Software Product Line Variability: A Systematic Mapping Study

Shahid Mujtaba¹,², Kai Petersen¹,², Robert Feldt¹, Michael Mattsson¹
¹School of Engineering, Blekinge Institute of Technology
Box 520 SE-372 25 Ronneby, Sweden
{shahid.mujtaba | kai.petersen | robert.feldt | michael.mattsson}@bth.se
²Ericsson AB, Box 518 SE-371 23 Karlskrona, Sweden
{shahid.mujtaba | kai.petersen}@ericsson.se

Abstract

Software product line engineering (SPLE) enables software mass customization and allows development costs to be cut, lead delivery times to be reduced and throughput to be optimized. The concept of variability is central in SPLE methodologies and for success when adopting a software product line approach. Even though much research has been done on this topic there is no clear view of the current state of knowledge. In this study, we used a systematic method to develop a software product line variability map and classified relevant literature. The resulting map gives an overview of the existing work on software product line variability and makes it possible to identify gaps and plan and position future research.

1. Introduction

Software product lines is a development paradigm which allows software companies to build customized software and get a shorter time to market and higher quality through systematic reuse [16]. In order to achieve systematic reuse, product line engineering relies on two core concepts, namely domain engineering and application engineering [89]. In domain engineering, a product line platform is created, which then becomes the base from which an actual product can be developed. In application engineering, the actual derivation of the products of the product line takes place. A core concept for both domain and application engineering is software product line variability. So far, no comprehensive birds eye view on existing research focusing specifically on product line variability literature has been published. To address this, we have conducted a systematic mapping study of existing research in product line variability.

In particular, following contributions are made in this paper: Firstly, we apply a systematic approach for creating a map of product line variability research in order to identify 1) which areas in software product line research have been emphasized, 2) which areas are not well covered and thus require further research, and to classify 3) which types of research have been conducted and 4) which types of contributions in terms of method, process, tools etc. have been made. Secondly, we provide one example of how to adapt systematic mapping studies to the software engineering domain. So far, only one short paper has been published that piloted systematic mapping in software engineering [7]. This is beneficial as future variability research can be easily positioned within the map and relevant related work is easily identified. Furthermore, the research method is described in detail, allowing researchers to update the map with very little effort and to conduct mapping studies of other software product line or software engineering research areas.

2. Research Methodology

The research methodology used is a Systematic Mapping Study. We have provided a detailed description of the mapping process, outlined the methodology and compared systematic maps with systematic reviews in another study [63]. In this study, we present the results of applying systematic mapping to software product line engineering domain. We aim to answer the following research questions: RQ1: What areas in software product line variability have been addressed and how many articles cover the different areas? RQ2: What types of papers are published in the area and in particular what type of evaluation and novelty do they constitute?

2.1 Search Strategy and Screening

The search string used to identify literature for the mapping study is defined through which population of papers
we target and the intervention they concern. The population comprises the software domain, product lines, product families and system families. The intervention is concerned with variability and variation handling in product lines. The resulting search string is: Software AND (Product Line OR Product Family OR System Family) AND (Variability OR Variation)

We applied the search string on all Software Product Line Conferences (SPLC) and Product Family Engineering Workshops (PFE) published between (2000-2007). To further strengthen results we included top international, peer-reviewed journals published in the last ten years (1997-2007). The databases, we searched were Inspec and Compendex, ACM Digital Library and IEEE Xplore [22]. In total, 153 papers were identified. The inclusion and exclusion criteria were pilot-tested by three of the authors on a random sample of 20 papers and a high agreement was achieved. After the pilot, two of the authors screened the remaining papers by skimming through the abstracts and marked them as included, excluded or uncertain. In the end, the authors reviewed all disagreements by looking at the abstracts together and conflicts were resolved through discussion. After the screening process a total of 91 relevant papers remained.

2.2 Establishing a Classification Scheme

A few classifications of variability have been done in the past. However, they are not based on an overview of the literature, but on industrial case studies (e.g., [77, 92]). As these classifications are not based on a broad sample of product line literature, there is a risk that some important categories are missing. Thus, we followed a keywording strategy to develop our own classification scheme. The keywording process is shown in Figure 1. It is done in two steps. First, the reviewers read abstracts and looked for keywords and concepts that reflect the contribution of the paper. While doing so the reviewer also identified the context of the research. When this is done, the set of keywords from different papers were clustered and combined together to develop a high level understanding about the nature and contribution of the research and to form the categories for the map. This helped the reviewers to come up with a set of categories which is representative of the underlying population. The resulting classification scheme is presented in Section 3. The abstracts which were of too poor quality to allow meaningful keywords to be extracted, reviewers choose to study the introduction or conclusion sections of the paper.

2.3 Mapping of Studies and Analysis

In this phase each included article was mapped to a particular intersection set within the scheme. The process was driven by the definitions of the categories identified during the development of the classification scheme described in Section 3. The map produced a clear picture of how past research is scattered across multiple facets. It helped us to purposefully analyse the results while focusing on the frequencies of publications, thereby determining the coverage of the research field. Different facets of the scheme were combined to answer specific research questions.

3. Classification Scheme

Papers are classified based on three different facets. Each facet consists of a set of categories to which the papers can be mapped. The facets are variability context, contribution type and research type. Tables 1, 2 and 3 detail the classification categories we have used.

Variability context: Which type of activity in software development variability does the paper focus on? Through the keywording process, we identified the categories requirements, architecture, implementation, verification and validation, management, and orthogonal variability.

Contribution type: What was developed in order to achieve advances in the focus area? Possible contributions are processes, methods, models, metric and tool. Those categories were also developed during the key-wording process.

Research type: What is the novelty level and how the research is characterized? Based on an existing classification proposed by Wieringa et al. [90] we categorize the papers into validation research, evaluation research, solution proposal, philosophical papers, opinion papers, and experience papers.

4. Mapping

This section describes the result of our mapping within the variability context facet and we exemplify typical papers for each category. Then we describe the combinations of the variability context facet with the contribution type facet and research type facet. The complete map of software product line variability is shown in Figure 2. The mapping of papers
Table 1. Variability Context Categories

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td>In this category, variability in requirements is discussed. Furthermore, binding of variability on the requirements level is included.</td>
</tr>
<tr>
<td>Architecture</td>
<td>This category includes articles considering variability in software architecture construction and modeling. This includes binding of architecture variability.</td>
</tr>
<tr>
<td>Implementation</td>
<td>This category is concerned with literature on how to incorporate variability within the implementation (source code). This includes binding in implementation.</td>
</tr>
<tr>
<td>Verification</td>
<td>This category includes quality assurance methods considering variability. This includes testing as well as formal verification and consistency checking techniques.</td>
</tr>
<tr>
<td>Management</td>
<td>This area is mainly concerned with managing requirements and dependencies at all abstraction levels keeping track of versions (configuration management, variability in time), managing traceability and architecture evolution</td>
</tr>
<tr>
<td>Orthogonal Variability</td>
<td>Orthogonal variability is a concept that considers variability independently of any development artifacts. This implies that a separate model for variability is maintained, and its variants and variation points are traced to all other development artifacts.</td>
</tr>
</tbody>
</table>

to the facet Variability context shows that a clear majority of variability research is concerned with requirements and architecture.

Requirements: For the requirement part, a large amount of literature was concerned with modeling variability in the form of feature models, which represents the main requirements artifact in software product line engineering [18]. Furthermore, research focused on classifying dependencies and binding features in a consistent way, i.e., considering dependencies between features so that only consistent combinations of product features are selected [87, 53].

Architecture: On the architecture level, product line research produced results regarding processes for architecture creation. For example, [79] proposed a design process for preparation, modeling and evaluating architectures. Furthermore, the modeling of architecture has to consider variability, which for example was addressed by [95] who extended UML models for architecture development. Besides this, several architecture papers described architecture patterns [49, 31]. In order to make the complexity of architecture manageable and to tailor the description of architectures to specific stakeholders, different architecture views are proposed in [1]. In addition to that, the variability on architecture level has to be bound in order to derive a product architecture from the product line platform (as is the case on the feature level). In order to achieve this, [32] for instance provided mechanism for binding on architecture level.

Implementation: When looking at the implementation, [60] proposes transforming models into text representing source code, for example by transforming different associations for class diagrams into text). Furthermore, they describe how high-order transformation can be used to resolve variability and generate code for domain artifacts. In addition to code generation, aspect orientation and feature oriented analysis are combined to capture crosscutting concerns in the implementation [46]. Furthermore, generative programming is an important concept when implementing product lines allowing to generate source code for specific products, for example by using compiler flags [54].

Verification and Validation (V&V): V&V is done for requirements, architecture and implementation. For example, [86] discuss to derive test cases from use cases containing variability to verify the actual implementation. Requirements verification for feature models is for example done by checking if selected features fulfill logical expressions representing feature models [52, 53]. Furthermore, tool support (RequiLine) is provided which allows to check whether variation point resolutions in feature models are free of conflict [86]. On the architecture, we discovered no verification techniques for product line architectures.
Table 3. Research Type Facet

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation Research</td>
<td>Techniques investigated are novel and have not yet been implemented in practice. Techniques used are for example experiments, i.e., work done in the lab.</td>
</tr>
<tr>
<td>Evaluation Research</td>
<td>Techniques are implemented in practice and an evaluation of the technique is conducted i.e., it is shown how the technique is implemented in practice (solution implementation) and what are the consequences of the implementation in terms of benefits and drawbacks (implementation evaluation). This also includes to identify problems in industry.</td>
</tr>
<tr>
<td>Solution Proposal</td>
<td>A solution for a problem is proposed, the solution can be either novel or significant extension of an existing technique. The potential benefits and the applicability of the solution is shown by an example or a good line of argumentation.</td>
</tr>
<tr>
<td>Philosophical Papers</td>
<td>These papers sketch a new way of looking at existing things by structuring the field in form of a taxonomy or conceptual framework.</td>
</tr>
<tr>
<td>Opinion Papers</td>
<td>These papers express the personal opinion of somebody whether a certain technique is good or bad, or how things should been done. They do not rely on related work and research methodologies.</td>
</tr>
<tr>
<td>Experience Papers</td>
<td>What and how something has been done in practice. It has to be the personal experience of the author.</td>
</tr>
</tbody>
</table>

Management: Management is concerned with keeping track of versions (configuration management) and traceability. The traceability aspect is discussed in [10] where traces between solution space (architecture, design and implementation) and problem space (requirements and domain analysis) are recorded. Furthermore, the requirements management tool RequLine allows to manage version of requirements, support prioritization and achieve pre-traceability. Furthermore, features can be linked to other requirements types.

After having provided an isolated overview of the variability context facet, we combine the different facets with each other, resulting in a map which for example shows how much evaluation research has been done in comparison to validation research. Figure 2 shows an overview of the Software Product Line Variability Map (SPL-V Map). The bubbles show how many publications are identified for each interaction set of the variability context facet with evaluation facet and contribution facet. Also, it is important to note that the total number of papers on either side of the map are not equal to each other. This is because few papers are mapped to multiple categories in a single facet i.e. a paper concerning with requirements variability context could present a model as well as the tool support to build that model. The map shows that solution proposals (56 papers) and evaluation research (50 papers) are the most dominant types in the research type with no validation research done in the area so far. We will further discuss the reasons for this in Section 5. The solution proposals are mainly concerned with methods and models. A large amount of literature is devised on domain analysis techniques and methodologies to handle product line variations using feature modeling [5, 20] which include literature on staged configurations [17, 18, 19], feature binding [14, 43] and meta-modeling [26]. There are some proposed mechanisms to create architectures that support dynamic instantiation and configuration of product line components [29, 44, 82]. Similarly, the research which constitutes evaluations mainly assessed methods and tools with relatively less work addressing implementation or other variability categories. Despite the notion that a considerable amount of research is characterized as empirical mainly because of their industrial context, a close look reveals that the evaluations are largely based on industrial examples and may not be truly recognized as techniques which are actually practiced in the industry. Several such papers reported case studies such as [31, 69, 85] which lack solid research design and validation.

5. Discussion

Evaluation, Validation and Empirical Evidence: The second largest set of articles relates to evaluation research (39.06%). As discussed in the previous section, it indicates that a high volume of research is claimed to be implemented and evaluated in industrial practice. For example, [3] implemented a multi-view architectural modeling approach and illustrated it by showing models from the medical domain. Other articles followed the same pattern of illustrating their industrial solution implementation while varying strongly in detail. A very detailed illustration of a solution of how to arrive at domain requirements is illustrated in [57]. As pointed out by Wieringa et al. [90] an important part of evaluation research is to apply sound research methods. In order to judge this, a detailed description of the research methodology is required. We only identified one paper that clearly described the data collection process and threats to validity [65]. In order to be able to establish the current state of evidence for product line variability research, we emphasize the importance of improving on the use empirical research methods and reporting procedures in product line variability research. Besides evaluation research, the highest amount of papers uses solution proposals and works with examples as proof of concepts (43.75%). We identified varying degrees of detail in the examples, ranging from very
superficial to more detailed ones. One possible reason for the high number of solution proposals is that introducing a product line is a costly decision which affects large parts of a company. That is, without the proper industrial setup new solution proposals cannot be evaluated properly. Furthermore, it is conspicuous that no validation research has been done at all, one reason being that it is hard to scale software product line problems to the lab. The map also shows that the highest amount of evaluation research and solution proposals is related to requirements and architecture. However, this is due to that those categories comprise the highest amount of articles.

**Emphasis on Models and Methods:** The map shows that there is a clear emphasis on models (35.59%) and methods (38.98%) to support product line variability. For example, on the requirements level the highest amount of papers discuss of models (14.41%). The models that are most frequently referred to are feature models. An industry evaluation of feature models has shown that there is still some improvement needed [33]. The major improvement needed from an industrial perspective is the improvement of cross-referencing and dependency modeling between features. Furthermore, features are essential during product derivation (binding). Methods relate to feature modeling in the way that they propose methods of how to derive products specific models from domain models (like feature models), see for example [17]. From the coverage perspective, we see that architecture variability is covered by process support. When looking at the areas of the map that are covered by a very small number of papers, it becomes apparent that there is a lack of research on tools and metrics.

**Few Philosophical Papers and Experience Reports:** The map shows relatively few philosophical papers (10.9%) and experience reports (6.25%). It is important for the product line community to urge people to share more knowledge and experience. Both philosophical and experience papers tend to make important contributions to the field through providing conceptual basis or highlighting key issues. Few papers tried to discuss and categorize product line variability taking into account various perspectives. For instance, classification of product families [92], realization and modeling techniques [77, 72] and variant binding [37].

6. Conclusions

Our systematic map shows that the majority of research has been on requirements and architectural variability and with a focus on developing models and methods. There is relatively few papers on processes, tools and metrics and relating to other variability contexts than requirements and architecture. The main type of research has been solution proposals, closely followed by evaluation research involving a practical evaluation of the proposed, or existing, contribution. However, only one evaluation paper clearly describe the data collection process and validity threats.

Based on our map there seems to be a relative lack of research on how to support SPL variability in implementation and management as well as how to verify and validate it. This can be a fruitful future area for more research. We also see a need for better empirical research methods, using case study methodology and experiments on isolated parts of SPL variability solutions. The way empirical research is reported also needs to be strengthened. The classification scheme we have developed can help in this regard.

References


![Figure 2. SPL-V Map](image-url)
K. Berg, J. Bishop, and D. Muthig. Tracing software product
A. Braganc¸a and R. J. Machado. Extending uml 2.0 meta-
A. Jansen, R. Smedinga, J. van Gurp, and J. Bosch. First
feature-oriented analysis and aspect-oriented programming
in software product lines. In Proc. of the 5th International
Workshop on Software Product-Family Engineering (PFE 2003),

M. Jaring and J. Bosch. Representing variability in software

M. Jaring and J. Bosch. Variability dependencies in product
family engineering. In Proc. of the 5th International Work-
shop on Software Product-Family Engineering (PFE 2003),

M. Jaring, R. L. Krikhaar, and J. Bosch. Representing vari-
ability in a family of mri scanners. Softw., Pract. Exper.,

E. Kamsties, K. Pohl, S. Reis, and A. Reuys. Testing vari-
abilities in use case models. In Proc. of the 5th Interna-
tional Workshop on Software Product-Family Engineering

B. Keepence and M. Mannion. Using patterns to model vari-
ability in product families. IEEE Software, 16(4):102–108,
1999.

T. Kishi and N. Noda. Formal verification and software

T. Kishi, N. Noda, and T. Katayama. Design verification for
product line development. In Proc. of the 9th Int. Conf. on

C. W. Krueger. Variation management for software produc-
tion lines. In Proc. of the Second Int. Conf. on Software

J. Lee and K. C. Kang. Feature binding analysis for product
line component development. In Proc. of the 5th Interna-
tional Workshop on Software Product-Family Engineering

J. Lee and K. C. Kang. A feature-oriented approach to develop-
ing dynamically reconfigurable products in product line
engineering. In Proc. of the 10th Int. Conf. on Software

J. Lee and D. Muthig. Feature-oriented variability manage-
ment in product line engineering. Commun. ACM, 49(12):55–59,
2006.

K. Lee, K. C. Kang, M. Kim, and S. Park. Combining
feature-oriented analysis and aspect-oriented programming
for product line asset development. In Proc. of the 10th Int.
Conf. on Software Product Lines (SPLC 2006), pages 103–

J. Liu, J. Dehlinger, and R. R. Lutz. Safety analysis of soft-
ware product lines using state-based modeling. Journal of

F. Loesch and E. Ploedereder. Optimization of variability
in software product lines. In Proc. of the 11th Int. Conf. on

R. R. Lutz. Extending the product family approach to sup-

A. Maccari and A. Heie. Managing infinite variability in
537, 2005.

A. Maccari and C. Riva. Architectural evolution of legacy
product families. In Proc. of the 4th International Workshop
on Software Product-Family Engineering (PFE 2001), pages

M. Mannion. Using first-order logic for product line model
validation. In Proc. of the Second Int. Conf. on Software

M. Mannion and J. Camara. Theorem proving for product
line model verification. In Proc. of the 5th International Work-
shop on Software Product-Family Engineering (PFE 2003),

I. McRitchie, T. J. Brown, and I. T. A. Spence. Manag-
ing component variability within embedded software prod-
uct lines via transformational code generation. In Proc. of
the 5th International Workshop on Software Product-Family

K. Mohan and B. Ramesh. Tracing variations in software

M. Moon and Y. Keunhyuk. An approach to develop require-
ment as a core asset in product-line. IEICE transactions on

M. Moon, K. Yeom, and H. S. Chae. An approach to develop-
ing domain requirements as a core asset based on com-
monality and variability analysis in a product line. IEEE

H. Muccini and A. van der Hoek. Towards testing product
2003.

A. Oberweis, V. Pankratius, and W. Stucky. Product lines
for digital information products. Inf. Syst., 32(6):909–939,
2007.

J. Oldevik and O. Haugen. Higher-order transformations for
product lines. In Proc. of the 11th Int. Conf. on Software

F. G. Olumofin and V. B. Misic. A holistic architecture as-
sessment method for software product lines. Information &

J. Pérez, M. A. Laguna, Y. C. González-Carvajal, and
B. González-Baixauli. Requirements variability support
through mda\textsuperscript{tm} and graph transformation. Electr. Notes

K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson. Sys-
of the 2008 annual research conference on Evaluation and
Assessment in Software Engineering (Accepted).


