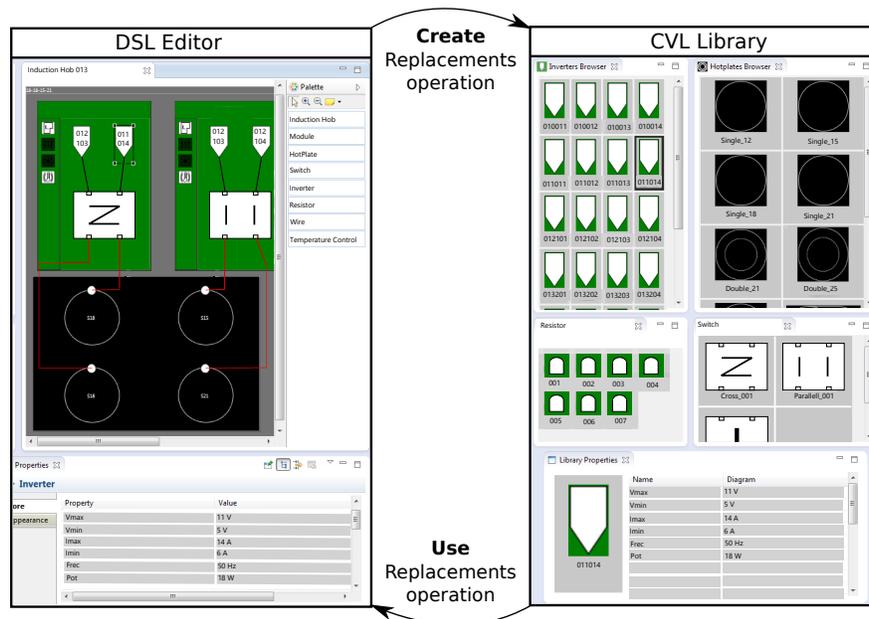


Fig. 1. Induction Hob MT+CVL

¹ Example of model fragment operations: <http://folk.uio.no/oystein/demo1.htm>

4 Experimental Study

4.1 Context of the Experiment

We have applied CVL to the modeling tool of our industrial partner. That is, we have augmented the Modeling Tool including and integrating the CVL operations and library (as presented in section 3), resulting in the MT+CVL that will be used through the rest of the study. This is the usual operation for augmenting an existing Modeling tool with CVL, and would be the same when applying to any other modeling tool.

Left part of Figure 1 shows the graphical editor of the models. Right part shows the replacements library, where all the replacements that are part of the MT+CVL are shown. In addition, the create replacement operation, enables engineers to create new replacements fragments that are included into the library. The Replace operation enables the engineers to substitute model elements from a product model (open in the editor) by replacements of the replacements library.

4.2 Experimental Object

To evaluate the usability of the MT+CVL we had to know the main tasks that the end users perform in the main facets of variability modeling: Scope (in CVL, the creation and elimination of fragments), Configuration (the derivation of products) and Visualization (to make the user aware of the variability). Six executable tasks were produced as output ²:

- T1** The induction hob IH013 has a problem with the module MOD008 and this module must be replaced by the module MOD014. In the other induction hobs the module MOD008 must not be replaced.
- T2** The inverter INV016034 in the module MOD017 in the induction hob IH021 does not run correctly. The module must assemble the inverter INV019034. This replacement must affect every induction hob with the above module.
- T3** The induction hob IH021 in the module MOD073 has the inverter INV015034. Its parameter is wrong. A new inverter must be created by cloning the wrong inverter. The new inverter has its parameter VMAX equal to 42. The replacement must affect every induction hob with the above module.
- T4** The module MOD021 in IH003 must replace the inverter INV015042 by INV016042. This replacement must not affect to other induction hobs.
- T5** To detect all components in the induction hob IH021.
- T6** Which is the module most widely used of the set of modules (MOD021, MOD014, MOD017, MOD101)?

The tasks (T1) and (T2) are from configuration facet tasks, (T3) and (T4) are from scope facet tasks and, finally, (T5) and (T6) are from visualization facet tasks.

² The identification of the components has been sanitized in order to preserve confidential information. However, omitted information is not relevant for the approach.

4.3 Evaluation Without Users

The Inspection Method (without users) chosen is Keystroke-Level Model. The Keystroke-Level Model has two phases. The first phase is to determine what physical and mental steps a user performs to complete one or more tasks with the CVL Modeling tool in order to predict the time that the user needs to do the task. To do this, a duration is associated to each one of these actions or sequence of operators (physical or mental), and then they are totaled. This duration is calculated by using the average time that it takes a skilled user to complete the action, as suggested by reference time values of [13].

The second phase is to analyze the above steps, looking for problems. Some usability problems that the Keystroke-Level Model might reveal are that it takes too many steps to perform a simple task, or it takes too long to perform the task, or there is too much to learn about the interface, etc. [1]. Furthermore, the amount of time that the user needs to do each task is obtained. In our experiment a Usability Engineer performed every task of section 4.2. For instance, the task1 is composed by four subtasks and the total time predicted to perform the task is 21.1 seconds.

4.4 Evaluation With Users

The objectives of this phase are the assessment on Usability Measures and the identification of usability problems. To achieve these objectives the following UEM are used: Demographic Questionnaire, Performance Measurement, Satisfaction Questionnaire and Interviews. These UEM are characterized by the participation of the end users. The evaluation with users was as follows [1]:

1. End users were given information about the goals and objectives of the evaluation. They were told that it is not a test of their abilities. They were also informed that their interaction will be recorded.
2. End users attended to a small tutorial about the MT+CVL.
3. End users were asked to fill in a demographic questionnaire.
4. End users were then given a series of clear instructions that were specific for the Performance Measurement. They were advised to try to accomplish the tasks without any assistance, and that they should only ask for help if they felt unable to complete the task on their own.
5. End users were asked to complete the six tasks detailed in the section Experimental Object 4.2. To avoid a possible ceiling effect, there was no time limit to complete the tasks.
6. End users were asked to complete a System Usability Scale questionnaire.
7. End users were asked to answer an interview about CVL Modeling tool.

The evaluation involved the end users of the tool whom are engineers of our Induction Hob industrial partner. The human computer interaction research advises to use five end users in the usability test to obtain 80% of the usability problems [17]. For this reason, we chose a usability evaluation with five end users.

Performance Measurement The goal of this evaluation step was to evaluate how well or poorly the MT+CVL performed for users. Specifically, we measured user effectiveness and efficiency (ISO 1998). An Instructor, an Evaluator and five end users participated in the study. The function of Instructor was to explain the test to the end users and to solve doubts of the end users. The goal of Evaluator was to collect data about the end users action.

In this UEM users performed a predefined set of test tasks (see 4.2) while time and error data was collected. Quantitative data includes performance times, error rates, completed tasks or number of assistance. This data enables the calculation of efficiency and effectiveness. Usability problems will come from the notes that the Evaluator has taken down during the test or extracted from an audio or video recording of the session.

Table 1. Results of Effectiveness and Efficiency

		Unassisted Task Effectiv. (%)	Assisted Task Effectiv. (%)	Time (min)	Completion rate/ Task time	Asistence
Configuration	Mean	85%	9%	0,86	132,20%	0,2
	Std desv	31%	21%	0,66	55,47%	0,42
	Min	20%	0%	0,16	44,78%	0
	Max	100%	0%	2,23	206,25%	1
Scope	Mean	65%	6%	1,91	49,93%	0,1
	Std desv	41%	19%	1,39	41,52%	0,32
	Min	0%	0%	0,57	0,00%	0
	Max	100%	61%	4,80	139,53%	1
Visualization	Mean	88%	0%	1,94	64,62%	0
	Std desv	25%	0%	1,75	34,85%	0,00
	Min	36%	0%	0,82	14,96%	0
	Max	100%	0%	6,68	111,11%	0

Measures of effectiveness take into account percent of right finished unassisted tasks, percent of right assisted tasks, frequency of assists to the participant. The efficiency value is the ratio between percent of right finished unassisted tasks and the time to finish these tasks according to Common Industry Format (CIF) for Usability Test Reports [3].

The values in Table 1 indicate that the most difficult or problematic tasks are the scope tasks. In contrast, the end users performed with great easy configuration tasks. On the other hand, the end users performed correctly the visualization tasks, but it took them too much time taking into account the calculated values with Keystroke-Level Model (see Section 4.3).

Satisfaction Questionary After the performance measurement, a satisfaction questionnaire was filled by the end users. This questionnaire was System Usability Scale (SUS). SUS was used to determine user's subjective satisfaction with the

SPL tool. Measuring user satisfaction provides a subjective usability metric. The questionnaire was composed by a ten questions with a Likert scale.

The data collected with SUS must be introduced in a spreadsheet to process them. The global score was 73%, which shows that the end users classified the CVL Modeling tool as “good”, according to the scale suggested by [4].

Interview The last UEM used in this phase is Interview. The objectives of this interview were (1) to determine the understanding by the end user of the CVL Modeling tool and (2) to obtain qualitative data from user comments.

The Interview Questions to perform this step had open questions and closed questions. The closed questions were directed to check the understanding of the tasks in the MT+CVL by the end users. For instance, the Instructor shown two pictures to the end user with the state of the CVL Modeling tool after a task and the end user had to choose wich picture is the correct. The open questions aim was to detect the parts of the MT+CVL that were more problematic from a usability point of view, along with the real causes of the problems [12]. For instance, a question was “What have been the more difficult of the tasks for you?”.

5 Conclusion

We believe the results of the usability evaluation are relevant to model-based software developers, OMG variability standardization process and variability tools vendors as follows:

From the point of view of model-based software developers, as the case of our industrial partner, the usability evaluation results suggest that CVL can complement their current modelling tools to formalise and configure variability (according to the result of Effectiveness and Efficiency of variability tasks). The CVL library of model fragments turns out to enable them to shift from a Clone & Own approach to a systematic reuse of model fragments.

From the point of view of current OMGs variability standardization process this paper provides evidence that the current CVL proposal should be extended to provide a visual notation for the model fragment concepts. That is, current CVL proposal introduces the concepts of model placement and model replacement but the proposal lacks a concrete syntax to denote the model fragment boundaries. This lack of visual notation leads modellers to miss variation points in the models.

Finally, from the point of view of tool vendors, the usability evaluation results reveal that modellers require new editing capabilities to work with independent model fragments such as explicit creation, fragment comparison, fragment-based filters and propagations of changes.

References

1. S. Abrahão, E. Iborra, and J. Vanderdonckt. Usability evaluation of user interfaces generated with a model-driven architecture tool. In *Maturing Usability*, pages 3–32. 2008.
2. D. Amyot, H. Farah, and J.-F. Roy. Evaluation of development tools for domain-specific modeling languages. In R. Gotzhein and R. Reed, editors, *System Analysis and Modeling: Language Profiles*, volume 4320 of *Lecture Notes in Computer Science*, pages 183–197. Springer Berlin Heidelberg, 2006.
3. ANSI/NCITS. Ansi/ncits-354 common industry format (cif) for usability test reports. Technical report, NIST Industry Usability Reporting, 2001.
4. A. Bangor, P. Kortum, and J. Miller. Determining what individual sus scores mean: Adding an adjective rating scale. *JUS*, 4(3):114–123, 2009.
5. A. Bariic, V. Amaral, and M. Goulao. Usability evaluation of domain-specific languages. In *Quality of Information and Communications Technology (QUATIC), 2012 Eighth International Conference on the*, pages 342–347, Sept 2012.
6. G. Botterweck, S. Thiel, D. Nestor, S. bin Abid, and C. Cawley. Visual tool support for configuring and understanding software product lines. In *Software Product Line Conference, 2008. SPLC '08. 12th International*, pages 77–86, Sept 2008.
7. F. Fleurey, Ø. Haugen, B. Møller-Pedersen, G. K. Olsen, A. Svendsen, and X. Zhang. A generic language and tool for variability modeling. *Technical Report SINTEF A13505*, 2009.
8. P. Grubacher, R. Rabiser, and D. Dhungana. Product line tools are product lines too: Lessons learned from developing a tool suite. In *Automated Software Engineering. 23rd IEEE/ACM International Conference on*, pages 351–354, 2008.
9. Ø. Haugen. Common variability language (CVL) - OMG revised submission. *OMG document ad/2012-08-05*, 2012.
10. Ø. Haugen, B. Møller-Pedersen, J. Oldevik, G. K. Olsen, and A. Svendsen. Adding standardized variability to domain specific languages. In *Proceedings of the 2008 12th International Software Product Line Conference, SPLC '08*, pages 139–148, Washington, DC, USA, Sept 2008. IEEE Computer Society.
11. A. Holzinger. Usability engineering methods for software developers. *Communications of the ACM*, 48(1):71–74, 2005.
12. N. J. Juzgado and X. Ferré. How to integrate usability into the software development process. In *ICSE*, pages 1079–1080, 2006.
13. D. Kieras. Using the keystroke-level model to estimate execution times. *University of Michigan*, 2001.
14. A. Murashkin, M. Antkiewicz, D. Rayside, and K. Czarnecki. Visualization and exploration of optimal variants in product line engineering. In *Proceedings of the 17th Software Product Line Conference, SPLC '13*, pages 111–115. ACM, 2013.
15. R. Rabiser, D. Dhungana, W. Heider, and P. Grubacher. Flexibility and end-user support in model-based product line tools. In *Software Engineering and Advanced Applications, 2009. SEAA '09. 35th Euromicro Conference*, pages 508–511, 2009.
16. I. Reinhartz-Berger, K. Figl, and y. Haugen. Comprehending feature models expressed in cvl. In *Model-Driven Engineering Languages and Systems*, volume 8767 of *Lecture Notes in Computer Science*, pages 501–517. 2014.
17. R. A. Virzi. Refining the test phase of usability evaluation: how many subjects is enough? *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 34(4):457–468, 1992.