Universidade Federal de Pernambuco
Centro de Informática

Bacharelado em Ciência da Computação

Applying Systematic Reviews on Requirements Variability Models for Software Product Lines: An Experience Report

Paola Accioly

Trabalho de Graduação

Recife
03 de dezembro de 2009

Universidade Federal de Pernambuco
Centro de Informática

Paola Accioly

Applying Systematic Reviews on Requirements Variability Models for Software Product Lines: An Experience Report

Trabalho apresentado ao Programa de Bacharelado em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: *Paulo Henrique Monteiro Borba*
Co-orientador: *Rodrigo Bonifácio*

Recife
03 de dezembro de 2009

Acknowledgements

Gostaria de agradecer a minha família e amigos que sempre me deram apoio irrestrito nas minhas decisões e aos meus professores, orientador e co-orientador pela ajuda na elaboração deste trabalho.

"My mama always said: life is like a box of chocolates. You never know what you're gonna get."

—FORREST GUMP (Personagem interpretado por Tom Hanks no filme Forrest Gump, o contador de histórias)

Abstract

Software Product Line Engineering (SPL) is a development approach that implements a family of systems instead of a single system. Some of the benefits of SPL approach are: reduction of time-to-market, reduction of maintenance effort, improvement of cost estimation and indirectly enhancement of products quality [45]

Managing commonality and variability across the product line is the main concept of SPL engineering. From design to implementation, all artifacts, including requirements artifacts, produced on domain level need the ability of representing variability. Many researchers have been proposing different techniques to model requirements variability for SPL. Some examples of these techniques are PLUC [5], VML4RE [2] and MSVCM [10]. On the other hand, there has not been yet to much effort on mapping all these techniques and comparing their main characteristics.

The present work aims to:

- Present a literature review that identifies and compares different requirement variability management techniques. This work will be done following the Systematic Review and EBSE approaches proposed by [34, 18];
- Give some guidelines of how to conduct systematic reviews and evaluate the process of executing this process;

Our results concluded that the research field of requirements approaches for SPL is still poor of methods and metrics for evaluating and comparing different techniques. With few exceptions, there has not been effort on producing papers that experimented and analyzed these techniques using more formal methods.

Keywords: Software Product Line Engineering, Requirements Variability, Systematic Reviews

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Software Product Line Engineering (SPL) is a development approach that implements a family of systems instead of a single system. Some of the benefits of SPL approach are: reduction of time-to-market, reduction of maintenance effort, improvement of cost estimation and indirectly enhancement of products quality [45].

Systems from the same family have many features in common but with some variations from one system to another. The engineering paradigm of SPL is divided in two levels: the domain level and the application level. The domain level engineering is where SPL commonality and variability is defined and implemented, is the platform that generates all the products from the same line. The application level is where each product derived from the product line is generated by the reuse of the domain artifacts.

Managing commonality and variability across the product line is the main concept of SPL engineering. From design to implementation, all artifacts, including requirements artifacts, produced on domain level need the ability of representing variability. Feature models are a visual representation of common and variable features of a SPL. According to Kang [31], feature models could be used to represent requirements of the product lines. Figure 1.1 show an example of feature models. At this figure we can see that feature models can describe which features are mandatory, optional or alternative.

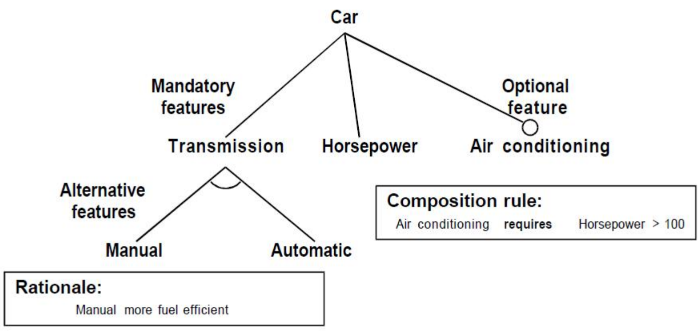

Figure 1.1 Feature Model. Figure extracted from [31].

Since features can embrace a set of functional and non-functional requirements, it is hard to map from feature models to a product's requirements artifact. According to Bühne [7] feature models are not sufficient to model system's requirements.

In order to fill this gap, many researchers have been proposing different techniques to model requirements variability for SPL. Some examples of these techniques are PLUC [5], VML4RE [2] and MSVCM [10]. On the other hand, there has not been yet to much effort on mapping all these techniques and comparing their main characteristics. This work has the objective to investigate what are the existent techniques to model requirements variability for SPL and learn about the the status of existent evaluation methods for analyzing and comparing them.

The methodology that guides our investigation and analysis of requirements variability management approaches is the systematic review's method, the main tool of the evidence-based software engineering (EBSE) proposed by Kitchenham [34]. Next section discusses this work objectives.

1.1 Work Objectives

The present work aims to:

- Present a literature review that identifies and compares different requirement variability management techniques. This work will be done following the Systematic Review and EBSE approaches proposed by [34, 18];
- Give some guidelines of how to conduct systematic reviews and evaluate the process of executing this process;

The rest of this document is structured as follows: Chapter 2 describes evidence-based software engineering and systematic review's process. The approach the we used to execute the systematic review in this work is described on Chapter 3. Chapter 4 discusses the result achieved by the systematic review's execution and, finally, Chapter 5 concludes this works and suggests future works.

CHAPTER 2
Systematic Reviews

Evidence-based practice has its origins in medical research from the early 90's and it was best defined by Sacket as a means to improve patients healthcare by integrating the best evidence from empirical studies together with internal practical experience [47]. Medicine was the first discipline to embrace Evidence-based practice to extract and analyze common guidelines from multiple published studies, synthesize the best practices and apply them on a specific environment.

To put together and analyze the best evidence from empirical researches, evidence-based practitioners have developed a rigorous method called Systematic Review (SR). Systematic Review is a method that collects, analyze and extract the state-of-the-art from literature. Systematic Reviews can be used to achieve answers to a research question, to obtain the state-of-the-art before starting a new project or simply to learn good practices and keep up with the new evidences on a specific area.

Medicine's successful adoption of evidence-based practice has prompted many other disciplines to adapt this approach to their specific needs. Thats why, in 2004, Kitchenham [34] proposed the idea of a Evidence-based Software Engineering (EBSE) that aims to enhance services provided by software engineering by gathering empirical research evidences.

The main difference between a regular survey and a systematic review is the definition of the review protocol. The review protocol anticipates how each phase of the systematic review will be executed. More refined protocols tend to provide more reliable evidence and that is exactly the goal of EBSE. Chapter 3 will describe what are the phases of the review protocol and how the protocol was defined and executed on this work.

Since 2004, Systematic Reviews begin to be used by software engineering researchers and practitioners to collect the best studies available from a certain domain, analyze them using rigorous methods and extract data that summarizes best evidences acquired in that area [18].

2.1 Some Examples of Systematic Reviews

Since the proposal of EBSE, there's has been a good number of published surveys using systematic reviews method coming from different Software Engineering subareas, some examples that we'll discuss next are [17, 19, 13].

In 2007, Dias Neto [17] published a systematic review of Model-based Testing Approaches (MBT) for software development aiming to "provide a range of options towards the selection of an MBT approach to a test practitioner for a given project". During his work it was possible to analyze characteristics considered important to test engineering researchers like efficiency and coverage of model-based approaches and how much effort is done in order to implement these techniques on real software projects.

In 2008, Dyba [19] published a systematic review of agile software development methods. This work analyzed 36 published papers about agile methods and the synthesis of its findings led to a discussion about what is currently known about agile methods, their benefits and limitations and how strong were the evidences discussed.

In 2009, Chen [13] conducted a systematic review with the aim to “review the status of evaluation of Variability Management (VM) approaches in Software Product Line Engineering (SPLE)”. Chen’s work objective is similar to our systematic review objectives since both of them discuss variability management approaches for software product line engineering. But Chen’s work embraces all kinds of techniques to manage variability and our work focus only on requirements techniques for SPL. After analyzing 97 papers, Chen concluded that “only a small portion of the identified approaches were evaluated using rigorous scientific methods” and that scientific evaluation methods for VM approaches were still precarious.

As we can see, systematic reviews can be applied to diverse study types and in this work we aim to conduct the execution of a systematic review process, making some observations and evaluating the process as a whole. Chapter 3 will describe the approach followed on this work to execute the systematic review.

CHAPTER 3

An Approach for Conducting a Systematic Review

The traditional process of conducting Systematic Reviews, originally proposed by Kitchenham [34] is like a cascade model typically divided in three phases:planning, conducting and reporting the review. Figure 3.1 shows a flow chart describing the whole process.

At the planning phase the need for the research is identified, that is, the problem that led to the execution of the systematic review is documented together with the definition of the research questions that will guide the objectives of the Systematic Review. After defining the research questions, the protocol is defined. The review protocol describes in advance the methodology that will be used during the execution phase. After that all protocol phases are defined, the systematic review can be executed and its results can be reported.

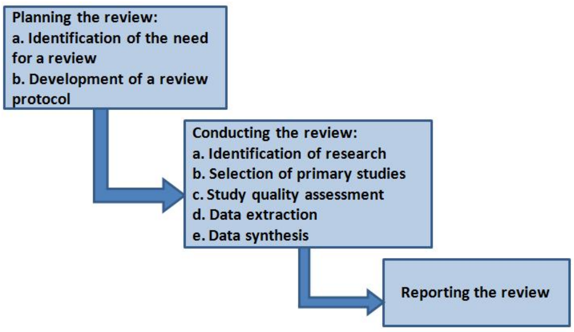

Figure 3.1 Traditional Systematic Review process as a cascade model.

Unlike the traditional cascade model, Dyba[18] recommended to execute pilot parts of the review during the planning phase. Following this recommendation, and improving it, we have decided to use an iterative model to run our systematic review. This way, after defining each step of the protocol, we could execute it and perform a validation. For example, after defining the search strategy, we performed the search and analyzed its results, if we found that the results were not comprehensive enough, we could go back to the planning and change the search criteria. The decision of changing the traditional model to an iterative one was really useful since we could anticipate many problems and correct them right away. Figure 3.2 shows the process that guided our work.

Next sections describe each phase of the review's protocol and how each of them was conducted in our work.

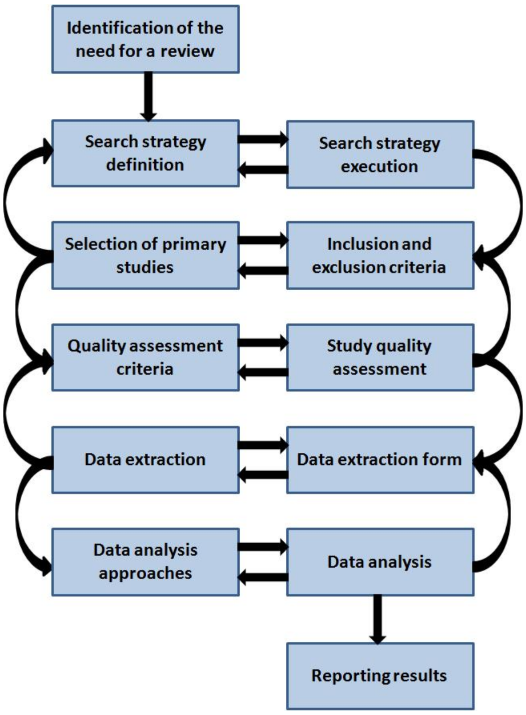

Figure 3.2 Systematic Review Process and an Iterative Model.

3.1 Identification of the need for a review

The identification of the need for a review is the beginning of systematic review's process. Here the objectives of the review are defined and the research questions are formulated. As it was discussed on Chapter 1, many approaches to model requirements for software product lines have been proposed but yet there has not been a comprehensive analysis of them.

Proposed techniques use completely different notations and tools to model different types of requirements. The main objective of this systematic review is to draw a map of existent approaches to manage requirements variability on software product lines. This map will help to learn more about the existent techniques, their main characteristics and how these methods are being proposed.

Considering the main objective described above, 4 research questions where formulated. These questions will serve as guidelines to the next phases of the review.

1. What are the Requirements Variability Management Techniques available?

2. Where do these techniques come from? In what context are they being used?

3. What are their main characteristics?

4. How is this type of research being conducted? What methods are being used to experiment, analyze and compare the techniques?

3.2 Definition of the Search Strategy

The first phase of the review protocol is the development of the search strategy that defines keywords, main search terms and a list of bibliographic databases that will be used during the search phase. According to Dyba [18], the definition of the search strategy is the moment to strike a balance between precision and comprehensiveness to the results that the specific review needs to reach. In this work we have targeted comprehensiveness since we wanted to reach as many techniques as possible.

After executing the search strategy some times and analyzing its results, some keywords like Model Driven and Metamodel were added to increase the comprehensiveness of our search. The final list of search terms is the following:

- Variability AND Management;
- Requirements AND Variability;
- Requirements AND (Software Product Lines OR Product Lines);
- Requirements AND (Product Family OR Software Product Family).
- Models And Software PL
- Managing Requirements Variability
- Model Driven
- Product Line Requirements
- Modelling Requirements
- Metamodel

To perform the research, the following electronic databases were chosen:

- Periodicos CAPES
- ACM Digital Library
- Citeseer
- IEEE Xplore
- ScienceDirect
- Google Scholar

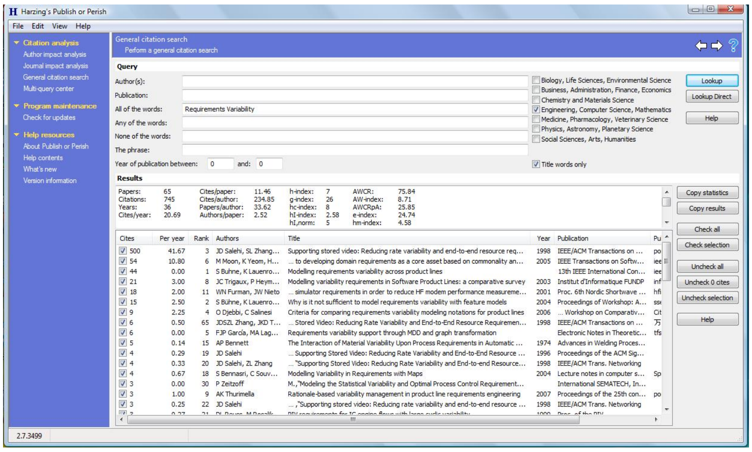

Figure 3.3 Graphical user interface of the *Publish or Perish* tool.

In addition, a software called *Publish or Perish* was very useful at this phase of research. *Publish or Perish* [27] it's a free software that was designed for retrieving information about academic citations using Google Scholar as its data source. Publish and Perish software was used as a means to "check" if all relevant articles related to the search were retrieved by the defined search criteria.

Publish or Perish was also used to retrieve the number of citations of each paper. Figure 3.3 shows a picture of Publish or Perish main interface which provides great filters to get the exact information needed.

After exploring the databases using the search terms combined in different ways, a total of 102 papers were retrieved. At this phase there was no selection criteria since we wanted to get as many papers as possible. The next phase of the review defines some criteria by which the paper will get included or not into the research.

3.3 Selection of Primary Studies

Now that the search strategy is complete, the next step of the review is the selection of primary studies. After executing the search strategy there is a need to analyze all the publications retrieved and choose the ones that match with the review's objectives. Inclusion and exclusion criteria must be defined in order to select the most adequate publications for the review.

At this phase the abstract of all retrieved papers were analyzed and the ones we have included matched with at least one of the following criteria:

- Proposed a new approach to manage requirements variability;
- Described empirical data on requirements variability management techniques;
- Proposed an extension to an already existent approach;
- Made experiments using one, two or more techniques;

- Compared one, two or more techniques;

The excluded studies were the ones that:

- Discussed variability management techniques concerning other aspects of software product lines approach, like feature models, code variation points, tests, architectures, etc;
- Articles that discussed SPL requirements as feature models;
- Articles that were too short (2 or 3 pages).

After analyzing the abstract of all 102 papers, a total of 40 studies were filled our inclusion criteria and were selected to the review's next phase.

3.4 Quality Assessment

After selecting the primary studies using the inclusion and exclusion criteria you need to know more about the selected studies. More specifically, there is the need to investigate certain aspects of the publication in order to analyze its quality. In order to assess the quality of the studies, we first classified them into the following categories:

- Category [A]: A study that proposed a new requirements variability technique;
- Category [B]: A literature survey or a study that discussed an experiment/case study using a single technique or comparing two or more techniques.

Works that proposed a new technique but also described a literature survey, case study or a comparison between techiques Figure 3.4 shows the results after dividing the 40 studies in both categories. Therefore, most works (35) proposed a new technique and only 5 papers were surveys/case study of already existent approaches.

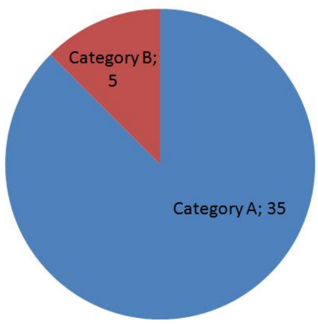

Figure 3.4 Graphic showing A versus B papers.

To analyze the studies from category [A] the following questions were defined:

- QA1. Did the study propose a totally new technique or just modified/extended an already existent one?
- QA2. How did the author(s) justify the need for the new technique? Did he make a literature review point some limitations?
- QA3. Did the author(s) analyze the new technique after introducing it? If yes, how was it done? Was there a comparison with QA4.other already existent approaches? Were there metrics?

To analyze category [B] the following question was defined:

- QB1. How many techniques were mentioned?
- QB2. Did the study reached the state-of-the-art?
- QB3. What method the article used to analyze the technique(s)?
- QB4. Were there metrics to compare the techniques? Which ones?

Using the questions described above, a quality assessment table was defined and, in order to extract the required information, each paper was analyzed by hand without any tool support. With almost all studies, reading the abstract and the introduction was sufficient to learn the method that guided the author(s) during their research. Quality assessment phase do not exclude papers from the review. The results of quality assessment phase will be shown and discussed on next Chapter.

3.5 Data Extraction

During this stage, data needs to be collected from the selected studies. The research questions will guide the task of defining what characteristics to extract from the selected papers. In this work data was manually extracted from each paper according to the following extraction forms.

Papers characteristics, applied to all categories of articles:

- Authors;
- Institution(s) that published the paper;
- Nationality (institution's nationalities);
- Year of publication;
- How many times it was cited;
- In what context it was used? (Academy, industry, government).

Techniques characteristics, applied only to category (A):

- Method's name;
- Brief Method Description
- Type of requirement that can be modeled (functional, not-functional, both);
- How requirements are being represented? (UML diagrams, Goals, scenarios description, activity diagram, etc);
- What element is used to model variation point (stereotypes, tags, aspects, etc);
- What notation is being used? (Graphical, textual);
- Is there a support tool? In case it does, is it free or paid?
- Variation mechanism (annotation-based or composition-based [32]);

Additional to (B) category:

- What are the paper's objective?
- Publication context (academy, industry, etc).

Next subsections discuss the meaning of some items present on the extraction form. The results from data extraction phase will be compiled and discussed next chapter.

3.5.1 Variation Mechanism

The item "Variation mechanism" from the extraction form was defined by Kästner at [32] where he discusses that there are mainly two ways to model variability on software product lines: using annotations or compositions. Annotation-based techniques model variabilities introducing implicit or explicit annotations on the requirements description. For example, if the the annotation-based approach models requirements using UML use case diagrams, some marks like some especial tags or stereotypes would be introduced on the diagram to indicate variability.

An example of an annotation-based technique is Model Template [15]. Model Template uses UML activity diagrams as the representation form of requirements. To model variability, Model Templates uses two annotation elements: presence conditions and meta-expressions. Presence conditions indicate the presence or absence of optional features, it works as a configuration knowledge to generate a product line member's requirements. Meta-expressions are used as annotations directly on the activity diagram to compute optional attributes on the activity diagram control flow. Figure 3.5 shows an example of an activity diagram modeled with model template technique. At this figure we can see the list of presence conditions and tags located between brackets that indicates which feature that branch belongs to.

Composition-based techniques, on the other hand, implements features and other variations as distinct modules and so, to generate a product line member's requirements artifact, there must be a composition of these modules. An example of a composition-based technique is

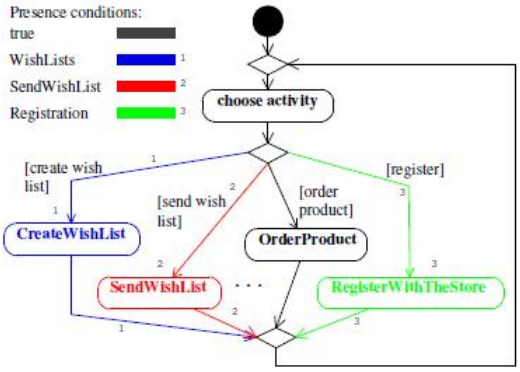

Figure 3.5 Model template: an example of an annotation-based technique. Figure extracted from [15].

DECIMAL [44]. DECIMAL is an automated tool used for product lines' requirements specification. Requirements descriptions along with some rules and attributes are defined and saved in the system as separated modules and, to generate the requirements of a member from the product family, DECIMAL tool composes the requirements artifacts and automatically checks for requirements consistency, that is, if the presence or absence of a feature breaks any rule defined for that product line. Figure 3.6 shows a picture of DECIMAL interface used to insert an attribute that varies according to the features chosen to compose the new product.

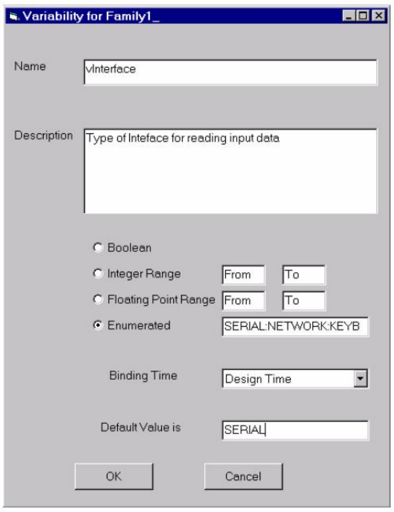

Figure 3.6 Decimal: an example of an composition-based technique. Figure extracted from [44].

3.5.2 Sources of Variation

At first it was our intention to extract each technique's supported sources of variation [3]. According to Bachmann there are 5 possible sources of variation in a software product line. Variation points can come from a function, a data structure, the control flow or behavior of the system, quality goals and the environment chosen to host the system. For example, some techniques may support function variability but may not support quality goals variability.

Extracting the supported sources of variation from retrieved techniques would help us to learn more about the technique's capabilities and limitations. But retrieving this information required a significant effort since almost none of the authors has analyzed their techniques using such notation and to discover it we would have to apply case studies and analyze the techniques' behavior to each source of variation. On chapter 5 we'll leave this task as a future work.

3.6 Data Analysis

In this work we have opted for a quantitative analysis. All data extracted from quality assessment tables and data extraction forms will be compiled and synthesized on different tables and graphics that will help to answer the research questions proposed at the beginning of the Systematic Review. Chapter 4 shows the results achieved by data analysis phase.

CHAPTER 4

Results

This chapter discusses the 4 research questions proposed at the beginning of the systematic review and answer each one of them using the data retrieved from the 40 selected papers.

4.1 What are the Requirements Variability Management Techniques available?

Our search strategy was able to retrieve 35 papers that proposed new techniques or at least extended or improved an already existent one. Some papers that were written by the same authors sometimes talk about the same technique with some improvements. Table 4.1 shows the papers' titles and the index given to each one of them. Since not all of the techniques retrieved by our search had a name, from now on we'll be using this index to refer to them.

The remaining 5 papers retrieved by our search described literature surveys or experiments that compared two or more techniques. These papers were useful to find out what methods are being used to analyze and compare different approaches that model requirements variability. Papers from [B] category will be discussed on Section 4.4

4.2 Where do these techniques come from? In what context are they being used?

Crossing the information extracted from both categories papers, we have compiled a list of countries and institutions involved with requirements variability management research. Table 4.2 shows this list. Observing the list we can notice that not only universities were involved with this research field but also some multinational enterprizes like Nokia and Siemens, and some government institutions like Ministry of Work and Social Affairs of Spain.

To learn how influent a country was in this knowledge domain figure 4.1 describes a graphic showing the number of papers that each country was involved with. We can see that Germany is the most relevant country on the domain of requirements variability management since it was involved with 7 papers, while Brazil, Canada and USA gets the second place, participating in 6 of the 40 papers retrieved.

Table 4.1 Papers retrieved and classified as category [A]

Index	Title
1[A]	Aspectual Support for Specifying Requirements in Software Product Lines [49]
2[A]	Concept Analysis for Product Line Requirements [43]
3[A]	Configuring Features with Stakeholder Goals [53]
4[A]	From Stakeholder Goals to Product Features: Towards a Role-Based Variability Framework with Decision Boundary [8]
5[A]	Modeling Software Product Lines with AoURN [41]
6[A]	Non-Functional Requirements Analysis Modeling for Software Product Lines [42]
7[A]	Requirements Variability Models: Meta-model based Transformations [36]
8[A]	Security Requirements Variability for Software Product Lines [39]
9[A]	Use Case Description of Requirements for Product Lines [4]
10[A]	Modeling Variability by UML Use Case Diagrams [51]
11[A]	Tailoring Use Cases for Product Line Modeling [52]
12[A]	Modelling Behaviors in Product Lines [54]
13[A]	DECIMAL: A Requirements Engineering Tool for Product Families [44]
14[A]	Marrying Features and Use Cases for Product Line Requirements Modeling of Embedded Systems [38]
15[A]	Visual Variability Analysis for Goal Models [25]
16[A]	Using Goal-Models to Analyze Variability [26]
17[A]	A Domain Analysis Method for Software Product Lines Based on Scenarios, Goals and Features [33]
18[A]	An Approach to Developing Domain Requirements as a Core Asset Based on Commonality and Variability Analysis in a Product Line [40]
19[A]	Extended Kaos to Support Variability for Goal Oriented Requirements Reuse [22]
20[A]	Mapping Features to Models: A Template Approach Based on Superimposed Variants [15]
21[A]	A Variability Management Process for Software Product Lines [16]
22[A]	Product-Line Requirements Specification (PRS): an Approach and Case Study [23]
23[A]	A Model-Driven Approach for Software Product Lines Requirements Engineering [37]
24[A]	A Metamodel for Aspectual Requirements Modelling and Composition [2]
25[A]	Tracing between Features and Use Cases: A Model-Driven Approach [1]
26[A]	Modelling Requirements Variability across Product Lines [12]
27[A]	Modeling Scenario Variability as Crosscutting Mechanisms [10]
28[A]	Requirements Engineering for Product Families [35]
29[A]	Generic Variability Management and Its Application to Product Line Modelling [48]
30[A]	Model Composition in Product Lines and Feature Interaction Detection Using Critical Pair Analysis [29]
31[A]	Managing requirements specifications for product lines: An approach and industry case study [20]
32[A]	Requirements Modelling and Design Notations for Software Product Lines [11]
33[A]	Use Case-based Testing of Product Lines [5]
34[A]	Multiple-View Meta-Modeling of Software Product Lines [24]
35[A]	Product Line Modeling with Generic Use Cases [30]

Table 4.2 Countries and institutions involved with requirements variability modeling research

Country	Institutions
Belgium	Facultés Universitaires Notre-Dame de la Paix, University of Namur
Brazil	PUC-Rio, Universidade Estadual de Maringá, Universidade de São Paulo, Universidade Federal de Pernambuco, Universidade de Pernambuco
Canada	University of Toronto, University of Ottawa, University of Waterloo
Corea	Sogang University, Pusan National University
Denmark	IT University of Copenhagen
Finland	Nokia Research Center, Helsinki University of Technology
France	IRISA, Université Paris Est, Université Paris Descartes
Germany	RWTH Aachen University, Fraunhofer Institute for Experimental Software Engineering, siemens, University of Duisburg, pure-systems GmbH, Software.Process.Management, SynSpace, Technical University of Munich, Continental, Automotive Systems Division, Hewlett-Packard, Harman/Becker Automotive Systems GmbH, Fraunhofer IESE, Technical University of Berlin, arvato direct services GmbH
Italy	Università di Firenze, Area della Ricerca di Pisa
Portugal	Universidade Nova de Lisboa
Spain	Valladolid University, Ministry of Work and Social Affairs, University of Castilla La-Mancha
Sweden	BAE Systems Hägglunds
United Kingdom	The Open University, Lancaster University, The Queen's University of Belfast
USA	University of Nebraska, George Mason University, Iowa State University, University of Oregon, Texas Tech University

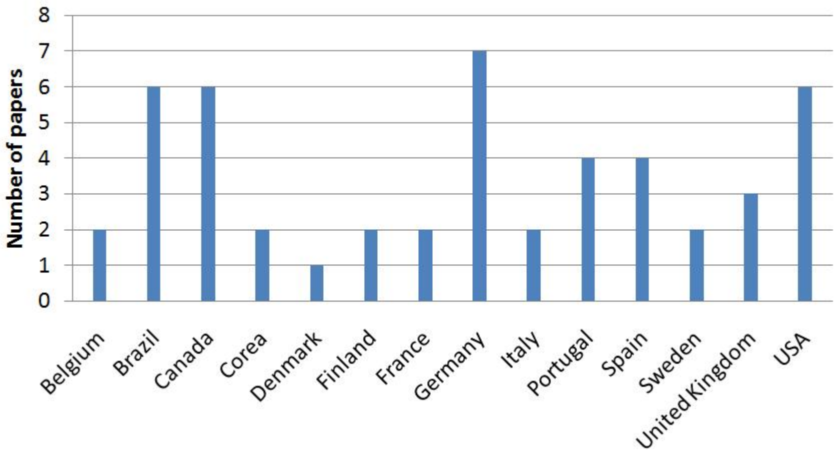

Figure 4.1 Graphic showing how many papers each country was involved with.

4.3 What are their main characteristics?

The extraction form presented on data extraction section of chapter 3 describes a set of relevant characteristics to learn about the retrieved techniques. This characteristics are:

- Type of requirement that can be modeled (functional, not functional, both);
- How requirements are being represented? (UML diagrams, Goals, scenarios description, activity diagram, etc);
- What element is used to model variation point (stereotypes, crosscutting concerns, etc)
- What notation is being used? (Graphical, textual);
- Is there a support tool? In case it does, is it free or paid?
- Variation mechanism (annotation-based or composition-based [32]);

These characteristics were extracted by hand, without any tool support, from each of the 35 papers. Table 4.3 describe the techniques' characteristics

Figure 4.2 shows the proportion between techniques that support only functional requirements, non-functional requirements and both. The techniques that support both functional and non-functional requirements are mostly based on i-star (i*), NFR [14] and requirements descriptions. 6[A] and 8[A] techniques manage exclusively non-functional requirements, 8[A] deals only with security requirements. Graphic shown on Figure 4.3 shows the proportion between techniques that model requirements using graphical, textual or both notations. Figure 4.4 show composition-based techniques versus annotation-based techniques. In general

Table 4.3 Requirements modeling techniques for software product lines

Index	Requirements	Representation Model	Elements of variation	Notation	Support tool	Variation Mechanism
1[A]	Both	Scenario descriptions	crosscutting concerns	Textual	Prototype	Composition
2[A]	Both	Scenario descriptions	crosscutting concerns	Textual	No	Composition
3[A]	Both	i* goal model	maps i* to feature models	Graphical	Yes	Composition
4[A]	Both	i* goal model	maps i* to feature models	Graphical	No	Annotation
5[A]	Both	URN	crosscutting concerns	Graphical	No	Annotation
6[A]	Non-Functional	Use cases description	extended PLUS approach	Textual	No	Annotation
7[A]	Both	goal model	extended goal model	Graphical	No	Annotation
8[A]	Non-Functional	uml class diagram	meta-model	Graphical	No	Annotation
9[A]	Functional	Use cases description	extends use cases description	Textual	No	Annotation
10[A]	Functional	uml use case diagram	extends use cases diagrams	Graphical	No	Annotation
11[A]	Functional	Use cases description	extends use cases	Both	No	Annotation
12[A]	Functional	uml class diagram	HMSC	Graphical	No	Annotation
13[A]	Functional	req. Description	automated tool	Textual	Yes	Composition
14[A]	Functional	Use cases description	extends use cases description	Both	No	Annotation
15[A]	Both	goal model	meta-model	Graphical	Yes	Annotation
16[A]	Both	goal model	meta-model	Graphical	No	Annotation
17[A]	Both	goal and scenario descr.	meta-model	Both	No	Annotation
18[A]	Functional	uml use case diagram	meta-model	Graphical	Yes	Annotation
19[A]	Both	Kaos	meta-model	Graphical	No	Annotation
20[A]	Functional	uml activity diagrams	meta-model	Graphical	Yes	Annotation
21[A]	Functional	uml use case diagram	meta-model	Graphical	No	Annotation
22[A]	Both	req. Description	meta-model	Textual	No	Composition
23[A]	Functional	uml diagrams	automated tool	Graphical	Yes	Composition
24[A]	Functional	uml diagrams	Domain Specific Language	Graphical	Prototype	Composition
25[A]	Functional	use cases	meta-model	Both	Prototype	Composition
26[A]	Both	Scenario descriptions	meta-model	Textual	Yes	Composition
27[A]	Functional	Scenario descriptions	crosscutting concerns	Textual	No	Composition
28[A]	Both	req. Description	meta-model	Graphical	Yes	Annotation
29[A]	Both	generic specification	decision model	Both	No	Composition
30[A]	Functional	uml activity diagrams	meta-model	Graphical	Yes	Composition
31[A]	Functional	Scenario descriptions	annotations	Textual	Yes	Annotation
32[A]	Functional	extended feature model	meta-model	Graphical	Yes	Annotation
33[A]	Functional	Scenario descriptions	annotations	Textual	No	Annotation
34[A]	Functional	uml use case diagram	extends use case diagrams	Graphical	Prototype	Annotation
35[A]	Functional	uml and textual use cases	annotations	Both	No	Annotation

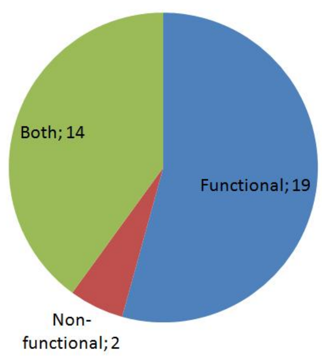

Figure 4.2 Graphic showing what type of requirements can be supported by the retrieved techniques.

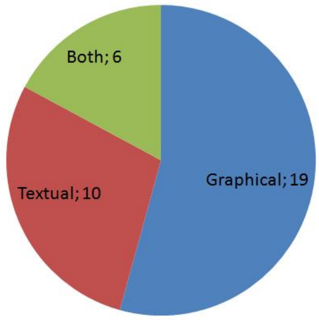

Figure 4.3 Graphic showing the proportion between the notations used by the retrieved techniques.

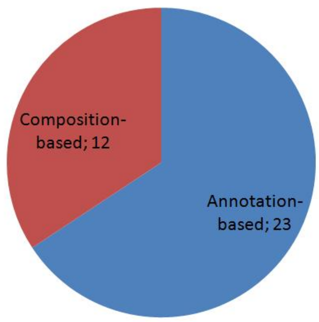

Figure 4.4 Graphic showing the proportion between composition-based and annotation-based techniques.

4.4 How is this type of research being conducted?

4.4.1 Category [A] Quality Assessment

The quality assessment tables defined on Chapter 3 had the intention to analyze two main aspects from the retrieved techniques: what motivation did the authors give to their technique and how the technique was presented and analyzed on the paper. Next subsections will be discussing the results of these aspects.

4.4.1.1 Motivation Analysis

We chose to assess the motivation description from papers classified as category [A] so that we could learn about the quality and rigor that the research field of requirements models to software product lines has achieved. After going through all papers from category [A] we could classify the motivation description of each paper into one of the following classes:

A. The author(s) did not justify the need for the new technique being proposed. The paper goes straight to the technique's description without a motivation;

B. The author(s) made brief and informal comments on the introduction section about the problems that he had faced on the already existent techniques and assumed that the new technique would correct the described limitations;

C. The author(s) made a literature survey on related works to raise existing problems and justify the need for the new technique;

D. The author(s) justified the need for the new technique with motivating examples that pointed out some limitations of new techniques.

Graphic 4.5 shows the results obtained by the motivation's description assessment. We can see that 20 of 35 papers didn't present, using more formal methods, the reasons that led to the design of a new technique. As a matter of fact, 4 of these 20 papers didn't present at all a motivation before introducing the new technique. Only 12 papers made some kind of literature survey on related works in order to find gaps and limitations that would be overcome by the new technique proposed. In addition, 3 papers justified the need for a new technique discussing some motivational examples to point out some limitations existent on already proposed techniques.

We also learned that analyzing the papers' motivation description could help us to learn more about what problems and limitations are most frequently pointed by the authors as a motivation to the design of a new technique. Getting to know these problems would give us a general idea of the most important obstacles to overcome when designing new requirements modeling approaches for software product lines. Since this questioning was not on the scope of our research questions we chose to leave this task as a future work.

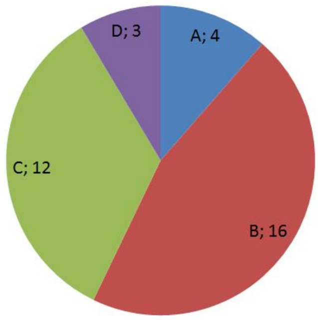

Figure 4.5 Motivation description assessment.

4.4.1.2 Methods' Analysis

The second aspect assessed by the quality assessment phase was what methods did the researchers use to analyze the proposed techniques. After assessing all 35 papers the analysis methods could be classified in one of the following classes:

A. The author(s) presented some examples during the approach's description but there was no further analysis;

B. The author(s) presented a case study using the proposed technique to analyze its results pointing benefits and limitations;

C. The author(s) presented a case study using the proposed technique and other existent techniques to make a comparison between them;

Graphic 4.6 shows the results obtained by the methods assessment. The retrieved results shows that 20 papers used some examples while describing the approach, but did not use any method to analyze the proposed technique after describing it. Other 13 papers used a case study as a method for analyzing benefits and limitations and only 2 retrieved papers made a case study using 2 or more techniques. The same papers that compared their proposed techniques with others (27[A] and 31[A]) were the only ones that used metrics to compare the results between techniques, all the other papers made textual analysis.

4.4.2 Category [B] Quality Assessment

At the beginning of quality assessment phase, the 40 selected papers were divided into [A] and [B] categories. A paper was classified as [B] category if it was a literature survey or an experiment with requirements modeling approaches for software product lines. Only 5 papers were classified as [B] category. This section will briefly present each one of them. Table 4.4 shows papers' title and their respective indexes.

4.4.2.1 Analysis of paper 1[B]

Paper 1[B] is a literature survey that compares existent types of notation to model requirements to software product lines. The first part of the discussion classifies the techniques to

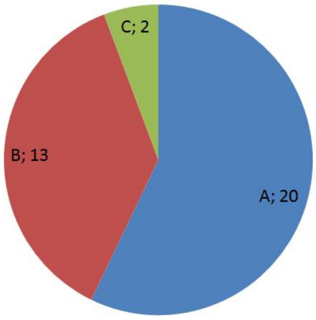

Figure 4.6 Methods' analysis.

Table 4.4 Papers retrieved and classified as category [B]

Index	Title
1[B]	Modelling variability requirements in Software Product Lines: a comparative survey [50]
2[B]	Using Requirements Management Tools in Software Product Line Engineering: The State of the Practice [6]
3[B]	On the Benefits of Scenario Variability as Crosscutting [9]
4[B]	Separating Variability Concerns in a Product Line Re-Engineering Project [28]
5[B]	Modelling Variation in Quality Attributes [21]

their specific notation, then, using some examples, there's a comparison between graphical and textual-based techniques pointing some general characteristics, their benefits and limitations. The final parts describes a case study and its results made with all cited techniques. There was no metrics to analyze the results, all the analysis was textual.

4.4.2.2 Analysis of paper 2[B]

The objective of paper 2[B] is to propose a set of requirements to requirements variability management tools' development. To achieve this goal the paper first describes a literature survey made with the intent to find an study that already described such benchmark. Since the survey was not successful, the authors collected a set of industrial scenarios and modeled them using some of the existent tools for requirements' variability management for software product lines.

After a analysis of tools' features and limitations they described a set of functional requirements to developing tools for requirements' variability management.

4.4.2.3 Analysis of paper 3[B]

The objective of paper 3[B] is to create a discussion about the benefits around the use of crosscutting concerns to model requirements variability. First the paper describes the technique that was already proposed by the same author on 27[A] paper. After presenting a case study made with 3 different techniques, the paper describes a framework of metrics suited to compare different requirements variability techniques. After comparing the different techniques using the proposed metrics' framework, the authors arrives to some conclusions about the benefits of crosscutting concerns.

4.4.2.4 Analysis of paper 4[B]

The objective of paper 4[B] is to present the process followed by the authors to elicit the variability of PloneMeeting, an example of software product line, and report on the results obtained when applying variability modeling techniques promoting separation of concerns between software variability and product line variability.

To achieve both goals described above, the researchers first describes the the variability modeling approach adopted to the case study. Then there's a description of PloneMeeting, the example of product family chose to run the case study. Then the paper delineates the research context of the case study, and states the research question. After describing the environment where the experiment took place, the experiment itself is presented followed by some textual analysis. There was no metrics collection.

4.4.2.5 Analysis of paper 5[B]

The objective of paper 5[B] is to present a survey of existing approaches for specifying variation of quality requirements. First there is a discussion about the importance of quality requirements for software product lines, then the paper describes the techniques chosen to run the case study. The case study is described and analyzed just by narrative discussion without and a framework for comparing quality attributes modeling techniques for software product lines is proposed.

4.5 Conclusions

All research questions could be answered using the information retrieved from the 40 selected papers. This section will discuss some conclusions about the retrieved techniques and about the quality and rigor present on the methods used to propose new techniques.

At the total 14 countries were involved until now with the research and proposal of requirements models for SPL. From these 14 countries, Germany is the most relevant one followed by Brazil, Canada and USA. We can see that there is a good number of new approaches being proposed and there are great differences among them.

Some techniques use more traditional models from requirements engineering like UML diagrams, requirements textual descriptions or scenarios description, others use newer models like i-star, NFR, KAOS and URN. The type of model used to represent requirements for SPL is an important factor to take into account before adopting one technique. A company that is already used to model requirements using scenario descriptions for example, feels much more comfortable choosing a technique that uses the same type of representation.

Some approaches support only functional requirements modeling. These techniques are mostly based on strictly functional requirements models like UML activity and use case diagrams. On the other hand, those that support both functional and non-functional requirements are based mostly on i-star and NFR. There are 2 techniques that focus exclusively in non-functional requirements, 6[A] and 8[A]. 6[A] works for any type of non-functional requirement and 8[A] support exclusively security requirements.

One alarming information is that from 35 new techniques only 11 presents tool support.

Perhaps some of the 24 other techniques are still working on tool support or maybe has already proposed a tool for supporting their technique. Either way the lack of a tool is another important factor to analyze before adopting a new approach, especially for product families with bigger systems with extensive requirements artifacts.

As for the quality and rigor present on the research methods used to propose and compare new techniques we can see that this research field has not achieved yet a high level of maturity. The efforts on this research field are mostly concentrated on proposing new techniques instead of analyzing and comparing already existent ones. From the 40 selected papers, 35 proposed new techniques and only 5 proposed a study or survey to analyze and compare existing approaches.

We can also notice that the majority of the papers retrieved did not present a formal literature survey or examples to motivate the need for a new technique. Another negative finding is that after describing the proposed technique, most papers don't present a case study, a comparison between techniques or any other type of evaluation method to justify why the proposed technique should be adopted.

From 40 selected papers only 3 collected metrics after running case studies to compare the techniques. As far as we can see, frameworks describing methods and metrics for comparing requirements models for SPL have not been yet formally proposed, will present this task as a future work on Chapter 5.

CHAPTER 5

Conclusions and Future Work

Our work was based on two main objectives:

- To run a systematic review of existent requirements management approaches for SPL, learn their main characteristics and get to know what methods and metrics researchers have been using until now to propose, evaluate and compare them;

- Study, execute and evaluate the process of conducting systematic reviews.

Next sections will discuss some final remarks and propose future works for both goals described above.

5.1 Requirements Variability Models for Software Product Lines

The systematic review was able to conclude that many new techniques for solving the problem of representing requirements for domain and application level of software product lines are being proposed. These techniques use the most varied types of notation to describe functional and non-functional requirements. The diversity among techniques is important because makes it easier for a company to adopt the technique that fits best to its development process. On the other hand, it serves nothing to have so many different techniques if they won't provide a good tools support.

Our results concluded that the research field of requirements approaches for SPL is still poor of methods and metrics for evaluating and comparing different techniques. With few exceptions, there has not been effort on producing papers that experimented and analyzed these techniques using more formal methods. Because of that, one suggestion of future works that we suggest is the proposal of a framework of methods and metrics that would serve to analyze and compare different techniques.

Since our research did not target to find and discuss individual technique's benefits and limitations, a suggestion for future work that we thought of was a survey, possibly using the systematic review process, to search for the limitations described by the literature of this area, to learn what are the main issues found on the different types of requirements techniques for SPL.

Another possible path to learn more about techniques benefits and limitations is an experiment of running comprehensive case studies (with product lines that embraced many types of sources of variation) with different approaches looking the search of supported sources of variation.

5.2 Systematic Review

Systematic review's process has proven to be a very powerful tool to to extract essential information from the literature and analyze this information according to a previously chose method. The presence of the review protocol makes the survey more reliable and less biased. More refined protocols tend to provide more reliable the results and that is exactly the main goal of evidence based software engineering.

In order to achieve more refined protocols, we proposed a new approach for conducting systematic reviews. Differently from the original process defined by Kitchenham [34], where the review's execution only began after the whole protocol was defined, this new approach is an iterative model where when each part of the protocol is defined, it gets executed and validated. Using an iterative model you can anticipate problems that would only be found much later on the review's process.

Systematic reviews are more efficient than ordinary reviews because they follow a protocol. But defining a protocol, executing it and analyzing its results is a task that requires a great amount of effort. Searching for all studies that match the keywords defined on the search strategy is tiring because information is wide spread all over the world wide web and some databases do not support more sophisticated strings. Extracting information from selected papers is really hard when there is no tool support, and, once the information is extracted, synthesize them and get to the results is also a difficult task.

A solution to diminish the fatiguing work of systematic reviews could be an integrated development environment for conducting systematic reviews. The environment could provide tools to help with the search strategy, data extraction and data synthesis. The search strategy tool could use some a web search algorithm to crawl the web in the search of papers that matched with the defined keywords. In our work we used *Publish or Perish* tool as a checklist of what we had gathered from the web.

To help with the data extraction, the environment could provide a tool to extract parts of the papers that included some key terms or expressions defined by the user. While the information was being extracted, the user would be able to store them into a database using a data warehouse schema [46], defined previously by the user, so that, after extracting all the data, it would be easier for him to get different views from the data and get to the conclusions. We leave this IDE for conducting systematic reviews as a suggestion for future works. To conclude, systematic reviews are the key concept of evidence based software engineering, but its process has a lot yet to improve.

Bibliography

[1] M Alférez, U Kulesza, A Moreira, J Araújo, and V Amaral. Tracing between features and use cases: A model-driven approach. *sse.uni-due.de*.

[2] Mauricio Alférez, Uirá Kulesza, Nathan Weston, João Araujo, Vasco Amaral, Ana Moreira, Awais Rashid, and Michael C. Jaeger. A metamodel for aspectual requirements modelling and composition., 2008.

[3] Felix Bachmann and Len Bass. Managing variability in software architectures. *SIGSOFT Softw. Eng. Notes*, 26(3):126–132, 2001.

[4] A Bertolino, A Fantechi, S Gnesi, G Lami, and A. Maccari. Use case description of requirements for product lines. *International Workshop on Requirements Engineering for Product Lines (REPL'02)*, 2002.

[5] Antonia Bertolino and Stefania Gnesi. Use case-based testing of product lines. *SIGSOFT Softw. Eng. Notes*, 28(5):355–358, 2003.

[6] D. Beuche, A. Birk, H. Dreier, A. Fleischmann, H. Galle, G. Heller, D. Janzen, I. John, R.T. Kolagari, T. von der Massen, and A. Wolfram. Using requirements management tools in software product line engineering: The state of the practice. In *Software Product Line Conference, 2007. SPLC 2007. 11th International*, pages 84–96, Sept. 2007.

[7] S Bühne, K Lauenroth, and K Pohl. Why is it not sufficient to model requirements variability with feature models. *Proceedings of Workshop: Automotive Requirements Engineering (AURE04)*, 2004.

[8] Catalin Bidian. From stakeholder goals to product features: towards a role-based variability framework with decision boundary. In *PST '06: Proceedings of the 2006 International Conference on Privacy, Security and Trust*, pages 1–5, New York, NY, USA, 2006. ACM.

[9] Rodrigo Bonifácio, Paulo Borba, and Sérgio Soares. On the benefits of scenario variability as crosscutting. In *EA '08: Proceedings of the 2008 AOSD workshop on Early aspects*, pages 1–8, New York, NY, USA, 2008. ACM.

[10] R Bonifácio and P Borba. Modeling scenario variability as crosscutting mechanisms. *Proceedings of the 8th ACM International Conference on Aspect-Oriented Software Development (AOSD'09)*, 2009.

[11] TJ Brown, R Gawley, I Spence, P Kilpatrick, C. Gillan, and R. Bashroush. Requirements modelling and design notations for software product lines. *1st International Workshop on Variability Modelling of Software-intensive Systems*, 2001.

[12] S. Buhne, K. Lauenroth, and K. Pohl. Modelling requirements variability across product lines. In *Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on*, pages 41–50, Aug.-2 Sept. 2005.

[13] L Chen, MA Babar, C Cawley, and 2009. A survey of evaluation of variability management approaches. *staff.ul.ie*.

[14] B. A. Yu Chung, L. Nixon and J. E. Mylopoulos. Non-functional requirements in software engineering. *Kluwer International Series in Software Engineering*.

[15] K Czarnecki and M Antkiewicz. Mapping features to models: A template approach based on superimposed variants. *Lecture notes in computer science*, 2005.

[16] Edson Alves de Oliveira, Junior, Itana M. S. Gimenes, Elisa Hatsue Moriya Huzita, and José Carlos Maldonado. A variability management process for software product lines. In *CASCON '05: Proceedings of the 2005 conference of the Centre for Advanced Studies on Collaborative research*, pages 225–241. IBM Press, 2005.

[17] Arilo C. Dias Neto, Rajesh Subramanyan, Marlon Vieira, and Guilherme H. Travassos. A survey on model-based testing approaches: a systematic review. In *WEASELTech '07: Proceedings of the 1st ACM international workshop on Empirical assessment of software engineering languages and technologies*, pages 31–36, New York, NY, USA, 2007. ACM.

[18] T. Dyba, T. Dingsoyr, and G.K. Hanssen. Applying systematic reviews to diverse study types: An experience report. In *Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on*, pages 225–234, Sept. 2007.

[19] Tore Dybå and Torgeir Dingsøyr. Empirical studies of agile software development: A systematic review. *Inf. Softw. Technol.*, 50(9-10):833–859, 2008.

[20] Magnus Eriksson, Jürgen Börstler, and Kjell Borg. Managing requirements specifications for product lines - an approach and industry case study. *Journal of Systems and Software*, 82(3):435 – 447, 2009.

[21] L. Etxeberria, G. Sagardui, and L. Belategi. Modelling variation in quality attributes. In *First International Workshop on Variability of Software-Intensive Systems (VaMos 2007)*, 2007.

[22] Semmak F., Gnaho C., and R. Laleau. Extended kaos to support variability for goal oriented requirements reuse. *Model Driven Information Systems Engineering: Enterprise, User and System Models (MoDISE-EUS)*, 2008.

[23] Stuart R. Faulk. Product-line requirements specification (prs): An approach and case study. *Requirements Engineering, IEEE International Conference on*, 0:0048, 2001.

[24] H. Gomaa and M.E. Shin. Multiple-view meta-modeling of software product lines. In *Engineering of Complex Computer Systems, 2002. Proceedings. Eighth IEEE International Conference on*, pages 238–246, 2002.

[25] B. Gonzalez-Baixauli, J. C. S. do Prado Leite, and J. Mylopoulos. Visual variability analysis for goal models. *Proceedings of the IEEE Int. Req. Eng. Conf. (RE)*, 2004.

[26] B. Gonzalez-Baixauli, M.A. Laguna, and J.C. Sampaio do Prado Leite. Using goal-models to analyze variability. *Second International Uorkshop on Variability Modelling of Software-intensive Systems*, 2008.

[27] Harzing.com. Publish or perish, 2009.

[28] A. Hubaux, P. Heymans, and H. Unphon. Separating variability concerns in a product line re-engineering project. In *EA '08: Proceedings of the 2008 AOSD workshop on Early aspects*, pages 1–8, New York, NY, USA, 2008. ACM.

[29] P Jayaraman, J Whittle, AM Elkhodary, and Hassan Gomaa. Model composition in product lines and feature interaction detection using critical pair analysis. *Lecture notes in computer science*, 2007.

[30] I John and D Muthig. Product line modeling with generic use cases. *Second Software Product Line Conference, San Diego*, 2002.

[31] Kyo C. Kang, Sholom G. Cohen, James A. Hess, William E. Novak, and A. Spencer Peterson. Feature-oriented domain analysis feasibility study. *Software Engineering Institute, Pittsburgh*, 1990.

[32] Christian Kästner, Sven Apel, and Martin Kuhlemann. Granularity in software product lines. In *ICSE '08: Proceedings of the 30th international conference on Software engineering*, pages 311–320, New York, NY, USA, 2008. ACM.

[33] Minseong Kim, Hwasil Yang, and Sooyong Park. A domain analysis method for software product lines based on scenarios, goals and features. *Asia-Pacific Software Engineering Conference*, 0:126, 2003.

[34] Barbara A. Kitchenham, Tore Dyba, and Magne Jorgensen. Evidence-based software engineering. In *ICSE '04: Proceedings of the 26th International Conference on Software Engineering*, pages 273–281, Washington, DC, USA, 2004. IEEE Computer Society.

[35] Juha Kuusela and Juha Savolainen. Requirements engineering for product families. In *ICSE '00: Proceedings of the 22nd international conference on Software engineering*, pages 61–69, New York, NY, USA, 2000. ACM.

[36] Miguel A. Laguna and Bruno Gonzalez-Baixauli. Requirements variability models: meta-model based transformations. In *MIS '05: Proceedings of the 2005 symposia on Metainformatics*, page 9, New York, NY, USA, 2005. ACM.

[37] Alferez M., Kulesza U., Moreira A., Araújo J., and Amaral V. A model-driven approach for software product lines requirements engineering. *In Proceedings of the 20th International Conference on Software Engineering and Knowledge (SEKE'2008)*, 2008.

[38] Eriksson M. and Börstler J.and Borg K. Marrying features and use case for product line. *Proceedings of the Fourth Conference on Software Engineering Research and Practice in Sweden SERPS'04*, 2004.

[39] Daniel Mellado, Eduardo Fernandez-Medina, and Mario Piattini. Security requirements variability for software product lines. In *ARES '08: Proceedings of the 2008 Third International Conference on Availability, Reliability and Security*, pages 1413–1420, Washington, DC, USA, 2008. IEEE Computer Society.

[40] Mikyeong Moon, Keunhyuk Yeom, and Heung Seok Chae. An approach to developing domain requirements as a core asset based on commonality and variability analysis in a product line. *IEEE Transactions on Software Engineering*, 2005.

[41] Gunter Mussbacher, Daniel Amyot, ao Araújo, Jo and Ana Moreira. Modeling software product lines with aourn. In *EA '08: Proceedings of the 2008 AOSD workshop on Early aspects*, pages 1–8, New York, NY, USA, 2008. ACM.

[42] Quyen L. Nguyen. Non-functional requirements analysis modeling for software product lines. *Modeling in Software Engineering, International Workshop on*, 0:56–61, 2009.

[43] Nan Niu and Steve Easterbrook. Concept analysis for product line requirements. In *AOSD '09: Proceedings of the 8th ACM international conference on Aspect-oriented software development*, pages 137–148, New York, NY, USA, 2009. ACM.

[44] P Padmanabhan and R Lutz. Decimal: A requirements engineering tool for product families. *International Workshop on Requirements Engineering for Product Lines*, 2002. 8 cites.

[45] K Pohl, G Böckle, and F van der Linden. *Software Product Line Engineering: Foundations, Principles and Techniques 2005*.

[46] CSR Prabhu and 2004. Data warehousing: Concepts, techniques, products and applications. *books.google.com*.

[47] David L. Sackett, William M. C. Rosenberg, Muir J. A. Gray, Brian R. Haynes, and Scott W. Richardson. Evidence based medicine: what it is and what it isn't. *BMJ*, 312(7023):71–72, 1996.

[48] Klaus Schmid and Isabel John. A customizable approach to full lifecycle variability management. *Science of Computer Programming*, 53(3):259 – 284, 2004. Software Variability Management.

[49] Harvey Siy, Prasanna Aryal, Victor Winter, and Mansour Zand. Aspectual support for specifying requirements in software product lines. In *EARLYASPECTS '07: Proceedings of the Early Aspects at ICSE*, page 2, Washington, DC, USA, 2007. IEEE Computer Society.

[50] J.C. Trigaux and P. Heymans. Modelling variability requirements in software product lines: A comparative survey. *Technical report PLENTY project, Institut d'Informatique FUNDP*, 2003.

[51] T von der Maßen, H Lichter, and R Aachen. Modeling variability by uml use case diagrams. *Proceedings of the International Workshop on Requirements Engineering for Product Lines (REPL'02)*, 2002.

[52] T von der Maßen, H Lichter, and R Aachen. Modeling variability by uml use case diagrams. *Proceedings of the International Workshop on Requirements Engineering for Product Lines (REPL'02)*, 2002.

[53] Yijun Yu, Julio Cesar Sampaio do Prado Leite, Alexei Lapouchnian, and John Mylopoulos. Configuring features with stakeholder goals. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, pages 645–649, New York, NY, USA, 2008. ACM.

[54] Tewfik Ziadi, Loïc Hélouët, and Jean-Marc Jézéquel. Modelling behaviors in product lines. 2002.

This volume has been typeset in LaTeX with the UFPEThesis class (www.cin.ufpe.br/~paguso/ufpethesis).