Modelling and Managing Variability with Feature Assembly – An Experience Report

Lamia Abo Zaid
Vrije Universiteit Brussel
Pleinlaan 2
1050 Brussels, Belgium
Lamia.Abo.Zaid@vub.ac.be

Olga De Troyer
Vrije Universiteit Brussel
Pleinlaan 2
1050 Brussels, Belgium
Olga.DeTroyer@vub.ac.be

ABSTRACT
Feature models have been commonly used to model the variability and commonality in software product lines. We have defined the Feature Assembly Modelling, a feature modelling technique that allows to model variability in software adopting a multi perspective approach. Furthermore, the approach allows modelling software by combining both variability and reusability, i.e. we have developed an approach to take reusability into account while defining new software. To support the approach, we have also developed an information retrieval framework that provides an interactive visualization of the feature models. The visualization allows users to explore and query the existing models. In this paper, we report on our experience in introducing this variability modelling approach into a small-scale software company. This experience was very useful for both parties. The company was able to uncover the structure of their software and the modelling exercise provided them better insight in their products. For us, it has helped to better understand the needs of companies, to evaluate the usability of our Feature Assembly approach and the associated learning curve, as well as revealing its current limitations. Moreover, as we are aware of the fact that classical feature modelling is not yet a practice adopted by companies, it was interesting to see that our approach was well accepted and appreciated by the company.

Categories and Subject Descriptors
D.2.13 [**Software Engineering**]: Reusable Software – *Domain Engineering.*

D.2.10 [**Software Engineering**]: Design – *Methodologies, and Representation.*

General Terms
Design, Management.

Keywords
Feature Models, Feature Assembly, Software Product Lines, Variability, Variability Management, Experience Report

1. INTRODUCTION
Companies developing multiple related software products or products having different variants (i.e. software product lines) are faced with many challenges as the complexity of the software increases considerably by introducing variability. For instance, how can they keep an overview on the many different variants of the products produced and installed for different customers; how do the different features of the variant products relate to each other; which dependencies exist between the different features; what is the impact of making some changes to a certain feature; etc. We claim that in order to avoid problems in later stages of the software life cycle, it is necessary to perform a thorough modelling phase of the variability in a product (or family of products) in order to reveal the complexity introduced by common and variable features, feature relations, and feature dependencies.

Different feature oriented modelling techniques (e.g., [1] [2]) exist that allow to model variability and commonality in software. The term "feature" is used to denote an abstraction that different stakeholders can understand [1]. Despite the relevance of feature modelling for industry, a recent study [3] reveals that there are very few reports in the feature modelling literature on the application of feature models in practice. Furthermore, a few of the detected papers discuss successful and unsuccessful applications of feature modelling in practice. Results show that companies often had doubts when applying the feature modelling technique and its usefulness was not fully anticipated beforehand [3].

Furthermore, it is agreed upon both in academia and industry, that analysis and design are often underestimated when developing new products [4] [5]. The impact of good design is obvious, yet good practice remains a challenge. Furthermore, it was found that in small and medium scale companies variability is not planned beforehand [4] [6] but actually evolves with time due to the expansion of the software to serve more customers or due to the need to customize some features to meet the different needs of different customers [7]. In these situations, a poor product design may create problematic situations as the software becomes difficult to extend, becomes extremely complex and unstable, and most of the company's development time will be spent in bug fixing, maintenance, and testing.

This paper presents our experience of introducing our own Feature Assembly Modelling approach to a software company. We aimed to validate the approach using a real case. Moreover, as we are aware of the fact that classical feature modelling is not yet a practice adopted by companies, we were interested in understanding the reasons for this and check if our Feature Assembly Approach could provide a solution for this. The paper is organized as follows: in section 2, we provide some background on variability modelling and feature modelling. Section 3 presents our Feature Assembly approach. Next, in section 4 we describe the case study and introduce a set of research questions that this case study aimed to answer. Next, in section 5, we present the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
EESSMOD'12, October 01 - 05 2012, Innsbruck, Austria
Copyright 2012 ACM 978-1-4503-1811-2/12/10…$15.00.

accomplishment of the case study and section 6 presents the results. Section 7 provides a discussion of the results and section 8 presents the threats to validity. Finally, section 9 provides the conclusions and an overview of our future work

2. BACKGROUND

Companies developing related software products or products having different variants (i.e. a software product line) are faced with many challenges as the complexity of developing, maintaining, and managing the software increases considerably due to introducing variability. Software variability is defined as "*the ability of a software system or artefact to be efficiently extended, changed, customized or configured for use in a particular context"* [8]. In order to gain the merits of variability, there is a need for expressing this variability through variability analysis and variability modelling. Feature oriented domain analysis techniques have been commonly used to analyse the variability and commonality of variable products [2]. The resulted variability model is referred to as "feature model" and has been commonly used to model the variability and commonality [1] [2] [9]. Feature models are visual representations (graphs) of the features which the software is composed of in addition to their feature relations, feature dependencies, and their contribution to the variably of the system.

Despite the value of feature modelling it has not found its way to industry. This could be due to the complexity of the modelling technique [10], missing support/training on how to apply feature modelling [3], the different tools with different functionality and usability support [11], or simply because variability was not anticipated from the start but developed overtime as the product matured. A variability modelling technique should be expressive and intuitive enough to capture and represent information about features composing the software product line in addition to how these features contribute to the variability of the software product line. In our previous works, we have identified limitations of mainstream feature modelling techniques [10]. In order to overcome these limitations, we have proposed the Feature Assembly Modelling technique [10].

In addition to modelling variability there is a need for managing this variability. Feature models act as a medium for communicating product capabilities between different stakeholders. As the number of features grows, along with the increasing number of relations between features, finding information manually becomes difficult. Therefore there is a need to allow stakeholders to browse these feature models for information [12]. In addition, there is a need for efficient retrieval of information (e.g. search) from feature models [13] [14].

3. FEATURE ASSEMBLY

The main goal behind *Feature Assembly* [10] [15] is to be able to specify variable software products (e.g., a software product line) by combining and reusing (existing) software features. In doing so, reuse is promoted and supported from the initial software conception phase through the complete software development life cycle.

The Feature Assembly approach should help companies define their products better, by first conceiving them in terms of "features". In addition, in the Feature Assembly modelling technique, one needs to distinguish between features that represent variability (i.e. variation points) and those that do not. In order to deal with the size and the complexity of the models, the concept of "*perspective*" was introduced. A perspective allows considering the modelling from one particular point of view at a time, e.g., the system perspective, the task perspective, or the user interface perspective (other perspectives may exist as well). The modelling technique separates features from how they contribute to variability. This makes it possible to reuse features in different variability contexts. Furthermore, this could encourage companies to consider reuse as early as the design phase, not only by allowing to reuse existing features but also by forcing them to design for reuse. Considering reuse at the design level will enable reuse at the implementation level, increasing the overall productivity and reducing cost.

Feature Assembly does not only allow modelling of variability, it also offers continuous management of the information contained in these models. This functionality is provided by our Feature Assembly Framework [15]. The framework is based on a repository for storing features, their relations and dependencies, called the *Feature Pool*. The Feature Pool Manager allows exploring and searching the pool for reusable features. The Feature Pool may also be used to store complete Feature Assembly Models. In this case, and using the Feature Pool, users can both browse and search the information contained in Feature Assembly models. The results are presented in a visual way, providing better insight in the information and models retrieved. This actually allows Feature Assembly models to act as an interactive documentation source, where users can readily find information. This should allow for improved understanding, management, and reuse of existing software features, as existing features can efficiently be identified as well as their dependencies. Additionally, it could support decision-making by unlocking the knowledge about the software features already developed in the company.

4. CASE STUDY - PLANNING

The case study presented in this paper was part of the living labs initiative of the VariBru[1] project. The idea was to conduct a pilot study in which our Feature Assembly approach was assessed by a company that encounters variability in their products and wants to explicitly represent this variability. The company did not apply the concepts of variability analysis and modelling before. The case study aimed applying the Feature Assembly approach for analysing and modelling variability from the domain analysis and design perspective. The objective was to evaluate the Feature Assembly approach and improve our understanding of the variability modelling needs of companies.

4.1 METHODOLOGY

We introduced the Feature Assembly approach to a small-scale software company, Antidot (located in Brussels), and applied it to a (variable) product of the company. Antidot is working in the domain of web-based IT solutions and services for corporations, companies and associations. To provide these services, they have developed their own Content Management System (CMS) which can be customized (i.e. configured) in different ways to serve the needs of their different customers. Antidot was interested in the approach as they wanted to increase the configurability of their product; they were looking for a way to help them keep track of the different variants of their features (in order to make more accurate customer offers) and the different configurations installed for different customers. Two employees of the company participated in the semi structured and flexible meetings we set up for conducting this case study; the first held the roles of CEO

[1] www.varibru.be

and Senior Project Manager; the second held the role of senior developer and designer (there is no dedicated team for the design). From our research team the authors were the participants, the case study was managed by the first author.

4.2 MOTIVATION

Antidot's CMS product has experienced an increase in features as well as feature variation (i.e. new variants of features were introduced) over time. This has led to a situation in which there was a need to track the available features, the features that hold variations or represent variation points, and the dependencies between these features.

As previous research [4] has showed that small and medium scale software companies are confronted with variability issues, it was interesting to validate the power of Feature Assembly in bringing variability modelling into practice for these companies in order to help solve their variability problems. We have formulated a set of questions that are oriented to measure the relevance of our Feature Assembly approach for the company. These questions can be applied to companies in a similar situation, i.e. that have some form of variability in their products, and which did not yet apply a feature modelling technique.

RQ1. Does the Feature Assembly approach bring variability modelling and management one step closer to industry, i.e. does the company Antidot see added value in adopting this variability modelling approach?

RQ2. Does the company have a problem of concealed information (i.e. information hidden in code, paper documents, or in the heads of the developers)?

RQ3. Can we promote reuse early in the development cycle? Will that make a difference for the development cost?

RQ4. Is our Feature Assembly approach effective and usable in practice?

The case study aimed at finding answers to these questions. Answering these questions should help us gain better understanding of the approach's feasibility as well as its limitations.

5. CASE STUDY - EXECUTION

We had several meetings with members of Antidot. In the first meeting, we introduced the Feature Assembly Modelling approach, i.e. the Feature Assembly Modelling Language and the concept of modelling by reusing features from an existing Feature Pool. We illustrated the concepts using an example. The company also explained how they currently manage their features. In order to help them understand the modelling technique, we made some models for their (existing) CMS and presented them in the next meeting; this quickly initiated a discussion as they saw mistakes in our models (which was not surprising as we didn't know all the details of their software), a sample of their corrections is shown in figure 1. We then asked them to do the modelling process on their own (as homework) and provided them with some basic documentation material about the modelling technique. During our meeting we used a pen-and-paper approach for creating the models (or rather modifying the created models). To help speed up the modelling process we have defined a Visio Stencil that draws the notations of the Feature Assembly Modelling

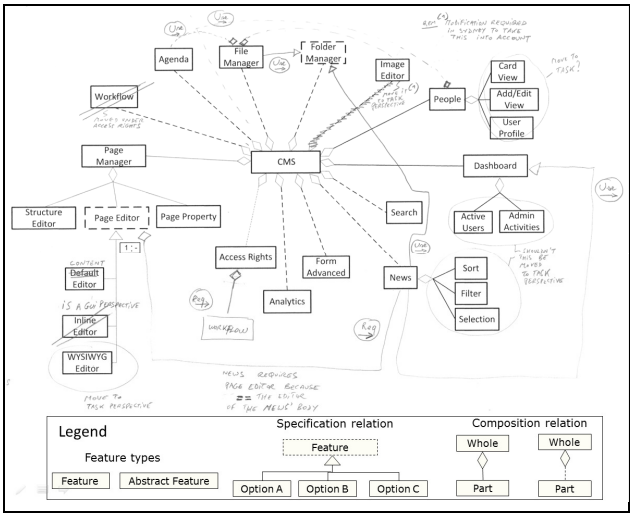

Figure 1. Excerpt showing early modelling of CMS to demonstrate modelling with Feature Assembly

Language, we provided Antidot with this stencil[2]. In the following meetings, we discussed their models, answered their modelling questions, and collected their comments on the ease of use and intuition of the modelling approach. We also introduced the Feature Assembly Framework prototype[3] that we created for testing the approach and asked them to try it out. In a next meeting, we collected their comments concerning the functionality and the usability of this prototype. By asking them to try out the prototype of the Feature Assembly Framework we wanted to investigate and better understand how companies want to be able to search for information about their designs, and what types of information they consider useful or essential. The available prototype visualizes the Feature Assembly models and allows users to navigate visually through the models in order to find features and information about features.

Figure 2. Screenshot showing how information can be found in Feature Assembly Models using form-based querying - Applied to the models of Antidot.

Furthermore, the prototype allows users to search for information based on predefined metadata such as feature name, feature description, feature type, feature definition date (as interval), perspective name, perspective description. Additionally, the features belonging to a specific perspective can be shown. Also a tag cloud is used to enhance the searching via tags that can be assigned to features. The tag cloud also indicates the frequency of tags used to label features.

---

[2] Note that at that time, a dedicated tool to create the Feature Assembly models was not yet available.

[3] The Feature Assembly prototype is a web-based application to visualize and explore feature assembly models (also used for Feature Pool contents). The examples can also be found on https://wise.vub.ac.be/feature_assembly/examples.html.

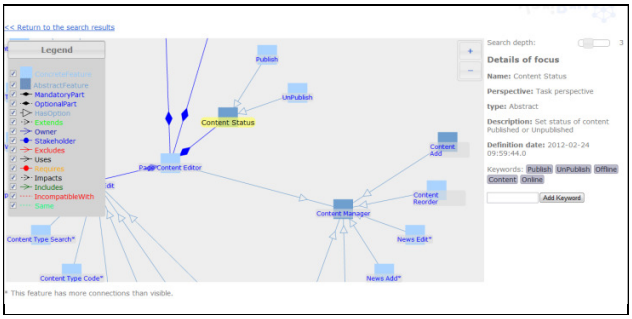

Figure 3. Screenshot showing how Feature Assembly Models are visualized allowing users to browse the information contained in the models - Applied to the models of Antidot.

6. RESULTS

In this section, we elaborate on the results obtained from this evaluation activity. We first present the results obtained for the Feature Assembly modelling technique and next the ones for the Feature Assembly Framework.

6.1 The Feature Assembly Modelling Technique

Concerning the effectiveness and ease of use of the Feature Assembly Modelling technique, Antidot's team reported the following. To analyse and model[4] one major module of the CMS, it took one person two hours and a half. The resulted model contained 28 features and 21 connections between features (14 feature relations and 7 feature dependencies). They found this an acceptable modelling time, although there was some overhead because it was the first time they use a variability modelling technique. In total, three persons were involved in the modelling of the CMS. A small issue was the learning time needed for the notations used, although they appreciated the similarity with the UML notations (as they are using UML for system modelling). Despite this, we have noticed that already after the second modelling meeting, the team was very comfortable with the modelling technique, capable of making decisions concerning feature types and dependencies. Other remarks concerned the expressiveness of the modelling approach. We report their major remarks:

R1. Some features needed many feature dependencies and this was cumbersome to specify.
R2. The distinction between some feature dependencies was not clear and this made it difficult to decide which one to use (e.g., 'uses' versus 'requires')
R3. They were wondering at which level of detail they had to model.
R4. It was not clear how they could specify external features/components.
R5. Sometimes it was difficult to decide which perspective to use for modelling certain features.
R6. It was not clear if and how they could model different versions of the same feature.

Some of these remarks are due to the lack of experience with the Feature Assembly modelling technique and the lack of good documentation for the method (e.g., an elaborated user guide), such as remarks R2, R3, and R5. Also remark R5 was given because they assumed that a feature could only belong to one perspective, which in not the case. Remark R1 triggers some important questions: Is it necessary to always model all dependencies? Is a high coupling not an indication of some bad design decision? And if all dependencies are really justified, can we not find an easy way to specify them, e.g., by introducing some abstraction mechanisms to reduce the number of links that need to be specified? There is no unique recipe for solving this issue, on the other hand knowing this kind of information at a design time allows considering design patterns to eliminate such coupling between features in code [16]. Remarks R4 and R6 were indeed very valuable, as the method currently doesn't provide support for this. Currently, the Feature Assembly Modelling technique treats all features (external and non-external) similarly. Also it does not support versioning of features. These issues should be considered in future work.

Furthermore, the case study has revealed/confirmed the following merits of adopting such a variability modelling technique:

1. Feature Assembly let them reconsider their "*features*" in order to increase the modularity of the software. Using the Feature Assembly Modelling technique, dependencies between features became more visible and they can use this to improve the design for achieving a lower degree of coupling between modules/components at the code level. In their own words they mentioned "*We found that our software is not as modular as we thought it was, therefore we are now rethinking our feature dependencies to make our components more modular to increase the reusability in our system*"
2. Explicitly modelling variabilities and commonalities triggered new potential variation points. As a result, more variability will be planned in the next version of the product.
3. Documenting and understanding the feature dependencies helps them in better defining their test scenarios, as the feature dependencies are reflected as module dependencies in the code. In their own words they mentioned *"understanding the feature dependencies already gives important information for building testing scenarios"*
4. Feature Assembly models help them better identifying the impact of change in features.
5. The system perspective provides a better view on the important features of their product, providing a different level of abstraction and understanding of their system.

The company also reported that Feature Assembly models will help them with understanding and managing the evolving variability of their product over time.

6.2 The Feature Assembly Framework

The evaluation exercise of the Feature Assembly Framework prototype performed by Antidot confirmed our hypothesis of the importance of the ability to unlock the knowledge contained in design models (i.e. Feature Assembly models). The team of Antidot confirmed that providing a visual navigation mechanism for inspecting the models was indeed useful. Furthermore, allowing users to visually interact with the Feature Assembly models is useful when tracing a certain feature for its relations or dependencies. In their case, they had some features that represented the backbone of their system and which they found

[4] This was their first use of a variability modeling technique and therefore we cannot compare the obtained results to a previous experience.

very useful to inspect using the prototype. This functionality is particularly important when more than one person is involved in the modelling (in their case three persons were involved). Also, they reported that being able to control the depth of display for a model during visualization is indeed useful for providing different levels of detail.

Furthermore, Antidot recommended adding some important meta-data to the information stored. For example, they recommended adding a description for each perspective and a definition date for the features. A definition date could also help them overcome the issue of lack of versioning support for the features mentioned in the previous section (we actually updated our prototype to include this and let them test it again).

Among the discussions we had was the discussion of the applicability of the Feature Assembly Framework [15] for reusing already existing features in the design of new products. Being a small size company their reuse schema was based on "opportunistic reuse" [17], i.e. the reuse of components and code at the implementation level. Reuse at a design level was not given much attention. Introducing them to the concept of "design with reuse" has actually led them to reconsider the independency of their features to enable more reuse opportunities. It was agreed that the power of considering reuse at the design level ("design with reuse") is that it promotes component reuse rather than code reuse and as such also enables "design for reuse". To achieve "design for reuse" the following guidelines were identified:

1. Identify which features are candidate standalone (i.e. consolidated and independent) features.
2. Analyse which of the feature dependencies are essential and should be enforced for these features when reused.
3. Improve the models such that the feature dependencies between standalone features are minimized.
4. Use the meta-data to describe these features, in order to be able to easy retrieve them later on, in particular by the use of tags. Restricting the tags to a specific set (e.g. using a predefined set of keywords) was not recommended, but rather a growing pool of tags was advised.

To enable "design with reuse" the following requirements were identified:

1. A good search mechanism is needed to identify already existing and reusable features.
2. The need to invest time in carefully modelling (existing) software features.

7. DISCUSSION

We can conclude that the work done during this evaluation, as well as the discussions held, confirmed the value of the presented approach; it also revealed interesting future work. The presented case study clearly answered our research questions stated earlier, the company clearly stated that they see added value in applying feature analysis and modelling to their product(s), this answers our first research question (RQ1).

The Feature Assembly Framework was also appreciated for providing an interactive medium for finding information about features in the Feature Assembly Models. For this to payoff, the company has to enforce a strict policy for adding meta information (e.g., feature description, feature keywords, stakeholders involved, customers who have this feature, etc.) and therefore making it available for later. From the discussions we had it was also clear that not all stakeholders need the same detailed level of information. For example, developers are interested in all levels of details for the modules they are responsible for, but for other modules they are only interested in the feature dependencies. It was clear that even this small company does have a need to unlock information implicitly available inside the company (RQ2).

The case study has also confronted us with the gap between industry and research in the domain of software variability. We started, as many other researchers, with an approach to be used when developing a new product line, however it turned out that in practice, small and medium scale companies acquire variability over time in their products and need mechanisms to deal with the variability of existing products or turn existing products into product lines. Although our approach was originally not designed for this purpose, it could however also be applied usefully in this context. Nevertheless, the experience indicated that (at least a part of the) research should be more driven by the challenges faced by the industry, and researchers should not try to push solutions for which there is no need in practice. Furthermore, the case study has confirmed the need to evaluate research prototypes in collaboration with industry to validate their effectiveness and to reveal additional needs.

The presented case study only provided a partial answer to our third research question considering feature reuse (RQ3). Feature Assembly modelling allows making more modular designs. Furthermore, the Feature Assembly Framework helps efficiently retrieve features for reuse. Therefore we may say that it increases the chances of successful reuse inside the company, therefore increasing the chances of reducing development cost. However, actual reuse can only be achieved while developing a new product. This has not been performed during the case study. Therefore, it was not possible to answer RQ3 with complete certainty.

The time taken by Antidot to learn to use the Feature Assembly Modelling technique was quite impressive. The company was also very positive on the ease of use and intuition of the modelling concepts and notations. They reported no problems with the understandability of the modelling semantics. The only negative issue mentioned was that sometimes it was not very obvious for them which feature dependency (we provide eight different dependencies) fits best to describe a certain situation. Also the notations used for the dependencies were sometimes difficult to remember. However, they found each one of the proposed dependencies essential. Therefore, we believe that this will improve with more practice of the technique. Although, we did not measure the effectiveness and usability in a quantitative way, the answer on our fourth research question (RQ4) is definitely positive.

Furthermore, research on software product lines focuses mostly on the architecture, implementation, and configuration levels. It is our opinion, and this is confirmed by our validation, that modelling issues concerning variable software are as important. In addition, there is a need for extending the research on variability with Information Management aspects to deal with growing amount of information needed for and about variable software. No matter how large or small the company teams are, tools that allow flexible information sharing are required.

8. THREATS TO VALIDITY

As we only validated the approach with one company, it may be possible that experiences in other companies could be different. However, the company was unknown to the researchers before the case study was started and the company also didn't have any

reason to favour the approach or the researchers. Therefore, we can state that the results obtained are rather objective.

The fact that the company is a small-scale company may have had an impact on the results.

As already mentioned, the company has not been using the concepts of variability modelling before, neither the concept of "feature" to describe their product capabilities. This may have affected the results in two different ways. First, introducing a new modelling technique may have introduced some learning time (which was indeed the case). Secondly, because Antidot has not used a variability modelling technique before they cannot compare the ease of use and expressiveness of Features Assembly to other feature modelling techniques.

The case study was done in a rather informal way, i.e. using meetings and discussion. We believe that this is justified for a first (pilot) validation case study, as the first purpose was to obtain as much spontaneous feedback as possible. In later case studies and experiments, a more rigorous approach will be used.

9. CONCLUSIONS AND FUTURE WORK

In this paper, we presented the results of an evaluation of our Feature Assembly approach performed by a small-scale software company. This evaluation was fruitful in many ways. Firstly, it gave us some insight on how a company works and on their challenges concerning managing the continuous growth and variation of their products. Secondly, the validation has clearly shown the importance of modelling software in terms of the composing features in order to better understand and identify the sources of complexity in the product. This is particularly important in products that contain variability or that acquire variability over time. We started this case study with some research questions in mind to help us evaluate our Feature Assembly Approach. Our questions have been answered, moreover, new issues were raised concerning the company's needs when modelling and managing the variability in their products (e.g., the need for explicit feature versioning). Most of these issues are worth further investigation.

For our future work, we seek applying the Feature Assembly approach to more industrial cases; this will certainly help improving the technique. It will also help us understand what meta-data is useful for unlocking information concerning features. Also the presented case study has pointed out the importance of tool support [11] that may go beyond simple proof of concept tools. We plan to develop a Feature Assembly modelling editor to help companies rapidly create their feature assembly models. Furthermore, the issue of feature versioning will be considered in future work.

10. ACKNOWLEDGMENTS

This research is sponsored by Innoviris, the Brussels institute for research and innovation (www.innoviris.be) through the VariBru project (www.varibru.be).

The authors like to thank Sebastien Le Grand and Frederic Arijs from Antidot for their valuable contribution during this case study. The authors also like to thank Tom Puttemans for his prototype implementation of the Feature Assembly Visualization tool.

11. REFERENCES

[1] K. Kang, S. Cohen, J. Hess, W. Novak, A. Peterson, "Feature-oriented domain analysis (FODA) feasibility study", Technical Report CMU/SEI-90-TR-021, Software Engineering Institute, Carnegie-Mellon University, 1990

[2] K.C. Kang, J. Lee, P. Donohoe, "Feature-Oriented Product Line Engineering", IEEE Software, vol. 19, no. 4, pp. 58-65, 2002

[3] A. Hubaux, , A. Classen, , M. Mendonca, , P. Heymans, A Preliminary Review on the Application of Feature Diagrams in Practice, In proccedings of VaMoS 2010 (VaMoS'10), Linz, Austria, January 27-29, University of Duisburg-Essen, pp. 53-59, 2010

[4] W. Codenie, N. González-Deleito, J. Deleu, V. Blagojevic, P. Kuvaja, J. Similä: A Model for Trading off Flexibility and Variability in Software Intensive Product Development, In proccedings of VaMoS 2009, ICB-Research Report, Universität Duisburg-Essen, pp. 61-70, 2009

[5] R. Van Ommering, J. Bosch, Widening the scope of software product lines — from variation to composition, In Proceedings of the Second International Conference on Software Product Lines (SPLC 2), Gary J. Chastek (Ed.). Springer-Verlag, London, UK, pp. 328-347, 2002

[6] B. Curtis, H. Krasner, N. Iscoe, A Field Study of the Software Design Process for Large Systems, Communications of the ACM, Volume 31 Issue 11, Nov. pp. 1268-1287, 1988

[7] D. M. Berry, K. Czarnecki, M. Antkiewicz, M. AbdelRazik, Requirements Determination is Unstoppable: An Experience Report, 18th IEEE International Conference on Requirements Engineering, pp. 311-316, 2010

[8] M. Svahnberg, J. van Gurp, J. Bosch, A Taxonomy of Variability Realization Techniques. in Software Practice & Experience, 35(8). pp.705-754, (2005).

[9] M. L. Griss, J. Favaro, M. d'Alessandro, Integrating Feature Modelling with the RSEB, Proc. Fifth International Conference on Software Reuse, pp. 76-85, Victoria, BC, Canada , 1998.

[10] L. Abo Zaid, F. Kleinermann, O. De Troyer: Feature Assembly: A New Feature Modelling Technique, In : 29th International Conference on Conceptual Modelling, Lecture Notes in Computer Science, 2010, Volume 6412/2010, Springer-Verlag, pp. 233-246

[11] M. El Dammagh, O. De Troyer, Feature Modelling Tools: Evaluation and Lessons Learned, ER Workshops 2011 pp. 120-129, 2011

[12] D. Nestor, S. Thiel, G. Botterweck, C. Cawley, P. Healy: Applying visualisation techniques in software product lines. SOFTVIS 2008: 175-184

[13] H. Wang, Y. Li, J. Sun, H. Zhang, J. Pan, A semantic web approach to feature Modelling and verification. In: Proceedings of Workshop on Semantic Web Enabled Software Engineering (SWESE'05), 2005

[14] L. Abo Zaid, F. Kleinermann, O. De Troyer, Applying Semantic Web Technology to Feature Modelling. In: The 24th Annual ACM Symposium on Applied Computing, The Semantic Web and Applications (SWA) Track march 2009.

[15] L.Zaid, F. Kleinermann, O. De Troyer, Feature Assembly Framework: towards scalable and reusable feature models, In Proceedings of the 5th Workshop on Variability Modelling

of Software-Intensive Systems (VaMoS '11). ACM, New York, NY, USA, pp. 1-9, 2011

[16] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-oriented Software, Addison-Wesley, 1995.

[17] R. van Ommering, Software Reuse in Product Populations, IEEE Trans. Software Eng., vol. 31, no. 7, 2005, pp. 537–550.