ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/271132608

Analysis of Non-functional Properties in Software Product Lines: A Systematic Review

Conference Paper · August 2014

DOI: 10.1109/SEAA.2014.48

CITATIONS
6

READS
135

5 authors, including:

Larissa Rocha Soares
Universidade Federal da Bahia

5 PUBLICATIONS **9** CITATIONS

SEE PROFILE

Pasqualina Potena
Swedish Institute of Computer Science

32 PUBLICATIONS **272** CITATIONS

SEE PROFILE

Ivan Do Carmo Machado
Universidade Federal da Bahia

46 PUBLICATIONS **437** CITATIONS

SEE PROFILE

Ivica Crnkovic
Chalmers University of Technology

239 PUBLICATIONS **3,667** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project Investigation on Product Line Architecture Recovery View project

Project SmartHomeRiSE DSPL View project

All content following this page was uploaded by Ivan Do Carmo Machado on 25 February 2015.

The user has requested enhancement of the downloaded file.

Analysis of Non-Functional Properties in Software Product Lines: a Systematic Review

Larissa Rocha Soares*, Pasqualina Potena†, Ivan do Carmo Machado*,
Ivica Crnkovic‡ and Eduardo Santana de Almeida*§
*Federal University of Bahia, Salvador - Brazil
†University of Alcalá, Alcalá de Henares, Madrid, Spain
‡Mälardalen University, Västerås, Sweden
§Fraunhofer Project Center for Software and Systems Engineering, Brazil
Email: {larissars,ivanmachado,esa}@dcc.ufba.br, p.potena@uah.es, ivica.crnkovic@mdh.se

***Abstract*—Software Product Lines (SPL) approach has been widely developed in academia and successfully applied in industry. Based on the selection of features, stakeholders can efficiently derive tailor-made programs satisfying different requirements. While SPL was very successful at building products based on identified features, achievements and preservation of many non-functional properties (NFPs) remain challenging. A knowledge how to deal with NFPs is still not fully obtained. In this paper, we present a systematic literature review of NFPs analysis for SPL products, focusing on runtime NFPs. The goal of the paper is twofold: (i) to present an holistic overview of SPL approaches that have been reported regarding the analysis of runtime NFPs, and (ii) to categorize NFPs treated in the scientific literature regarding development of SPLs. We analyzed 36 research papers, and identified that system performance attributes are typically the most considered. The results also aid future research studies in NFPs analysis by providing an unbiased view of the body of empirical evidence and by guiding future research directions.**

Keywords—Systematic Literature Review, Software Product Lines, Product Derivation, Non-functional Properties

I. Introduction

Software Product Lines (SPL) engineering practices have been proved to be widely applicable and a successful approach in both industry and academy, and from different perspectives. The key principles of SPL development are playing an increasingly important role in many specific practice areas in software engineering [1]. SPL engineering encompasses two main processes [2]: *domain engineering* and *application engineering*. The former aims at establishing a reusable platform, and the latter is responsible for deriving products from the platform. SPL products are distinguished in terms of features, i.e., end-user visible characteristics of products [3]. Based on the selection of features, stakeholders can derive tailor-made products satisfying a range of functional (FPs) and non-functional properties (NFPs).

FPs implement the tasks/functionalities of a software. On the other hand, NFPs are those that impose special conditions and qualities on the system [4]. They are often referred to as Quality Attributes (QAs)[1], usually observable by end-users. For instance, if a software system runs slower than expected, users may not be interested in using it, regardless the provided functionalities. Functional and non-functional aspects should work fine for a product to be viable in the market.

Etxeberria and Sagardui [5] showed that most approaches for quality analysis aims to investigate product line QAs, i.e., not observable via execution or development attributes. In this sense, this paper focuses on execution/runtime properties, visible and measurable at source-code or during the program execution [6], such as reliability and performance. These are highly relevant properties in next generation computing applications, such as embedded and real-time systems [7]. In fact, emerging computing application paradigms require systems that are not only reliable, compact and fast, but which also optimize many different competing and conflicting objectives, e.g., as response time and consumption of resources [8].

SPL in practice has been very successful in managing features that comprise both FPs and a large number of NFPs. However, there are many NFPs that cannot be expressed and then realized in form of features, but require different approaches. How to deal with them is still not well established, neither in theory nor in practice. There is a need to provide an systematic knowledge about this.

To the best of our knowledge, little work has focused on the analysis of QAs for SPL engineering [9] [10]. Besides, they are often incomplete in terms of comprehensive coverage, or focused on different research topics. In [11], Montagud and Abrahão proposed a catalog of measures for QAs found in the SPL lifecycle. In this present investigation, we elaborate on this work, by investigating how runtime NFPs are introduced in the SPL approaches.

We carried out a systematic literature review [12] of the published literature on SPL approaches reporting on NFPs. The goal of this is twofold: (i) to present a holistic overview of SPL approaches that have been reported regarding the analysis of runtime NFPs, and (ii) to categorize NFPs used in the scientific literature regarding development of SPL.

The remainder of this paper is organized as follows. Section II describes the research method used in this review. Section III reports the results. Section IV discusses the implications of our results for research and practice, along with the limitations of the review. Finally, Section V presents our conclusions and suggests areas for further research.

[1]In the following, we will interchangeably speak of QAs and NFPs.

II. Research Method

A systematic literature review (SLR) was performed to find and summarize available evidence of runtime NFPs in SPL engineering. A SLR provides an objective assessment of a research topic in a reliable, rigorous, and methodological way [11]. Along this Section, we detail each element composing this SLR, mainly the research questions, the strategy employed to locate primary studies, and the data extraction and synthesis process.

A. *Research Questions (RQs)*

This review aims at presenting a holistic overview of the existing studies that have been reported regarding the analysis of runtime NFPs. Concretely, we stated the following RQs:

RQ1. How SPL approaches handle runtime NFPs?

This question aims at identifying the approaches that handle runtime NFPs in SPL engineering. We analyzed the main goals of the approaches and which NFPs they address. The term "approach" is herein used to refer to a published document describing a method, technique, or process related to the topic under investigation. Thus, we split this question in three sub-questions:

- **RQ1.1 What approaches handle runtime NFPs in SPL?** The purpose of this question is to leverage approaches handling runtime NFPs in SPL.
- **RQ1.2 What NFPs emerge at runtime?** This question aims at identifying NFPs that are commonly handled at runtime. Such kind of properties are those which usually depend on observations via execution or operation [5].
- **RQ1.3 What application domains are best covered by the existing approaches?** This question aims to leverage the application domains (e.g., scientific systems, business systems etc.) where the identified approaches were used.

RQ2. How much evidence is available to support the approaches?

This question assesses the level of evidence of each identified approach. The level of evidence is a means to evaluate the maturity of methods and tools. Based on evidence-based healthcare research [13] [14], Kitchenham [12] proposed a hierarchy of Software Engineering study designs, classifying studies into five levels **(L)** of evidence. Later on, Alves et al. [15] proposed a more practical assessment, by defining the following hierarchy: **L1**- no evidence; **L2**- evidence obtained from demonstration or working out toy examples; **L3**- evidence obtained from expert opinions or observations; **L4**- evidence obtained from academic studies, e.g., controlled lab experiments; **L5**- evidence obtained from industrial studies, e.g., causal case studies; and **L6**- evidence obtained from industrial practice. The hierarchy consists of classifying from weakest to strongest evidence which each study presents. We found it to be a suitable classification to provide answers to this RQ.

RQ3. What are the limitations of the existing runtime NFPs support?

Understanding the limitations of the investigated approaches helps ensuring accurate interpretation of findings. Thus, we performed a critical appraisal on the quality of the included studies, under established criteria. Table I lists the quality criteria used in this assessment. They were defined based on the work of Bass [16] and Alves et al. [15], and cover the following issues [16]: (i) *Reporting* - assesses the quality of the study's rationale, aims, and context; (ii) *Rigor* - assesses the rigor of the research methods employed to establish the validity of data and the trustworthiness of the findings; (iii) *Credibility* - assesses the credibility of the study methods for ensuring that the findings were valid and meaningful; and (iv) *Relevance* - assesses the relevance of the study for the software industry at large and the research community.

TABLE I. Quality assessment criteria

	Criteria	Issue
RQ3.1	Is the paper based on research?	*Reporting*
RQ3.2	Is there a clear statement of the aims of the research?	
RQ3.3	Is there an adequate description of the context in which the research was carried out?	
RQ3.4	Was the research design appropriate to address the aims of the research?	*Rigor*
RQ3.5	Was the data analysis sufficiently rigorous?	
RQ3.6	Is there a way to validate the methods?	*Credibility*
RQ3.7	Is there a clear statement of findings?	
RQ3.8	Are there any guidelines to practitioner interested in using this method?	*Relevance*

B. *Search strategy*

The process of identifying primary studies started by executing a search query on the electronic databases for research (the *search engines*). The repositories used were: *ACM Digital Library, IEEE Xplore, Science Direct, Springer,* and *Scopus*. They, together, cover almost all important journals, conferences, and workshops papers in the Software Engineering field.

Based on our research goal, which is investigating runtime NFPs approaches in SPL engineering, the following major search keywords were used to formulate the search query: *quality attributes* and *software product lines*. Alternative words and synonyms were also used for such keywords. Then, it was created an initial pilot search string by joining major keywords with Boolean AND operators, and the alternative words and synonyms with Boolean OR operators. As a means to calibrate the search string, we first applied it in the search engines, in a general search, the query was computed on the whole body of searchable content. Next, it was carried out more specific searches by delimiting the scope to include some conferences and journals which we were aware of their likely results. Hence, after some trials, the search string employed in all source engines is represented as follows:

*("quality attributes" **OR** "non-functional" **OR** "extra-functional") **AND** ("software product line" **OR** "software product family" **OR** "SPL").*

We faced some problems when carrying out the automated search: a huge number were not relevant to the context of this study; it was found a number of publications, in the form of, e.g., summaries, tutorial abstracts, and so on, that cannot be considered as research papers. Conversely, our search did not retrieve important papers for the field we were aware of. In this effect, as a means to capture as many relevant citations as possible, we also carried out a manual, issue-by-issue, search over peer-reviewed journals, and conference and workshop proceedings. For the sake of space, all the reviewed sources are listed in the Review Website[2].

C. Study selection criteria

The search performed with the search string is a syntactic search, that results in a set of papers in which the research string appears. However, it is possible that a retrieved paper matches the search string, but it addresses topics other than the focus of this research. Therefore, a semantic check served to complement the syntatic search.

The semantic search we followed was earlier applied as a process to screening papers for subsequent review [17]. The first step is to read the title and abstract of the papers in order to ensure that the collection is consistent with the research area under study. Hence, while reading the titles and abstracts of the retrieved papers, we applied the inclusion and exclusion criteria to select the papers. Based on such criteria, we could select all candidate primary studies to be included in this review. These are papers that matched the inclusion criteria, and did not fit in any exclusion criterion. The inclusion and exclusion criteria used were:

- Inclusion criteria:
  - Papers published up to June in 2013.
  - Papers dealing with runtime NFPs in SPL engineering.
  - Papers which handle QAs from the exectution/runtime viewpoint, i.e., attributes that usually cannot be analyzed before product assembly.
- Exclusion criteria:
  - Papers not written in English.
  - Papers presenting QAs not suitable for prediction/ estimation/measurement in final products.
  - Introductory papers for special issues, books or posters.
  - Duplicate copy of the same research study.

Figure 1 illustrates the search and selection process employed in this review. Phase 1 denotes the automated search carried out within the search engines. The search engines retrieved a pool of 2,635 publications. We used the StArt tool [18] and electronic spreadsheets to help organizing the references. From such a set, 103 studies were duplicate, as more than one search engine indexed the same venues. We also found 3 literature reviews. Moreover, by reading the titles and abstract of the remaining papers, the number of 2,394 were considered to be irrelevant to the focus of this review, and as such they were discarded at this point.

[2]The website with supplementary resources can be found at http://bit.ly/sr-euromicro2014.

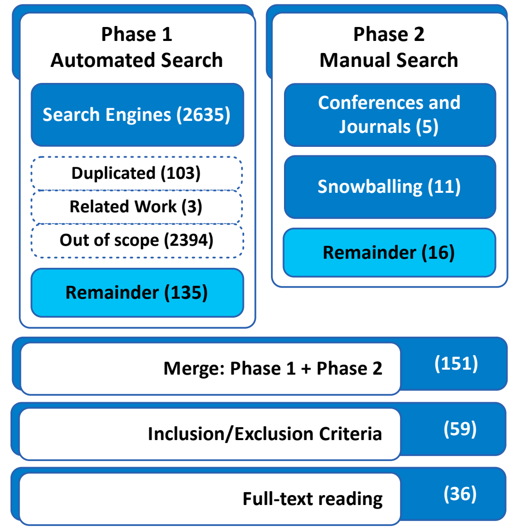

Fig. 1. Search and selection process.

Phase 2 encompasses the manual search. By visiting the proceedings of conferences and workshops, and journals, we found 5 additional papers. Furthermore, a snowballing search was employed, in which the references of every retrieved paper was visited. It resulted in 11 more papers. Hence, the manual task contributed with 16 more papers.

After merging the results from both automated and manual search, we had a pool of 151 papers. We applied the inclusion/ exclusion criteria in this set, to identify those to consider for a full-text reading. Then, we had a set of 59 papers. After reading the 59 primary study candidates, 36 papers were selected as primary studies in this SLR.

D. Data extraction and quality assessment

We created a data extraction form for the purpose of answering the RQs and extracting relevant information from the primary studies (S). The synthesis process and data extraction was carried out by reading all 36 papers, listed in Table II and Review Website (added more information), and extracting relevant data, as follows.

- Venue, title, year, and author(s).
- A short description of the proposed approach (RQ1), including motivation and goal of the study (RQ1.1).
- QAs addressed in the paper (RQ1.2).
- The application domains to which runtime NFPs are worked on (RQ 1.3).
- The evidence level of the proposed approaches, according to the list earlier presented in Section II-A (RQ2).
- The quality of the study, according to the quality assessment criteria presented in Table I (RQ3).

III. Results

Figure 2 shows the temporal view of the primary studies, in which the published papers were identified from the year 2003. We may notice a trend curve revealing an increase of the number of papers as of the year 2007. The small amount of studies in 2013 (compared with 2012) is justified by the fact that this SLR only covered the first semester of 2013. Such a trend indicates an increasing interest by the SPL community in creating novel approaches to satisfy users NFPs.

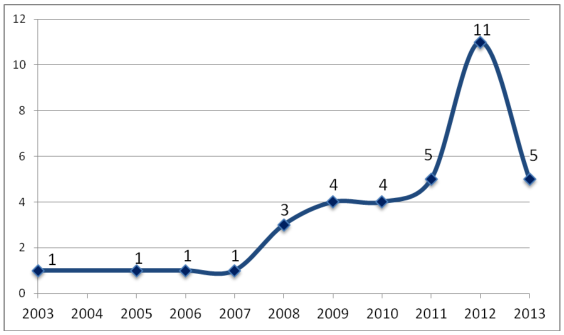

Fig. 2. Number of studies by publication year.

As described in Section II, we extracted and synthesized data to answer our RQs. In this section, we discuss the findings of the primary studies (listed in Table II) by considering the RQs.

TABLE II. PRIMARY STUDIES

Studies	Title
[$S1$]	D. Benavides et al., "Automated Reasoning on Feature Models,". in *CAiSE*, 2005.
[$S2$]	M. Aoyama and A. Yoshino, "AORE (Aspect-Oriented Requir. Engin.) Methodology for Automotive Software Product Lines," in *APSEC*, 2008.
[$S3$]	J. Bartholdt el al., "Integrating Quality Modeling with Feature Modeling in Software Product Lines," in *ICSEA*, 2009.
[$S4$]	L. Etxeberria and G. Sagardui, "Variability Driven Quality Evaluation in Software Product Lines," in *SPLC*, 2008.
[$S5$]	C. Ghezzi and A. M. Sharifloo, "Model-based verification of quantitative non-functional properties for software product lines," *Information and Software Technology*, 2013.
[$S6$]	J. González-Huerta et al., "Non-functional Requirements in Model-driven Software Product Line Engineering" in *NFPinDSML*, 2012.
[$S7$]	J. González-Huerta et al., "A Multimodel for Integrating Quality Assessment in Model-Driven Engineering," in *QUATIC*, 2012.
[$S8$]	V. Guana and D. Correal, "Variability Quality Evaluation on Component-based Software Product Lines," in *SPLC* 2011.
[$S9$]	S.S. Kolesnikov et al., "Predicting Quality Attributes of Software Product Lines Using Software and Network Measures and Sampling," in *VaMoS*, 2013.
[$S10$]	T.M. Dao et al., "Problem Frames-Based Approach to Achieving Quality Attributes in Software Product Line Engineering," in *SPLC*, 2011.
[$S11$]	B. Mohabbati, "Development and Configuration of Service-oriented Systems Families," in SAC 2011.
[$S12$]	V. Nunes, "Variability Management of Reliability Models in Software Product Lines: An Expressiveness and Scalability Analysis," in *SBCARS*, 2012.
[$S13$]	Q.L. Nguyen, "Non-functional requirements analysis modeling for software product lines",in *MISE*, 2009.
[$S14$]	R. Olaechea et al., "Modelling and Multi-objective Optimization of Quality Attributes in Variability-rich Software," in *NFPinDSML*, 2012.
[$S15$]	F. Roos-Frantz et al., "Quality-aware analysis in product line engineering with the orthogonal variability model," *Software Quality Journal*, 2012.
[$S16$]	N. Siegmund et al., "Measuring Non-Functional Properties in Software Product Line for Product Derivation," in *APSEC*, 2008.
[$S17$]	N. Siegmund et al., "Optimizing Non-functional Properties of Software Product Lines by means of Refactorings," in *VaMoS*, 2010.
[$S18$]	N. Siegmund et al., "Predicting performance via automated feature-interaction detection," in *ICSE*, 2012.
[$S19$]	N. Siegmund et al., "SPL Conqueror: Toward optimization of non-functional properties in software product lines," *Software Quality Journal*, 2012.
[$S20$]	N. Siegmund et al., "Interoperability of non-functional requirements in complex systems," in *SEES*, 2012.
[$S21$]	N. Siegmund et al., "Scalable prediction of non-functional properties in software product lines: Footprint and memory consumption," *Information and Software Technology*, 2013.
[$S22$]	J. Sincero et al., "Approaching Non-functional Properties of Software Product Lines: Learning from Products," in *APSEC*, 2010.
[$S23$]	J. Sincere et al., "Towards Tool Support for the Configuration of Non-Functional Properties in SPLs," in *HICSS*, 2009.
[$S24$]	S. Soltani et al., "Automated Planning for Feature Model Configuration Based on Functional and Non-functional Requirements," in *SPLC*, 2012.
[$S25$]	R. Tawhid and D.C. Petriu, "Automatic Derivation of a Product Performance Model from a Software Product Line Model," in *SPLC* 2011.
[$S26$]	R. Tawhid and D. Petriu, "User-friendly Approach for Handling Performance Parameters During Predictive Software Performance Engineering," in *ICPE*, 2012.
[$S27$]	V. Karina et al., "Towards Product Configuration Taking into Account Quality Concerns," *SPLC* 2012.
[$S28$]	H. Zhang et al., "Quality Prediction and Assessment for Product Lines," in *CAiSE*, 2003.
[$S29$]	G. Zhang et al., "Quality attribute modeling and quality aware product configuration in software product lines," *Software Quality Journal*, 2013.
[$S30$]	V. Guana and D. Correal, "Improving software product line configuration: A quality attribute-driven approach," *Information and Software Technology*, 2013.
[$S31$]	E. Bagheri et al., "Configuring Software Product Line Feature Models Based on Stakeholders' Soft and Hard Requirements", *SPLC*, 2010.
[$S32$]	J. Bosch and J. Lee, "Usage Context as Key Driver for Feature Selection," in *SPLC*, 2010.
[$S33$]	J. White et al., "Automating Product-Line Variant Selection for Mobile Devices," *SPLC*, 2007.
[$S34$]	J. White et al., "Selecting highly optimal architectural feature sets with Filtered Cartesian Flattening," *Journal of Systems and Software*, 2009.
[$S35$]	S. Jarzabek et al., "Addressing quality attributes in domain analysis for product lines," *IEEE Proc.-Soft.*, 2006.
[$S36$]	G. Jianmei et al., "A genetic algorithm for optimized feature selection with resource constraints in software product lines," *Journal of Systems and Software*, 2011.

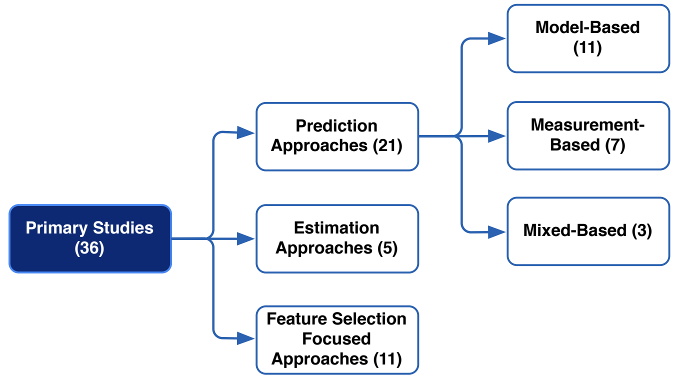

Fig. 3. Primary studies classification

A. RQ1 - SPL approaches

The quality of an SPL product is affected by the selected features and their composition. Different feature selections or compositions may differ for the software products NFPs, such as reliability, availability and performance. We next provide more details about the SPL approaches.

RQ1.1: What approaches handle runtime NFPs in SPL?

We categorized the primary studies into three groups, two for approaches presenting quality analysis (*Quality Prediction (QP)* and *Quality Estimation (QE)*), and one for approaches aimed at *Feature Selection (FS)*, as Figure 3 shows.

QP approaches seek to predict runtime NFPs of SPL products based on either historical data, expert knowledge or software metrics thresholds, or a model of the feature composition (i.e., by deriving the model of a product from the family model). **QE** approaches set runtime NFPs on the basis of measures obtained by either analyzing the source code of SPL products or observing their running software[3].

FS-focused approaches provide guidelines to select features according to FPs and NFPs. Therefore, their main goal is not the quality analysis. Also, some of these approaches suggest the optimal set of product configuration actions needed to tackle required NFPs.

It is worth mentioning that the quality analysis can be *quantitative* - returning numerical results (e.g., performance and reliability bounds) and *qualitative* - providing more general considerations on products quality. Here below we describe the categories of the primary studied identified.

*1) **Prediction Approaches**:* The approaches for predictive NFPs analysis of SPL products can be categorized by the kind of techniques used: (a) model-based, (b) measurement-based, and (c) mixed - combination of (a) and (b). These are discussed next. [4]

(a) *Model-Based* approaches basically give the guidelines to predict NFPs by considering UML models, like sequence diagrams [S5][S12], and variability modelling techniques, such as feature models [S10][S29]. These diagrams are enriched

[3] An SPL product (also called *variant*) is produced by mapping the selected features to implementation units (e.g., software components or services).

[4] Note that this kind of classification is typically used for quality prediction approaches (see [19] for performance prediction of component-based systems).

and annotated with QAs information provided, for example, by experts' knowledge.

On the one hand, the quantitative analysis of NFPs is based on, for example: (i) Bayesian Belief Network (BBN) model, to capture the design knowledge and experiences of domain experts [S13][S28]; or (ii) Analytic Hierarchical Process (AHP), which needs less experts efforts than BBN, according to [S29]; Also evaluation strategies have been adopted. For example, queuing network models have been used to support product performance evaluation [S25][S26], and parametric model checking techniques have been exploited to verify NFPs of different configurations of an SPL [S5][S12]. On the other hand, for example, in [S10] a qualitative framework has been introduced for reasoning impacts of inadequate/insufficient domain assumptions on NFPs, while in [S35] the feature-oriented domain analysis has been extended with qualitative goal-oriented analysis.

(b) *Measurement-based* approaches aim at predicting NFPs based on, for example, the identification of relevant feature interactions [S18], or relevant/common features sets [S9][S22][S23]. Generating all products from an SPL to perform the quality analysis for each is impratical. In an SPL, the number of products increases exponentially with the number of features. The ongoing work [S09], for example, predicts quality by using feature sets defined in a sampling process based on static program analysis. The work in [S22][S23] provide a feedback to stakeholders from products measured through a limited set of features, based on a testing infrastructure. Other works are able to predict program performance by detecting performance-relevant feature interactions and measuring to what extent they interact [S18].

(c) *Mixed* approaches provide both quality variability modeling and measurement techniques. The ongoing work in [S27] investigates (i) qualitative modeling techniques that better fit in security and performance properties, besides (ii) how to use expert knowledge on security, and static analysis of feature implementations to derive performance metrics. Regarding the specification of NFPs, model-driven engineering principles have been exploited, for example, in [S6]. In particular, a mechanism for the validation of NFPs fulfillment, through the association of measures and thresholds, is presented. The work in [S4] proposes to reduce evaluation cost and effort in evaluating the different products by creating a generic evaluation model. This latter also discuss how to select a limited set of designs or products to quantitatively evaluate the quality of products.

There is a sub-group of prediction approaches that, besides quality analysis, provides support for the optimization of product configuration using the Constraints Satisfaction Problem (CSP). For example, [S19][S20][S21] developed a technique called SPL Conqueror. [S20] works with NFPs limited to the evaluation scenario, such as picture frequency; [S21] works with quantifiable NFPs, exemplified by footprint and memory consumption; and [S19] is more generic.

*2) **Estimation Approaches**:* This category mainly encompasses studies that, besides providing a means to estimate NFPs, provide support for the selection of implementation alternatives for the selected features. For example, on the one hand, the work in [S16] focuses on a quantitative approach based on the source code analysis for estimating the binary size of feature implementations. On the other hand, the approach is based on the run of a benchmark, proposed for performance evaluation of the whole product. The work in [S17] consider refactorings (changes in the source code structure) in correspondence to certain NFPs. The estimation method [S16] is also exploited in the holistic approach [S19][S20][S21] to optimize NFPs in SPL engineering. Other works also developed approaches to remove/replace some components with alternative ones, in order to achieve required quality levels [S02][S11][S30]. Additionally, [S02] proposes an aspect-oriented methodology, [S11] addressed a service-oriented approach, and [S30] is based on a model-driven strategy.

*3) **Feature Selection focused**:* The works of this category are not specifically focused on the quality analysis. Thus, NFPs parameters are simulated, or random values are generated. Usually, these works do not explicitly distinguish between predicted values and estimated means.

The feature selection activity has been supported from different perspectives in order to, for example: (i) generate a CSP from the feature model (see, e.g., [S1][S33]) or an orthogonal variability model [S15] associated with quality information; (ii) exploit the multi-objective optimization for the product derivation [S14]; (iii) drive the feature selection with product usage contexts [S32]; or (iv) adopt artificial intelligence techniques, based on a fuzzy propositional language [S31], genetic algorithms [S36], and Hierarchical Task Network artificial technique [S24].

Another class of related papers proposes, for example, a model-driven approach for quality evaluation [S8], or deal with the integration of quality modeling with feature modeling (see, e.g., [S3][S7]), showing how the customer decisions affects the QAs [S3]. Also, the filtered cartesian flattening approximation technique [S34] was used for the feature selection.

RQ1.2: What NFPs emerge at runtime?

In Software Engineering there are different kinds of QAs [20] classified in different ways [6][21][22]. In [22], QAs are specified into two categories: evolution and execution QAs. Whereas evolution attributes are observable during system development lifecycle, execution ones are observable at runtime, and dependent on factors such as system behaviour, number of existing features and which are selected for the product. Research efforts have been devoted for evolution attributes [5]. In this sense, our proposal focuses on execution/runtime QAs, which are highly relevant and need to be properly addressed.

The work in [22] also defines the execution/runtime category by using eight attributes: performance, security, availability, usability, scalability, reliability, interoperability and adaptability. Each attribute represents a more conceptual term and encompasses a set of fine-grained attributes. For example, performance is related to the relationship between the level of performance of the software and the amount of resources used, under stated conditions. It can be composed by the attributes, such as response time, space, latency, throughput, execution speed, resource usage, memory usage, accuracy, etc. [20].

Table III reports the details of our NFPs categorization, found in the 36 primary studies. Despite some of them

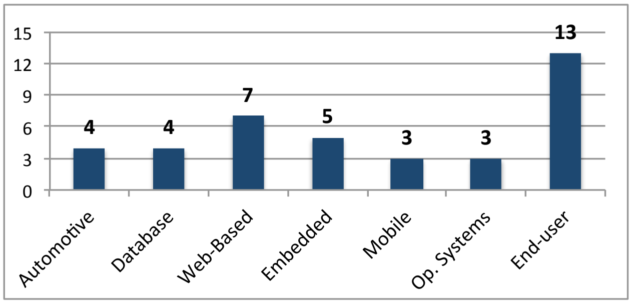

Fig. 4. Number of studies by application domain

are domain-specific attributes and resource-related constraints, such as Channel Capacity/Latency, WLAN bandwidth, Cycle and Speed, we considered the NFPs nomenclature exactly as mentioned in the original sources. As presented in the table, the large majority of studies is focused on performance issues (53.84%). However, we can also see from the primary studies' results that other attributes (like usability - 17.30%, reliability - 11.53%, and security - 11.53%) are also being investigated, and research effort have already been spent for their evaluation (see, e.g., [S5] for reliability).

Compared to [20], a systematic review with 182 studies, resulting in 114 identified different types of NFPs, our investigation identified a smaller number of attributes. Our results are grounded on two main points: (i) we only considered studies handling SPL; and (ii) the QAs investigated are those from execution point-of-view, usually dependent on system behaviour, application domain and users' needs. There is an increasing number of systems built for applications where the user feedback and hardware restrictions are the most important requirements. In this scenario, execution NFPs are as important as functional requirements.

RQ1.3: What application domains are best covered by the existing approaches?

The 36 studies concern seven main application domains (i.e., automotive, database management systems, web-based, embedded, mobile, operating systems and end-users' domain-specific). Figure 4 gives an overview of the distribution of the studies based on their application domain. Each primary study considers one or more application domains.

After examining the main runtime attributes (i.e., performance, security, availability, usability, scalability, reliability, interoperability and adaptability) covered in each study, we identified, from the included studies, that the performance and reliability properties are typically considered in the different kinds of domains. Each domain is related at least to one of the performance (and reliability) sub-attributes.

Regarding the other properties, we observed that the security attributes are considered in four domains (automotive, end-user, web-based and embedded), and usability is presented in three different domains (end-user, web-based and embedded). The complete mapping between the domains and their associated sub-attributes is presented in the Review Website.

B. *RQ2 - Available evidence*

As described in Section II, six different levels were used to assess the level of evidence of each identified approach. We can see from the primary studies' results that large majority of the 36 studies, about 53%, are academic works (L4); about 42% of studies show evidence obtained from demonstration or working out toy examples (L2), while about 5% of studies do not show maturity of their methods and tools (L1). No studies were found neither with evidence from expert opinions or observations (L3), from industrial studies (L5) nor industrial practice (L6).

C. *RQ3 - Limitations of the existing support*

The primary studies were categorized according to the quality assessment criteria presented in Table I. These criteria correspond to (YES or NO) questions. Regarding questions related to *Reporting* issue (i.e, RQ3.1, RQ3.2 and RQ3.3), all approaches were based on research with clear aims and suitable context's description. Thereby, these three criteria related to *Reporting* issue in general received a positive answer.

For the first question of the *Rigor* issue (RQ3.4), the results highlight, in general, that the studies present an appropriate design to address their goals. On the contrary, for RQ3.5 and other three questions related to *Credibility* and *Relevance* issues, discrepancies become more evident. Figure 5 shows percentage distributions of the answers (YES or NO) for these four questions (i.e., RQ3.5, RQ3.6, RQ3.7, RQ3.8).

For RQ3.5, the results show that few approaches (about 39%) perform a rigorous data analysis (e.g., by presenting experimental material, design and procedures). For the *Credibility* issue (RQ3.6), results show that about 58% of studies deal with the validation of their approaches. This result compromises the trustworthiness and usefulness of the approach and prevents the comparison with others approaches. On the contrary, the studies typically provide a clear statement of findings (RQ3.7), as well as a validation future work plan. However, in general the studies do not provide guidelines to follow when practitioners reuse the approach (RQ3.8). This result highlight that there are open research challenges due mainly to the lack of rigor in validation or lack of flexibility for approach reuse.

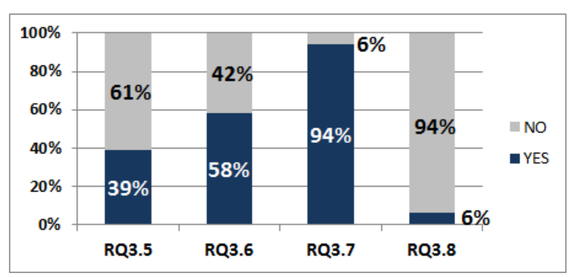

Fig. 5. Percentage distributions of the answers for RQ3.5, RQ3.6, RQ3.7 and RQ3.8

IV. Analysis and Discussion

This section discusses the key findings of our study, as well as validity threats in this review.

TABLE III. NON-FUNCTIONAL PROPERTIES (NFPS)

1. Accuracy	14. Dispatch response time	27. Memory consumption	40. Scalability
2. Availability	15. Ease of use	28. Message size	41. Security
3. Available CPU	16. Effort	29. Minimum waiting time	42. Service time
4. Channel capacity/latency	17. Energy consumption	30. Performance	43. Severity of product failures
5. Comfort	18. Energy efficiency	31. Picture frequency	44. Space - main memory
6. Communication performance	19. Execution time	32. Position accuracy	45. Space - secondary storage
7. Completeness	20. Fault tolerance	33. Privacy	46. Speed
8. Confidentiality	21. File size	34. Range	47. Time - throughput
9. Convenient use of the system	22. Footprint	35. Recognition time	48. Usability
10. CPU consumption	23. Friendly user interface	36. Reliability	49. Usability for APT resident
11. Customer satisfaction	24. Integrity	37. Response time	50. Usability for VIPs
12. Cycle	25. Usability for handicapped	38. Safety	51. Weight
13. Data access security	26. Latency	39. Save energy	52. WLAN bandwidth

We identified three main categories of approaches that handle runtime NFPs in SPLs (see Figure 3). Each of the approaches identified in the review has its specific focus and context that it is appropriate for. In fact, to get better understanding of the development of the field, we provided a critical appraisal on the quality of the included studies, under established criteria for the maturity, the validation rigor, and the approach applicability.

Since the effort of measuring all products can be very costly and error-prone during quality analysis, predictions arise as a strategy to overcome this problem. **QP** approaches usually make use of three strategies: (i) modelling techniques in combination with domain experts' judgments and experiences (*model-based*); (ii) measurement of a small set of products *(measurement-based)*; or even a combination with both *(mixed-based)*. These strategies have positive and negative points. For example, for measurement-based approaches, producing real products is a time-consuming task and source-code is necessary. Approaches based on models avoid instantiating real products, but provide rough quality assessment. The choice between these approaches must take into account aspects such as type of QAs analysed, the development stage of interest and stakeholders' needs.

For **QE** approaches, this group provides implementation alternatives in order to minimize/maximize NFPs according to customers preferences. **FS** approaches, although not providing an explicit quality analysis with discussions to predict or estimate NFPs, support optimizations by providing, in most cases, tools for automated feature selection during the product derivation process.

In the last years, the topic of definition and analysis of NFPs in SPLs has been studied in several communities (e.g., in the automotive domain [S2]). In fact, the trend of the temporal curve reveals an increase of the number of papers (see Figure 2). This is due to the fact that SPL principles have been proved to be widely applicable and a successful approach. However, notwithstanding the increasing interest and diffusion of these practices, grand research challenges still need to be addressed.

Although in recent years there was an increasing number of publications, in general only few researches involve real-world products (e.g., [S18][S19][S21]). Most primary studies used case studies as a proof of concept, as a way to validate or show their approaches in "practice". The methods are usually not properly applied and important details are missing, mainly related to experiment design and material, variables and analysis procedure.

General remarks in relation to the current state of the art are as follows: (i) the analysis of few NFPs is in general performed, and (ii) the SPL approaches considers the NFPs important for the SPL practices. In particular, the outcomes confirm that NFPs are highly relevant properties in next generation computing applications. Moreover, tradeoff analysis among multiple competing and conflicting objectives should be supported, which is in general missing in the state of the art and practice today.

Validity threats. The main threats to validity are bias in our selection of primary studies, classification and data extraction. We developed a review protocol containing detailed information about RQs, search strategy, selection criteria, data extraction and quality assessment. The protocol was designed by the first author, and was then reviewed by the others to check if each research step was appropriately planned. Although the protocol was revised several times until the authors have agreed on every point addressed, there was a risk of missing some relevant studies, not covered by our search strategy. To mitigate such a threat, we also carried out a manual search within the most important venues in the SPL engineering field. In addition, to handle threats surrounding the classification of NFPs, the first and second authors undertook an in-depth analysis on every included primary study, in order to collect enough information to the review, and also discussed the results until reach an agreement on the classification. However, it is possible that our classification may not be similar to other in the literature. To ensure correctness of the data extraction, a template was proposed for a consistent RQs extraction. Furthermore, RQ3 was designed aimed at assessing the limitations of every included primary study. It is a means to improve the reliability and guide the interpretation of the findings.

V. CONCLUDING REMARKS

The main objective of this systematic review has been to obtain a holistic view of SPL approaches that have been reported regarding the analysis of runtime NFPs. We have identified 36 primary studies. Each of them share a lot in common, e.g. focus, goal and application context. We have extracted commonalities and summarized the studies into three main categories of themes.

This systematic review might have implications for both research and practitioners. The analysis of the primary studies indicates a number of challenges and topics for future research: (i) it should be provided support for the tradeoff analysis among competing NFPs both at domain engineering and application engineering levels; and (ii) it is also necessary to analyze dependencies between different kinds of NFPs and the SPL lifecycle-related practices.

The QAs identified in each approach, together with the application domains, served as input for the analysis, where performance and usability attributes were the most commonly identified runtime properties, similarly to [20].

We claim that addressing the highlighted challenges will require the contributions from researchers and industrial experts in different fields. We could include optimization formulation (e.g., several metaheuristic techniques with different characteristics could be adopted depending on the nature of application domain), system properties assessments, and experimentation on real world large SPL, considering realistic model parameter values. As future work, we intend to understand practitioners perceived strengths, limitations, and needs associated with using NFPs for SPL practices in the industry.

Acknowledgment

This work was partially supported by the National Institute of Science and Technology for Software Engineering (INES[5]), funded by CNPq and FACEPE, grants 573964/2008-4 and APQ-1037-1.03/08 and CNPq grants 305968/2010-6, 559997/2010-8, 474766/2010-1 and FAPESB. This work was also partially supported by European Commission funding under the 7h Framework Programme IAPP Marie Curie program for project ICEBERG no. 324356.

References

[1] P. Clements and L. Northrop, *Software Product Lines: Practices and Patterns*. Addison-Wesley, 2001.

[2] K. Pohl, G. Böckle, and F. J. v. d. Linden, *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer-Verlag, 2005.

[3] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson, "Feature-Oriented Domain Analysis (FODA) Feasibility Study," Pittsburgh, PA, USA, Tech. Rep. CMU/SEI-90-TR-21, 1990.

[4] D. Lohmann, O. Spinczyk, and W. Schrder-preikschat, "On the configuration of non-functional properties in operating system product lines," in *Proc. of the 4th AOSD Workshop on Aspects, Components, and Patterns for Infrastructure Software*, 2005.

[5] L. Etxeberria and G. Sagardui, "Product-line architecture: New issues for evaluation," in *SPLC*. Springer-Verlag, 2005, pp. 174–185.

[6] I. Crnkovic, M. Larsson, and O. Preiss, "Concerning predictability in dependable component-based systems: Classification of quality attributes," in *Architecting Dependable Systems III*. Springer-Verlag, 2005.

[7] N. Siegmund, M. Rosenmüller, M. Kuhlemann, C. Kästner, S. Apel, and G. Saake, "SPL Conqueror: Toward optimization of non-functional properties in software product lines," *Software Quality Journal*, vol. 20, no. 3-4, pp. 487–517, 2012.

[8] M. Harman, W. B. Langdon, Y. Jia, D. R. White, A. Arcuri, and J. A. Clark, "The GISMOE challenge: Constructing the pareto program surface using genetic programming to find better programs," in *ASE*, 2012.

[9] X. Peng, Y. Yu, and W. Zhao, "Analyzing evolution of variability in a software product line: From contexts and requirements to features," *Information & Software Technology*, vol. 53, no. 7, pp. 707–721, 2011.

[10] V. Myllärniemi, M. Raatikainen, and T. Männistö, "A Systematically Conducted Literature Review: Quality Attribute Variability in Software Product Lines," in *SPLC*. ACM, 2012, pp. 41–45.

[11] S. Montagud, S. Abrahão, and E. Insfrán, "A systematic review of quality attributes and measures for software product lines," *Software Quality Journal*, vol. 20, no. 3-4, pp. 425–486, 2012.

[12] B. Kitchenham and S. Charters, "Guidelines for performing Systematic Literature Reviews in Software Engineering," Keele University and Durham University Joint Report, Tech. Rep. EBSE 2007-001, 2007.

[13] NHMRC, *How to use the evidence: assessment and application of scientific evidence*, Australian National Health and Medical Research Council, Feb 2000.

[14] J. Higgins and S. Green, Eds., *Cochrane Handbook for Systematic Reviews of Interventions*. The Cochrane Collaboration, 2011, version 5.1.0.

[15] V. Alves, N. Niu, C. F. Alves, and G. Valença, "Requirements engineering for software product lines: A systematic literature review," *Information & Software Technology*, vol. 52, no. 8, pp. 806–820, 2010.

[16] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*, 2nd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2003.

[17] P. A. M. S. Neto, I. C. Machado, J. D. Mcgregor, E. S. Almeida, and S. R. Lemos Meira, "A systematic mapping study of software product lines testing," *Information and Software Technology*, vol. 53, no. 5, pp. 407–423, 2011.

[18] S. Fabbri, E. Hernandes, A. Thommazo, A. Belgamo, A. Zamboni, and C. Silva, "Using information visualization and text mining to facilitate the conduction of systematic literature reviews," in *ICEIS*, 2012, pp. 243–256.

[19] S. Becker, L. Grunske, R. Mirandola, and S. Overhage, "Performance Prediction of Component-Based Systems - A Survey from an Engineering Perspective," in *Architecting Systems with Trustworthy Components*, ser. LNCS, vol. 3938. Springer, 2004, pp. 169–192.

[20] D. Mairiza, D. Zowghi, and N. Nurmuliani, "An investigation into the notion of non-functional requirements," in *SAC*, 2010, pp. 311–317.

[21] ISO/IEC 25000, "Software product Quality Requirements and Evaluation (SQuaRE), Guide to SQuaRE," 2011.

[22] M. Mari and N. Eila, "The impact of maintainability on component-based software systems," in *EUROMICRO Conference*. IEEE, 2003, pp. 25–32.

[5] INES - http://www.ines.org.br

View publication stats