The Journal of Systems and Software 103 (2015) 267–280

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Software requirements selection and prioritization using SBSE approaches: A systematic review and mapping of the literature

Antônio Mauricio Pitangueira[a,c,*], Rita Suzana P. Maciel[a], Márcio Barros[b]

[a] Federal University of Bahia, Computer Science Department, Brazil
[b] Post-graduate Information Systems Program, Unirio, Brazil
[c] Federal Institute of Bahia, Brazil

ARTICLE INFO

Article history:
Received 30 November 2013
Revised 15 September 2014
Accepted 24 September 2014
Available online 2 October 2014

Keywords:
Requirements selection
Requirements prioritization
Systematic review

ABSTRACT

The selection and prioritization of software requirements represents an area of interest in Search-Based Software Engineering (SBSE) and its main focus is finding and selecting a set of requirements that may be part of a software release. This paper presents a systematic review and mapping that investigated, analyzed, categorized and classified the SBSE approaches that have been proposed to address software requirement selection and prioritization problems, reporting quantitative and qualitative assessment. Initially 39 papers returned from our search strategy in this area and they were analyzed by 18 previously established quality criteria. The results of this systematic review show which aspects of the requirements selection and prioritization problems were addressed by researchers, which approaches and search techniques are currently adopted to address these problems, as well as the strengths and weaknesses in this research area highlighted from the quality criteria.

1. Introduction

Software requirements (SR) express the needs and constraints placed on a software product that contribute to the solution of some real-world problem, and in the software context, the SR process is concerned with the elicitation, analysis, specification, and validation of software requirements as well as the management of requirements during the whole life cycle of the software product (SWEBOK, 2014).

This process is crucial in software development because poor or lack of understanding of users' requirements increases the risk of not meeting users' needs (Sim and Brouse, 2014). In other words, the success of a software product depends on how well it covers the needs of users (Kheirkhah and Deraman, 2008).

Consequently, requirements engineering is extremely important to obtain this success because it is concerned with the relationship of these factors to precise specifications of software structure, behavior, and to their evolution over time and across software families (Zave, 1997). Requirements engineering processes goals are to create and maintain system requirement artifacts. During this process, multiple activities are performed, such as the analysis and management of changes throughout the software development life cycle (Sommerville, 2007). According to Azmeh et al. (2013), the success of this process plays a crucial role in the success of the whole software project and a part of this is achieved by the good selection of pertinent stakeholders and by the proper understanding of their particular needs.

Nowadays, software is becoming complex; the number of requirements that must be met is increasing and, consequently, software development processes are commonly performed in an incremental way. An increment is composed of a set of requirements that form an operational product of software, for the user (Pressman, 2002). Thus, the requirements engineer is faced with a scenario that will require several decisions, depending on many constraints (such as budgetary limitations and technical issues) and specifications regarding the selection of requirements that will be part of the next version of the software. Consequently, the main objective in selecting software requirements is to identify optimal choices for a set of requirements and to exploit trade-off decision-making to satisfy the demands of stakeholders, while at the same time making sure that there are sufficient resources to undertake the selected task (Zhang, 2010).

In a requirement engineering process, a human expert faces a scenario in which there are several decisions and considerations to be taken into account, such as requirement interdependencies,

* Corresponding author at: Federal University of Bahia, Computer Science Department, Brazil. Tel.: +55 7191582653.
E-mail addresses: antoniomauricio@ufba.br, amspitangueira@gmail.com, antoniomauricio@ifba.edu.br (A.M. Pitangueira), ritasuzana@dcc.ufba.br (R.S.P. Maciel), marcio.barros@uniriotec.br (M. Barros).

http://dx.doi.org/10.1016/j.jss.2014.09.038
0164-1212/© 2014 Elsevier Inc. All rights reserved.

project costs, different kinds of customers, and budget constraints. An analysis of software requirements, selection and prioritization of them is important because mistakes and misunderstandings at this early stage in the software development lifecycle can be extremely costly (Zhang, 2010).

Several approaches have been proposed in literature to support selecting and prioritizing requirements in software development projects (Karlsson et al., 1998), such as Analytical Hierarchy Process (Saaty, 1980), Cost-Value (Karlsson and Ryan, 1997), B-tree (Heger and Dominique, 2004), and other traditional methods (Aasem et al., 2010; Wiegers, 1999; Felows and Hooks, 1998; Boehm et al., 2001). Some of these approaches use techniques that can assist in finding a better set of software requirements according to a set of goals and constraints.

Recently, non-exhaustive and automated search methods have emerged as an alternative strategy to solve some problems of software requirements selection to meet the demands of the users of software. These methods are being addressed by the Search-Based Software Engineering (SBSE) field. This field is an approach to software engineering in which search-based optimization algorithms are used to identify optimal or near optimal solutions and to yield insight (Harman, 2007a,b). According to Harman et al. (2009a,b), SBSE is attractive when compared with other approaches, because it offers a range of adaptive automated and semi-automated solutions in situations typified by large complex problems with multiple, competing and conflicting objectives. SBSE has some important advantages in comparison to former techniques, including scalability, practicality, generality, robustness, and insight-richness (Harman et al., 2012; Harman, 2007a,b). Due to these advantages, SBSE approaches have been applied to problems throughout the Software Engineering lifecycle, for example in testing and debugging, management, design tools and techniques, maintenance, and requirements.

Regarding the process of software requirement selection and prioritization, a well-known strategy in SBSE is the Next Release Problem (NRP). According to Bagnall et al. (2001), the objective of NRP is to select a set of requirements that are deliverable within a company budget and which meet the demands of their customers. This objective allows stakeholders to receive an increment that is a complete system as soon as possible and ensures that the most important requirements are delivered first. Thus, new software requirements can be released during each increment according to stakeholder's feedback (Zhang, 2010).

Currently several types of these techniques have been proposed and significant results have been obtained in the selection process of software requirements. It is therefore necessary to understand how applications are being used in this area, what techniques are being used, the type of modeling to use in addressing this selection problem and what the current trends are in this area focusing on requirements selection and prioritization. As there is still no consensus about which techniques to apply, a systematic review was performed in order to obtain a better understanding and comprehension about software requirements selection and prioritization and trends in this area.

This paper presents a systematic review and a mapping of the literature, that followed the guidelines established by Kitchenham (2004), extending the results presented in Pitangueira et al. (2013). We are interested in understanding what has been proposed to solve problems related to the selection and prioritization of requirements using Search-Based Software Engineering. While the first version of the paper covers the period from 2001 to 2012, the present work covers the period from 2001 to 2013, including an updated search in the major conference for the area (SSBSE-Symposium on Search Based-Software Engineering and SBSE repository), an updated search in online digital libraries, and the inclusion of important conferences in Requirements Engineering. These added up to 39 studies from which new and significant results were obtained. Furthermore, the present version includes some hypothesis to be analyzed and verified. Finally, we also conducted a systematic mapping study to provide an overview of the area, identifying the quantity and type of research and results (Petersen et al., 2008).

The paper is structured as follows: Section 2 presents related work about software requirements selection and prioritization; the next section has some background about requirements selection and prioritization with the SBSE approach. Section 4 formally presents the methodology of the systematic review and mapping process. The results obtained and discussions about these results are described in Section 5. Finally, in Section 6 conclusions are drawn and future work is proposed.

2. Related work

At the time of our work, we were aware of a previous study which had conducted a systematic review in a similar area to ours. Svahnberg et al. (2010) investigated which strategic release planning models had been proposed, the degree of empirical validation supporting these models, the factor driving requirements selection, and whether they are intended for a bespoke or market-driven requirements engineering context.

To achieve these objectives, four research questions were established and two digital libraries were used to find previous works in the field: IEEE Xplore and ACM Digital Library. Regarding quality criteria, the authors proposed seven criterions and followed the protocol developed by Kitchenham (2004). In the end, 28 studies were selected and analyzed. According to the authors, the main findings were (1) that most of the methods analyzed focused on a limited set of requirements selection factors, with an emphasis on hard constraints (technical, budget and cost, resource, effort, and time-related constraints); (2) approximately half of the models were validated in the industry, while the other half was validated in the academia, being 80% case studies; (3) almost all models were intended for market-driven software development; and (4) most of the release planning models had been proposed in academic papers.

Our work is the similar to the systematic review presented by Svahnberg et al. (2010), as we follow the protocol proposed by Kitchenham. However, the present work has some different characteristics. First, we focused on studies using the SBSE approach and included the subject of requirement prioritization in our work. Besides proposing a systematic review, we conducted a mapping of the literature, checking the evolution of the SBSE area with the focus proposed in our work, showing the main research in the area, and to find out in which forum the studies are published. Regarding the systematic review, we proposed three main research questions and formulated three hypotheses to be discussed. Regarding the criteria to evaluate the studies, we defined 18 quality criteria, divided into five categories, which address the validity. Finally, some specific results were shown and discussed, and the gaps and tendencies in the area were explained.

3. Software requirements selection and prioritization

In the context of software development, the systematic review conducted here focuses on the selection of a set of requirements for the release of the next version of software. This approach was launched when Bagnall et al. (2001) formulated the "Next Release Problem" (NRP) as a search problem. This problem is illustrated in Fig. 1, which can be identified by a set $R = \{r_1, r_2, r_3, \ldots, r_n\}$ of n possible software requirements, which are offered by another set $C = \{c_1, c_2, c_3, \ldots, c_m\}$ formed by m customers.

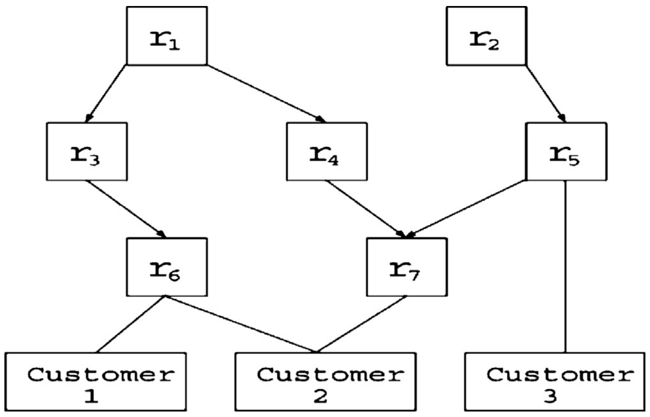


Fig. 1. The structure of the next release problem (Bagnall et al., 2001).

With these two sets, Bagnall assumed that there is a cost vector for the requirements in R called $cost = [cost_1, cost_2, cost_3, \ldots, cost_n]$. Each cost $cost_i$ is associated to fulfilling the requirement r_i (Harman et al., 2012). As each customer has a degree of importance for the company, there exist a set of relative weights $W = \{w_1, w_2, w_3, \ldots, w_n\}$ associated with each customer in C. In several situations, the same requirement is desired by more than one customer, although its importance may differ from customer to customer (Del Sagrado et al., 2011). Thus, the importance that a requirement r_i has for customer j is given by $value(r_i, c_j)$ where a value higher than 0 means the customer c_j needs the requirement r_i and 0 otherwise (Harman et al., 2012). Under these assumptions, the overall satisfaction or score is measured as a weighted sum of importance values for all the customers and this can be formalized as:

$$score_i = \sum_{j=1}^{m} w_j \cdot value(r_i, c_j). \tag{1}$$

and a solution vector $x = \{x_1, x_2, \ldots, x_n\} \in \{0, 1\}^n$, a binary string representing a subset of R. In general, the single-objective NRP can be modeled as follows:

$$\text{Maximize} = \sum_{i=1}^{n} x_i \cdot score_i \tag{2}$$

$$\text{subject to} = \sum\nolimits_{i=1}^{n} \cos t_i \cdot x_i \le B \tag{3}$$

where B is the budget designated by the company.

Consequently, NRP consists of selecting a set among all the requirements of a software package so that the cost, in terms of money or resources, of fulfilling these requirements is minimum and the satisfaction of all the users of that system is maximum (Durillo et al., 2009). Thus, it ensures that the most important requirements are released first and that the maximum attainable benefits of the new system are gained earliest (Zhang et al., 2008).

From this initial comprehension of the problem, new studies have emerged in the area, with a more specific focus. They can be divided into three dimensions, depending on the problem addressed and the subject specified by the authors of the papers in the studies selected: (1) regarding Release – Next Release Problem, Multi-Objective Next Release Problem, and Release Planning; (2) regarding Requirements Interaction Management; and (3) regarding Prioritization.

The Multi-Objective Next Release Problem (MONRP) is a multi-objective optimization version of NRP (Harman et al., 2009a,b). Normally, in MONRP there are two or more possibly competing objectives that the software engineer wishes to optimize, such as maximizing the total satisfaction of customers and minimizing the total cost of including new features into a software package (Zhang et al., 2007; Durillo et al., 2009).

Release Planning (RP) involves decision-making about what new features and change requests should be implemented, and in which release of a software system, and can be investigated from two dimensions – "what to release" and "when to release" (Saliu and Ruhe, 2007). More specifically, RP includes the assignment of requirements to releases so that all technical, risk, resource, and budget constraints should be fulfilled (Greer and Ruhe, 2004). In this case, new requirements or changes to requirements can be introduced during each increment according to stakeholders' feedback (Zhang, 2010). According to Harman et al. (2012), the NRP and RP differ slightly from each other because the problem of choosing the optimal set of requirements to include in the next release of a software system has become known as the Next Release Problem (NRP) and the activity of planning for requirement inclusion and exclusion has become known as release planning.

In specific contexts, some requirements might have technical, structural or functional correlations that need to be fulfilled together or separately. The analysis and management of dependencies among requirements is called Requirements Interaction Management (RIM) (Zhang et al., 2013). These relations among requirements can be seen, for instance, in requirements interdependencies (Carlshamre et al., 2001), requirement selection and scheduling (Li et al., 2007), and requirements interaction (Del Sagrado et al., 2011).

Finally, the process of requirements prioritization is used to indicate an order for the implementation of the requirements (Lima et al., 2011), and can be designed as an a *priori* or as an a *posteriori* process (Tonella et al., 2010). According to Firesmith (2004), the purpose of requirements priority can be to (1) determine the relative necessity of the requirements (while all the requirements are mandatory, some are more critical than others); (2) improve techniques through negotiation and consensus building to select potential requirements; and (3) schedule the implementation of selected requirements, determining what capabilities are implemented in what increment.

3.1. SBSE techniques applied to the software requirements selection and prioritization

To obtain solutions for software requirements selection and prioritization through SBSE approaches, there is a need for techniques that assist in the process of understanding, analyzing, and interpreting results obtained for the inclusion of requirements for the next version of the software. The types of techniques can be differentiated according to the way they approach the problem, e.g. single or multiple objective (Harman et al., 2012). In the simplest cases (single objective), a fair comparison between some algorithms (for example, simulated annealing and Hill Climbing), observing the amount of effort required by each search and elementary statistical analysis, can be used to analyze the best solution for a set of requirements (Harman et al., 2012) to here.

However, in practice, a software engineer is more likely to have many conflicting objectives to address when determining the set of requirements to include in the next release of the software. As such, the multiple objective formulation is more likely to be appropriate than the single objective NRP (Zhang et al., 2007). Therefore, when the goal of modeling the problem is the optimization of multiple objectives and multiple constraints, there is a need for more advanced techniques to improve the search for solutions because, contrary to single objective optimization, the solution of a multi-objective problem is not a single point, but a set of solutions. One technique is the use of Pareto Optimal, in which several optimization objectives are combined, but without needing to decide which takes precedence over the others (Harman et al., 2012; Harman, 2007a,b).

Given its efficiency, several studies have emerged using this technique in software requirements (Durillo et al., 2009; Saliu and Ruhe, 2007; Zhang et al., 2007).

Another type of technique that has been widely used by researchers in this area is Genetic Algorithm (GA), a bio-inspired search algorithm based on the evolution of the collection of individuals resulting from natural selection and natural genetics (Goldberg, 1989). According to Talbi (2009), a GA usually applies a crossover operator to two solutions which play a major role, plus a mutation operator that randomly modifies the individual contents to promote diversity.

One of the most commonly used is NSGA-II developed by Deb et al. (2002), which is often used in complex problems involving the selection and prioritization of requirements (Zhang and Harman, 2010; Finkelstein et al., 2008).

Another way of approaching NRP problems is the use of hybridization. According to Harman et al. (2012), hybridization with non-SBSE techniques can be beneficial, for example, in the case of Greedy algorithm (Talbi, 2009) which has been used to inject solutions into other evolutionary algorithms in order to accelerate the process of finding solutions. According to Grosan and Abraham (2007), some of the possible reason for hybridization are: (1) to improve the performance of the evolutionary algorithm, for instance, their speed of convergence; (2) to improve the quality of solutions obtained through executing the evolutionary algorithm; and (3) to incorporate the evolutionary algorithm as part of a larger system.

4. Systematic and mapping review methodology

A systematic review (SR) is a technique to identify, evaluate and interpret relevant research in a particular area of interest, a research question or a specific phenomenon of interest (Kitchenham, 2004). More specifically, systematic reviews provide a concise summary of evidence available regarding a given area of interest. This approach uses explicit and rigorous methods to identify, critically evaluate and synthesize relevant studies on a particular topic (Dyba et al., 2007). There are several reasons for performing a systematic review (Kitchenham, 2004), for example to summarize the available evidence regarding a treatment of a technology; identify gaps in current research in order to suggest areas to promote new research and provide a framework in order to properly position new research activities.

A SR usually comprises the steps of (a) planning (identification of the need for a review and development of a review protocol), (b) conducting (identification of research, selection of primary studies, study quality assessment, data extraction and data synthesis) and (c) report writing.

In addition to this systematic review, we also noticed the need to categorize, identify and classify the studies selected, by showing a visual and quantitative summary of the area. These tasks are related to a mapping study and aims to show how the studies are distributed in the research area.

Specifically, a mapping study was conducted to provide an overview of the software requirements selection and prioritization using SBSE approach. According to Petersen et al. (2008), a systematic mapping study is a research method that categorizes research reports and results that have been published, providing a visual summary of its results. The analysis is made on the publications in a research field and the results can be used to answer specific research questions. (Petersen et al., 2008) assert that to conduct a mapping study requires less effort and provides a more coarse-grained overview. The results coming from the mapping study subsidize the research question formulated in the systematic review, showing specific quantities and tendencies.

For this work, the goal of our systematic mapping was to check the evolution of the SBSE area with the focus on software requirements selection and prioritization, to identify the main researchers on the area and to find out in which forum the studies are published.

This systematic review and mapping is for a PhD project, showing the strengths and weaknesses of software requirements selection and prioritization using SBSE approaches and identifying possible research trends and fields of interest. The phases of the work are demonstrated and discussed. Two researchers participated in the process: a doctoral student who was responsible for carrying out the three steps outlined above, and a senior researcher who was responsible for validating the review protocol and monitoring all the steps comprising the systematic review.

4.1. Planning the review – fundamental questions

A systematic review requires a well-specified protocol that describes the dynamics of the process. One of the most important activities in the planning stage is the proper formulation of a research question or specific questions that must be answered by the systematic review. All other phases of the process are dependent on this formulation (Dyba et al., 2007).

The aim of our study was to examine the current state of software requirements selection and prioritization methods, focusing on Search-Based Software Engineering methods. From this, our research question (RQ) for this systematic review is:

> "*What is the state of the art for the application of search-based optimization in the process of software requirements selection and prioritization?*"

This is a topic of scientific and professional importance because this is becoming increasingly used, given the power to support the attainment of (close to) optimum solutions of a problem as complex as the selection and prioritization of requirements in software projects (Zhang et al., 2008).

The research questions are divided into mapping questions and review questions. To perform our systematic mapping, we defined the following questions:

- MQ1: How are the publications on software requirements selection and prioritization using SBSE approaches distributed across the years?
  In this question, we expect the answer to show the growth or reduction of the research area.
- MQ2: Who are the most active authors in the area?
  Through this question we expect to have an indication of the main researchers in the area, indicating a reference for related publications in future researches.
- MQ3: Which publication forums are the main targets for the research area production?
  In this question, we want to know where the papers addressing software requirements selection and prioritization can be found, as well as identifying good targets for publications of future studies.

For the systematic review, through a preliminary analysis, the following hypothesis and research questions were formulated:

Hypothesis 1. "The quantity of studies using Multi-Objective formulation problem is a recent trend and is increasing when compared to the number of studies done in single objective."

This hypothesis motivated us to construct the following research question:

- RQ1: What types of modeling are being used in selection and prioritization of software requirements with a focus on SBSE?

In this question, aspects concerning modeling relate to how the problem of software requirements selection and prioritization are covered and how they are addressed to find possible solutions. In other words, how organized the fitness function and the constraints are to form the model representation (Harman and Jones, 2001).

Hypothesis 2. "Meta-heuristics techniques are the most used by the researches to solve the problem of software requirements selection and prioritization."

According to this hypothesis, the following research question was formulated:

- RQ2: What methods are used for selection and prioritization of software requirements with a focus on SBSE?

  Regarding methods (RQ2), these can be understood as a way of obtaining solutions to the problem at hand, in general, such as meta-heuristics, hybridization or an exact method. In this case, approaches to operations research techniques can be used as a solution to the problem of selecting and prioritizing software requirements (Harman, 2007a,b). This research questions aims to know the scenario of methods usage in the problem addressed in this systematic review.

Based on the hypothesis above, we are interested to know the distribution of the algorithm's applications in software requirements selection and prioritization. Since there are a large variety of these techniques, we want to know the scenario of usage in SBSE area. Consequently, the hypothesis below was formulated:

Hypothesis 3. "In the SBSE area, a consensus of the search techniques usage does not exist".

Obviously, the research question regarding this hypothesis aims to identify the set of techniques used in SBSE. This is the question:

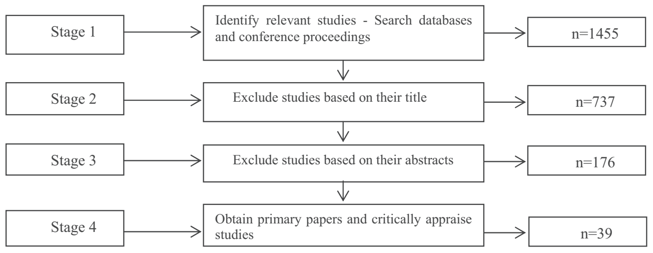
- RQ3: What search techniques are used for the process of selecting and prioritizing requirements with a focus on SBSE?

  RQ3 addresses search techniques for selecting and prioritizing software requirements. In this case, the term 'search technique' refers to the ways of tackling the problem by observing their proposed goals and objectives, for example, the use of a genetic algorithm or exact algorithm. RQ3 is derived from RQ2, showing in details the search techniques that were used in methods of approach.

These hypotheses and questions were essential for determining the structure and content of this review, and also for guiding the process because all other parts of this systematic review and mapping are derived from these issues.

4.2. Identification of research – search strategy

In order to perform an exhaustive search for primary studies, our search strategy consisted of a manual search in the major conferences and an online search in relevant digital libraries. To ensure the quality of this review, we manually searched in the major conference in this area, the Symposium on Search-Based Software Engineering (SSBSE) (2009–2013) and SBSE Repository.[1] After this, the following electronic databases were searched as we considered these to be most relevant: IEEE Xplore,[2] ACM Digital Library,[3] ISI Web of Science,[4] SpringerLink,[5] and Science Direct.[6] Additionally, we scanned the *International Requirements Engineering Conference* (RE Conference[7]) and *International Working Conference on Requirements Engineering: Foundation for Software Quality* (REFSQ)[8] because they are important conferences in the software requirements engineering area.



Stage 1
Identify relevant studies - Search databases and conference proceedings
n=1455
Stage 2
Exclude studies based on their title
n=737
Stage 3
Exclude studies based on their abstracts
n=176
Stage 4
Obtain primary papers and critically appraise studies
n=39

Fig. 2. Stages of the study selection process.

When using the online search in digital libraries, search keywords are very important to obtain good results, so they have to be chosen carefully. Finding an answer to the research question includes selecting appropriate information resources and executing an accurate search strategy. Then, based on our research question, we experimented with different search criteria using different combinations of strings. The following basic search strings were considered the most appropriate for the SR:

1. Software requirements AND selection
2. Software requirements AND prioritization
3. Search based AND requirements optimization
4. Search based AND requirements selection
5. Search based AND requirements prioritization

All these search terms were combined by using the Boolean "OR" operator, which implies that an article only had to include any one of the terms to be retrieved. That is, we searched: 1 OR 2 OR 3 OR 4 OR 5. The search terms "software requirements", "selection" and "prioritization" are derived from the terms of the specific area. The inclusion of the term "Search Based" was due to the need to restrict the search to papers that were in the area of software engineering with the SBSE approach, to avoid recovering papers that focused on requirements selection and/or prioritization in research fields other than SBSE. We applied the search terms to the titles, abstracts, and keywords of the papers in the identified electronic databases.

Due to the different functions and features of search engines, the search strings for the digital libraries were similar and we had to implement the search strategy individually for each database. This created considerable extra work, as we had to perform several trial searches only to find out how each of the databases handled different Boolean expressions. The stages of the study can be seen in Fig. 2.

In our previous paper (Pitangueira et al., 2013), 30 studies were in the final set. For the actual systematic review, we decided to rehash the query and nine new papers were found. They are added to the final publication set, which now is comprised of 39 papers.

[1] http://crestweb.cs.ucl.ac.uk/resources/sbse_repository/.
[2] http://ieeexplore.ieee.org/Xplore/home.jsp.
[3] http://dl.acm.org/.
[4] http://thomsonreuters.com/thomson-reuters-web-of-science/.
[5] http://link.springer.com/.
[6] http://www.sciencedirect.com/.
[7] http://requirements-engineering.org/.
[8] http://refsq.org/2014/past-conferences/.

Table 1
Quality criteria selected to assess SBSE approaches to software requirements selection and prioritization.

Quality criteria
General
1 – Is there a proper introduction to the article?
2 – Does it have a clear idea of the research objectives?
3 – Is there a clear statement of the results?
4 – Are there any limitations or restrictions on the results?
5 – Does the study make a projection of value for research and practice?
Internal validity
6 – Is the source code used in the study discussed and made available to other researchers?
7 – Does the article detail the procedure for collecting the data used in the experiment?
8 – Does the article present an explicit definition of the target instance (random and real)?
Conclusion validity
9 – Does the experiment consider random variations (different runs) in the algorithms presented?
10 – Is the hypothesis of the study formally presented?
11 – Are statistical inference tests used?
12 – Is there a significant basis for comparison between the techniques used and the best-known solutions?
Construction validity
13 – Does the article discuss the model adopted for the optimization process?
14 – Does the article discuss the validity of effectiveness measures?
External validity
15 – Does the study clearly show the strategy of selecting instances? (Real or randomly-generated data)
16 – Does the study treat the variation in the complexity of the instances used in the experiment?
17 – Does the study use real world instances in the experiment?
18 – Does the study present the parameter values used in the experiment?

4.3. Study selection criteria and procedures

The inclusion and exclusion criteria were defined in the review protocol that we developed for this systematic review. The included primary studies should belong to one of the following categories:

- Category A: studies proposing a new model or approach using SBSE that can be applied in software requirement selection and/or prioritization; or
- Category B: studies applying existing models and techniques to software requirement selection and/or prioritization using SBSE.

The following studies would be excluded during the study selection phase:

- Studies on software requirements selection or prioritization that do not use a Search-Based Software Engineering approach
- Studies on training, editorials, article, summaries, interviews, prefaces, news, reviews, correspondence, tutorials, poster session, workshops and panels.

Thus, studies were eligible for inclusion in our review if they presented all the characteristics from SBSE and passed the minimum quality threshold (Table 1 – Section Quality Assessment). As mentioned previously, the first query included studies published from 2001 to 2012 and only studies written in English were included. For this systematic review and mapping, new papers were found and included in the set, performing a total of 39 studies, from 2001 to 2013.

The reason for choosing studies from 2001 was due to usage optimization methods in software engineering as described in Harman and Jones (2001), when the term SBSE was created and the studies in this area were categorized in the SBSE field.

As can be seen in Fig. 2, our study selection started searching for relevant studies based on their titles, abstracts and keywords. Relevant studies are the potential candidates for primary studies and the full papers were fully analyzed. We reviewed each relevant study to determine whether it was a primary study. This was necessary because in the software requirements engineering area, title, abstract and keywords are not usually enough to determine the content of a paper. Therefore, we reviewed the full paper before making a final decision on whether to include or exclude it. More specifically, at stage 1, duplicate citations were identified and removed. In the second stage, we excluded studies that were clearly not about software engineering requirements. Titles that were clearly outside the scope of the systematic review were removed. However, in case of doubt, some papers were considered for the third stage. In this phase, studies were excluded based on their abstracts, whether their focus was not software requirements selection or prioritization using the SBSE approach. However, some abstracts are poor or misleading and several gave little information about what was in the article. Therefore, at this stage we included all the studies that shared some characteristics with our proposed SR. Finally, if the article was not clear from the title, abstract and keywords it was included for a deep quality assessment.

4.4. Quality assessment

We chose eighteen criteria to obtain and assess each primary study resulting from stage 4. Table 1 presents the quality assessment checklist used in this study.

These criteria were based on a former study (Barros and Dias-neto, 2011) in which the authors developed a systematic approach to assess experimental studies in the SBSE field. From this, we developed a checklist in which every question/criteria has a “yes” or “no” answer.

For a better understanding of the evaluation, we decided to divide these criteria into five groups. The first one refers to general aspects and structural information about the article and is to assess whether the selected work is within the context of experiments in the area of software engineering (Jedlitschka and Pfahl, 2005; Kitchenham et al., 2010; Sjoberg et al., 2005).

The other groups are related to threats to the validity of experiments in SBSE. As proposed by Wohlin et al. (2000) threats to validity are risks associated with the design, implementation and analysis of results from experimental studies. The criteria included in the group of internal validity are to evaluate if the relationship is between the observed treatment and the outcome. In other words, the experiment must ensure that it is a relationship of cause and not the result of a factor over which the researcher has no control.

The conclusion validity group contains criteria that relate treatment and outcomes, therefore statistical procedures must be used to evaluate the significance of results (Barros and Dias-neto, 2011). The construction validity group contains criteria concerned with the relationship between theory and observation (Wohlin et al., 2000). Finally, the last set of criteria deals with external validity by focusing on the generalization of the results observed in an experiment for a larger population, beyond the instances that make up the sample used in the study (Barros and Dias-neto, 2011).

In the remainder of this work, each quality criterion is identified with the Q and the number of the criteria, for example, Q1 refers to the first Quality criteria in Table 1.

5. Results and discussion

This section presents the results produced by conducting this systematic review and mapping study according to the protocol described in the previous section. Table 2 presents the papers that

Table 2
Selected studies.

2001	(1) The next release problem (Bagnall et al., 2001)
2003	(2) Quantitative studies in software release planning under risk and resource constraints (Ruhe and Greer, 2003)
2004	(3) Software release planning: an evolutionary and iterative approach (Greer and Ruhe, 2004)
2005	(4) Supporting software release planning decision for evolving systems (Saliu and Ruhe, 2005a) (5) Determination of the next release of a software product: an approach using integer linear programming (Van Den Akker et al., 2005a) (6) Software release planning for evolving systems (Saliu and Ruhe, 2005b) (7) Flexible release planning using integer linear programming (Van Den Akker et al., 2005b)
2006	(8) Search based approaches to component selection and prioritization for the next release problem (Baker et al., 2006)
2007	(9) The multi-objective next release problem (Zhang et al., 2007) (10) Bi-objective release planning for evolving software (Saliu and Ruhe, 2007) (11) Integrated requirement selection and scheduling for the release planning of a software product (Li et al., 2007)
2008	(12) Software product release planning through optimization and what-if analysis (Van Den Akker et al., 2008) (13) A systematic approach for solving the wicked problem of software release planning (Ngo-The and Ruhe, 2008)
2009	(14) Search based data sensitivity analysis applied to requirement engineering (Harman et al., 2009a) (15) A study of the multi-objective next release problem (Durillo et al., 2009) (16) A new approach to the software release planning (Colares et al., 2009) (17) A search based approach to fairness analysis in requirement assignments to aid negotiation, mediation and decision making (Finkelstein et al., 2009)
2010	(18) Search based optimization of requirements interaction management (Zhang and Harman, 2010) (19) Using interactive GA for requirements prioritization (Tonella et al., 2010) (20) An integrated approach for requirement selection and scheduling in software release planning (Li et al., 2010) (21) Ant colony optimization for the next release problem (Del Sagrado et al., 2010) (22) A hybrid ACO algorithm for the next release problem (Jiang et al., 2010a) (23) Approximate backbone based multilevel algorithm for next release problem (Jiang et al., 2010b)
2011	(24) Comparing the performance of metaheuristics for the analysis of multi-stakeholder tradeoffs in requirements optimization (Zhang et al., 2011) (25) Software next release planning approach through exact optimization (Freitas et al., 2011) (26) A study of the bi-objective next release problem (Durillo et al., 2011) (27) An ant colony optimization approach to the software release planning problem with dependent requirements (Souza et al., 2011) (28) A fuzzy approach to requirements prioritization (Lima et al., 2011)
2012	(29) Software requirements selection using quantum inspired elitist multi-objective evolutionary algorithm (Kumari et al., 2012) (30) Solving the large scale next release problem with a backbone-based multilevel algorithm (Xuan et al., 2012) (31) Evolutionary approaches for multi-objective next release problem (Cai et al., 2012) (32) A multiobjective optimization approach to the software release planning with undefined number of releases and interdependent requirements (Brasil et al., 2012) (33) Multi-objective optimization approaches to software release time determination (Li et al., 2012)
2013	(34) Empirical evaluation of search based requirements interaction management (Zhang et al., 2013) (35) Interactive requirements prioritization using a genetic algorithm (Tonella et al., 2013) (36) A scenario-based robust model for the next release problem (Paixão and Souza, 2013a) (37) A recoverable robust approach for the next release problem (Paixão and Souza, 2013b) (38) Hill climbing and simulated annealing in large scale next release problem (Mausa et al., 2013) (39) A hybrid of decomposition and domination based evolutionary algorithm for multi-objective software next release problem (Cai and Wei, 2013)

were selected after applying the inclusion and exclusion criteria depicted in our protocol. As can be seen, the selected studies are sorted by year of publication, and paper title.

5.1. Systematic mapping report

In this section, the answers to our mapping questions (MQ) are presented. Fig. 3 presents the results, on regard of how the publications were distributed across the years (MQ1). Considering that the area is recent in the Software Engineering field, the low number of publications in the first years and the existence of some peaks and troughs in the total period is normal. It is important to notice the growth of the number of studies from 2009 which can be related to moment when the Symposium on Search-Based Software Engineering was created.

Regarding the authors that published frequently in this area (MQ2), Fig. 4 shows this scenario. It is important to highlight that, sometimes, the same paper is published by more than one author. For instance, Harman and Zhang published various papers. Another important relevant data, the paper titled "The Next Release Problem" (Bagnall et al., 2001) was the most cited study by the papers analyzed in this systematic mapping review.

Fig. 5 shows the results for question MQ3, which is related to the main forums of accepting publications related to the area researched in this work. Regarding symposiums and conferences, as expected, the SSBSE (*Symposium on Search Based Software Engineering*) has the majority of publications, followed by GECCO (*Genetic and Evolutionary Conference*) and REFSQ (*Working Conference on Requirements Engineering: Foundation for Software Quality*). Amongst Journals, the *Journal of Information Software and Technology* was the most used vehicle for publication, followed by the Requirements Engineering Journal.

- SSBSE: Symposium on Search Based Software Engineering
- JISWT: Journal of Information Software and Technology
- GECCO: Genetic and Evolutionary Conference
- REQENG: Requirements Engineering Journal
- REFSQ: Working Conference on Requirements Engineering: Foundation for Software Quality
- IJCA: Informatics Journal of Computer Applications
- WOES: Workshop em Otimização em Engenharia de Software
- SOFTCOMP: Soft Computing Journal
- EMPSOFENG: Empirical Software Engineering Journal
- ICMS: International Conference on Software Maintenance
- ISESE: International Symposium on Empirical Software Engineering
- ESEC/FSE: European Software Engineering Conference/ACM SIGSOFT Symposium on the Foundations of Software Engineering

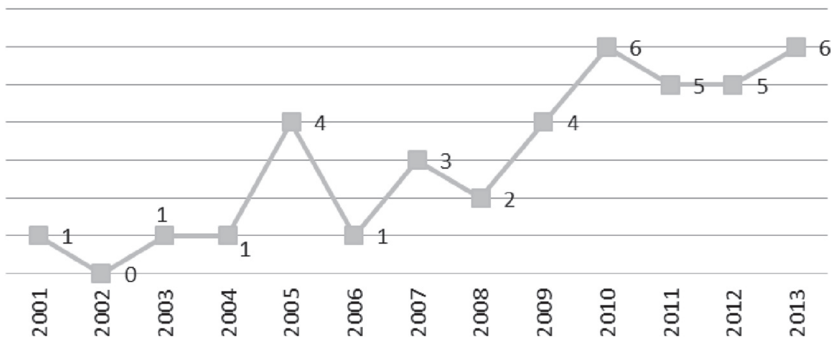


Fig. 3. Publication tendency by year.

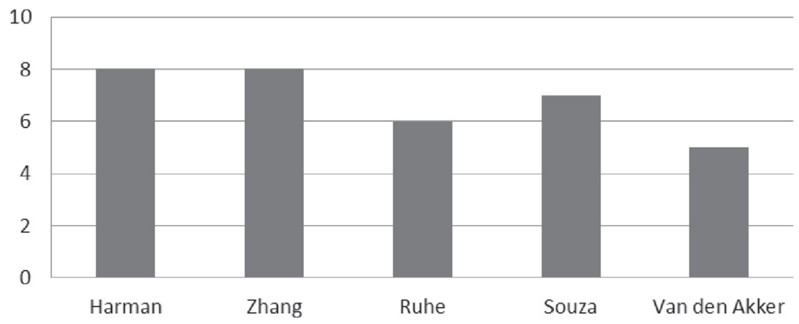


Fig. 4. Top five authors by publication.

- NASE SEW: IEEE/NASA Software Engineering Workshop
- SEDM: International Conference on Software Engineering and Data Mining
- ICAESM: International Conference on Advances in Engineering, Science and Management
- CAiSE: International Conference on Advanced Information Systems Engineering
- TRANSWENG: IEEE Transaction on Software Engineering Journal
- INNSSENG: Innovation Systems and Software Engineering Journal
- EUROCON: European Conference
- ICCA: IEEE International Conference on Control and Automation
- COMSOFT: Computing and Informatics Journal
- ENTRINSCON: Enterprise Information Systems Conference
- APOR: Asia-Pacific Journal of Operations Research

Fig. 6 presents a summary, highlighting the temporal distribution of our selected papers (from 2001 to 2013), separated by publication type (workshop, conference, journal and symposium) and showing the number of publications by year and type of event.

With all these results, it is possible for a researcher or Software Engineer to identify the specific forums of publications and find the papers desired, knowing the authors that have an important participation in software selection and prioritization using SBSE. This data subsidizes the results presented in the systematic review, where specific scenarios are shown and the papers are evaluated according to quality criteria.

5.2. *Systematic review report*

In this section, the systematic review results are presented including quantitative and qualitative results. Fig. 7 shows the results regarding the type of problem formulation addressed in software requirements selection (Research Question 1), highlighting three different kinds: studies that used only a single objective, studies that used only multiple objectives and studies that did experiments with both.

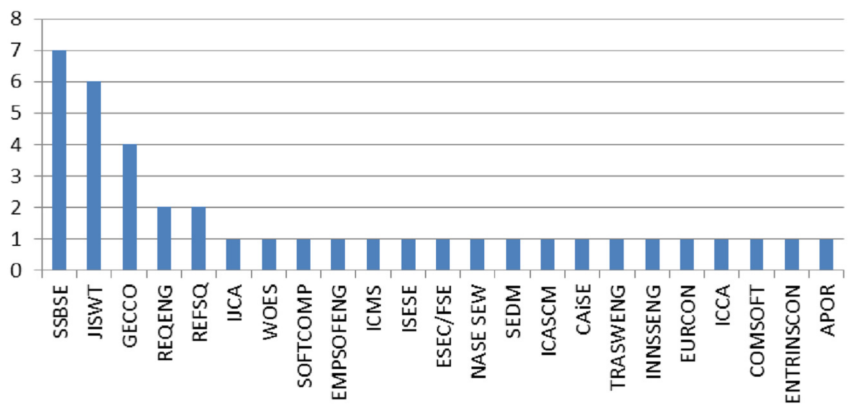


Fig. 5. Publication per forum.

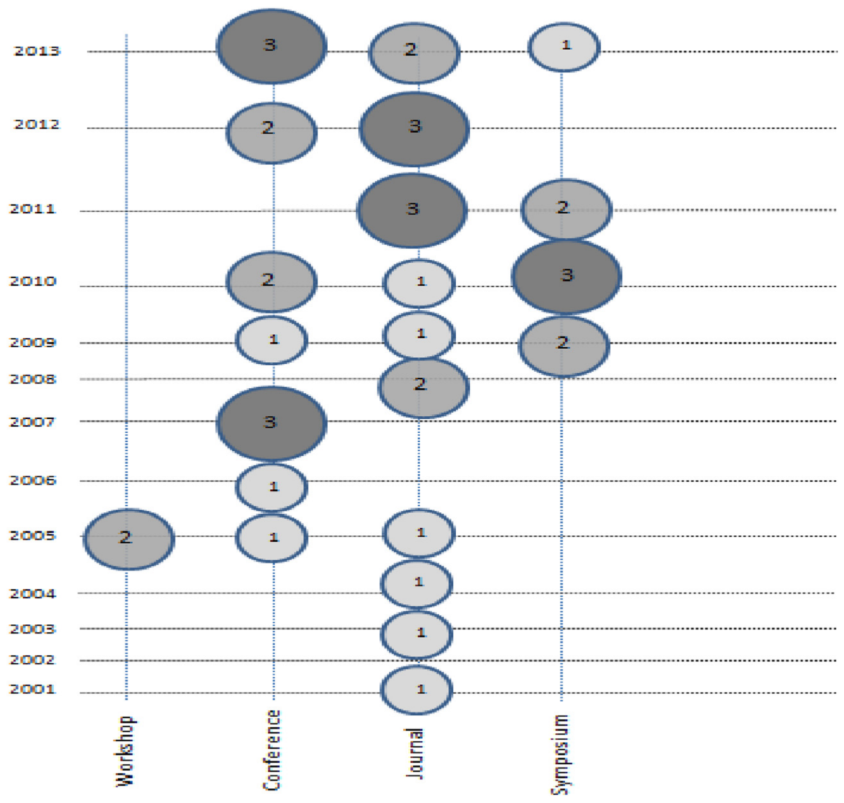


Fig. 6. Distribution by type of events.

The results show that the single-objective formulation is the most used problem formulation in software requirement selection and prioritization. However, since the paper entitled "The Multi-Objective Next Release Problem" (Zhang et al., 2007) was published, the number of papers using the multi-objective formulations increased. Authors generally agree that this formulation is more effective and closer to real-world software engineering than the single-objective one.

The main advantage of MONRP is the ability to combine multiple, conflicting objectives, such as to maximize customers' satisfaction and minimize the total effort involved in the development of the selected requirements. In addition, and distinct to the single-objective optimization, the solution of MONRP is not a single point, but a set of solutions that can be analyzed, showing different possibilities, and scenarios for the decision-maker. Recently, some interesting applications of MONRP have been used, such as, the use for software release time determination, considering maximizing reliability and minimizing cost (Li et al., 2012), and application based on stakeholder satisfaction, business value and risk management (Brasil et al., 2012).

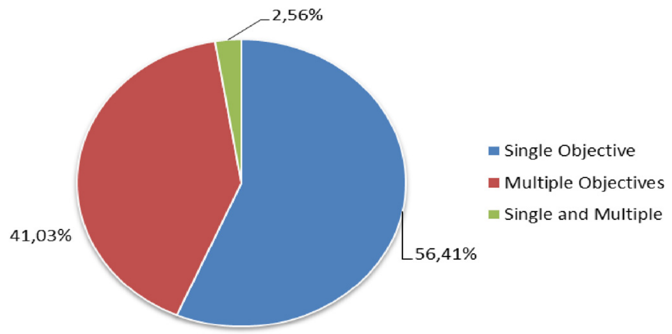


Fig. 7. Distribution of problem formulation.

On regard of problem formulation, most papers did not consider the uncertainties related to requirement importance and cost when solving the NRP through optimization techniques. Approaches that take into consideration such uncertainties as part of the optimization process are known as Robust Optimization. It is an operational research framework that identifies and quantifies uncertainties related to the problem and builds a model that seeks feasible solutions for every realization of the uncertainties (Beyer and Sendhoff, 2007). In the selected data set, two papers related to this approach were found: one about a scenario-based robust model for the NRP (Paixão and Souza, 2013a,b) and the other about a recoverable robust approach for the NRP (Paixão and Souza, 2013a,b). The first proposed a novel formulation to the NRP based on scenarios, taking into account uncertainties present in input variables. In this case, uncertainties related to the requirement importance were modeled in as discrete variables, while uncertainties related to requirement cost were modeled in as continuous variables. The second paper is an extension of the first and includes a recoverable solution, that is, for a limited set of uncertainties realizations, some solutions can be made feasible or recoverable, by applying limited effort (Liebchen et al., 2009).

Regarding the constraints used in the problem formulations, most of the studies use budget and cost constraints, while a few studies include requirement dependencies, time constraints, and resource-related factors.

Regarding Hypothesis 1 – if the quantity of studies using multi-objective formulation problem is a recent trend and is increasing when compared to the number of studies done in single objective, it seems to be a recent trend, however, that some studies using single objective continue to be published, so we cannot conclude that usage of single objective formulation may be discontinued and multiple objective will be the preferred option.

Observing and analyzing the total set of papers in our systematic review, normally, when a new approach is researched, the first formulation used is the single objective and, after some conclusions

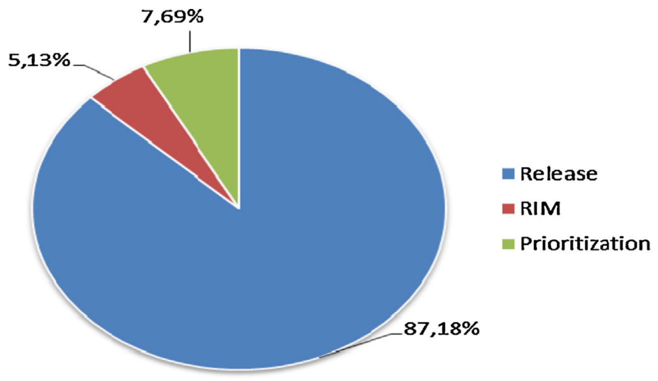


Fig. 8. Problems addressed by our selected studies.

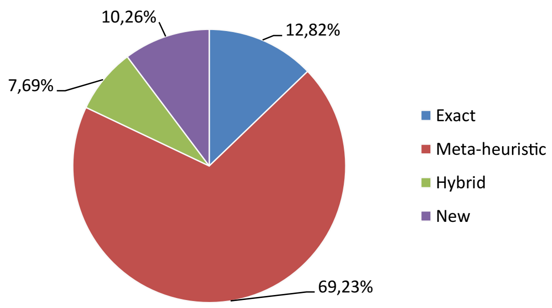


Fig. 9. Optimization methods adopted by our selected studies.

and experimentations, the usage of multi-objective formulation is used, aiming for more accurate results since, in the real world, various competing objectives appear in software engineering problems (Zhang et al., 2007).

For a better comprehension of the problem addressed, Fig. 8 displays the results. As seen in Section 3, the concepts and definitions of NRP, RP, RIM and prioritization are intertwined and interchangeable amongst themselves, depending on the viewpoint of the researcher. In our work, we decided to separate the category into three: Release, RIM and prioritization to follow the problem addressed and specified by the author in their papers.

In Fig. 8, the most frequently addressed problem is related to releases (Next Release Problem, MONRP – Multiple Objective Next Release Problem and RP – Release Planning), followed by Requirements Prioritization and, finally, RIM (Requirements Interaction Management). Next Release Problem and Multi-objective Next Release are the most used, and Release Planning has a significant participation in the problem addressed.

Regarding (RQ2), about what methods are used for selection and prioritization of software requirements with a focus on SBSE, the results can be seen in Fig. 9. They can be divided into four groups: (1) meta-heuristics, (2) exact search techniques to obtain optimal solutions, (3) hybrid methods containing both meta-heuristics and exact search techniques; and (4) a new formulation created to tackle the problem. It can be observed that meta-heuristics is arguably the most commonly applied method in experiments in search of solutions for requirements selection and prioritization problems, thus confirming Hypothesis 2. Indeed, given the complex interrelationships between requirements, generally it is quite costly to find an optimal solution. Therefore, the use of meta-heuristics may be more appropriate when there is uncertainty and difficulty in finding a single optimal solution.

Regarding hybrid methods, few studies have adopted this approach, for example, Jiang et al. (2010a,b) used a combination of Ant Colony optimization and Hill-Climbing, named HACO (Hybrid Ant Colony Optimization). According to the authors, the HACO outperformed Grasp and Simulated Annealing in terms of solution quality and running time. Other possibilities were used, such as the integration between an Integer Linear Programming and RCPSP (Li et al., 2010), and a hybrid of decomposition and dominated-based evolutionary algorithm proposed by Cai and Wei (2013), who compared their approach to NSGA-II and SPEA-2 in the MONRP obtaining better results in their hybrid proposal. These methods showed a potential for resolving real requirements selection and prioritization problems. In our opinion, as new techniques are being developed, there is always the possibility of combining them to improve the performance in finding better solutions.

Regarding the other methods adopted, the exact method is focused on integer linear programming, and new methods are new formulations created by the author to solve the problem, such as the EVOLVE method (Greer and Ruhe, 2004) and its new variations.

Finally, Fig. 10 shows the results regard to which search techniques are used for the process of selecting and prioritizing requirements with a focus on SBSE (RQ 3) and correlated with Hypothesis 3. Multiple search techniques are used in experiments to obtain good solutions. According to the data obtained, it is evident that the NSGA-II (Non-Sorted Genetic Algorithm II) is the most frequently used algorithm, followed by Genetic Algorithm, Integer Linear Programming, Simulated Annealing and Greedy.

Many authors consider NSGA-II to be the fastest and most effective solution for a search in large and complex spaces. Therefore, the choice of this method is the most used. About the experiments, some positive aspects were highlighted by some authors about some important techniques. For example, Zhang et al. (2007) presented evidence that NSGA-II is well suited to the MONRP problem formulation, and that this technique is able to find a more diverse solution set. The authors observed in their experiments that NSGA-II outperformed other techniques, for example, Pareto-GA, in finding a large and important part of Pareto front in large data scale problems.

In Durillo et al. (2011), some interesting results were presented, comparing some techniques and showing their specific advantages. In their experiments, the analysis showed that while NSGA-II was the technique computing the highest number of optimal solutions, MOCell provided the product manager with the widest range of different solutions and PAES was the fastest technique, albeit with the least accurate results. On the other hand, comparing the performance of meta-heuristics in the analysis of multi-stakeholders tradeoffs in requirements optimization, Zhang and Harman (2010)

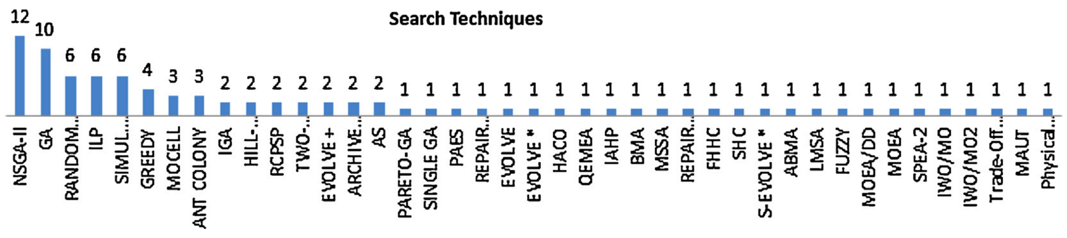


Fig. 10. Search techniques.

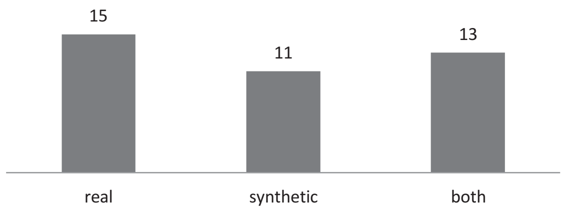


Fig. 11. Data type used in the experiments.

Table 3
Selected studies.

Data type	Paper number (according Table 2)
Real	4, 5, 6, 7, 8, 10, 12, 13, 19, 20, 21, 22, 33, 35, 38
Synthetic	2, 3, 9, 15, 16, 18, 25, 27, 28, 29, 32
Both type	1, 11, 14, 17, 23, 24, 26, 39, 31, 34, 36, 37, 39

observed that the results obtained revealed that the two-archive algorithm outperformed NSGA-II and random search in convergence as the scale of problem increase.

An important observation is related to random search usage in experiments. Despite the high occurrence of it, as seen in Fig. 10, this technique is used for the comparison of results and is not considered as a method to solve the problem of selecting and prioritizing requirements. To summarize, despite the major usage of NSGA-II, Fig. 10 shows a large diversity of search techniques used and it may indicate that there is no consensus in this area regarding the best technique to tackle the complex problem of selection and prioritization in software engineering, confirming our Hypothesis 3.

Additionally, other results are important to map the research area studied. The following figures show the results regarding data used in the experiments. Fig. 11 shows the experiments that used real data, normally industrial ones, synthetic data and some studies reported the use of real data as synthetic data in their experiments. Practically, 50% of the papers used exclusively real data. The results generated extra work in our systematic review and mapping because many papers do not explain clearly the selected instances used in their experiments, sometimes citing only a database or data used by other authors, without mentioning the type of data in explicit way.

The following table shows specifically, which papers belongs to each data type category used in the experiments (Table 3).

Regarding data scale, Fig. 12 shows the four types: small scale, medium scale, large scale and some studies that used a combination of two or more types of instance in their experiments. We decided to classify the scale using the same method adopted by (Zhang et al., 2010). In this case, the scale is proportional to the number of requirements, denoting, small scale (1–100 requirements), medium-scale (101–250 requirements), and large-scale (251 onward). Most studies used either small-scale instances or a combination of small- and medium-scale instances in their experiments.

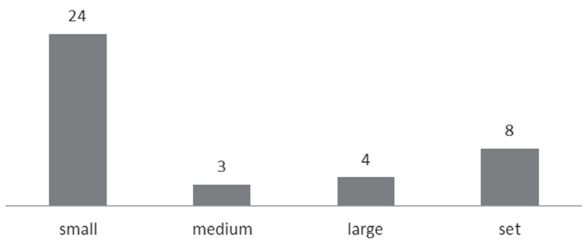


Fig. 12. Instance scale used in the experiments.

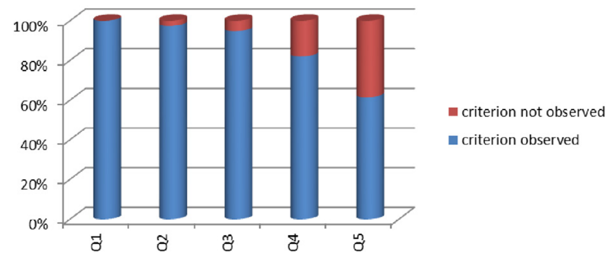


Fig. 13. Handling general validity threats in the selected studies.

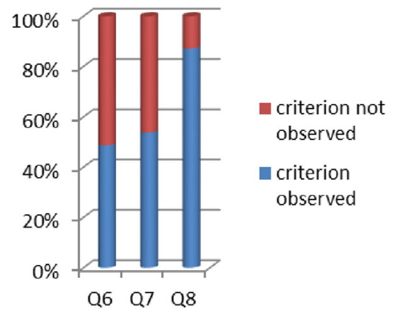


Fig. 14. Internal validity.

With respect to the constructs of quality criteria (see Table 1), regarding the general validity threats almost all items showed excellent rates (see Fig. 13). Only criteria 5 shows that practically 38% of the papers do not make a projection of the value to research and practice, or in other words, 38% of the papers do not identify new areas for the application of the research and the use in different populations.

Fig. 14 shows the results obtained regarding internal validity. These data show that nearly half of the papers do not provide the source code used in the experiments (Q6) and a meaningful quantity of the papers do not detail the procedure for collecting the data used in the experiment (Q7). Regarding question 8, definition of target instance (random or real), the majority of the papers meet this criterion.

Conclusion validity is probably the construct requiring specific attention due to the rates obtained. Fig. 15 shows the data. About Q9 (if the experiment considers random variations (different runs) in the algorithms presented), practically 38% of the experiments do not consider random variations. In the case of Q10 (the hypothesis of the study formally presented), more than 71% do not observe these criteria. Concerning the quality criterion 11 (Q11 – use of statistical inference test), only 13 papers observes this criterion, and finally, Q12 (if there is a significant basis for comparison between the techniques used and the best-known solutions), 48.71% meet this quality criterion.

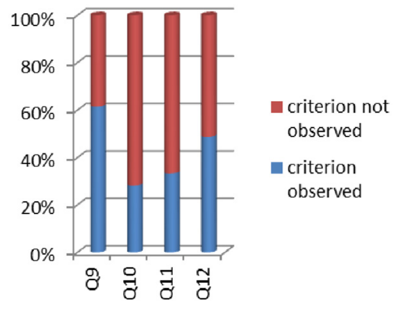


Fig. 15. Conclusion validity.

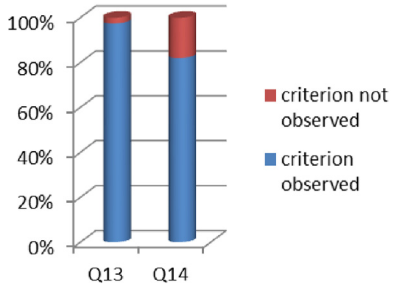


Fig. 16. Construction validity.

In relation to items that address construct validity (Fig. 16), good numbers are obtained. 38 items match criterion 13 (the article discusses the model adopted for the optimization process) and 32 meet criterion 14 (the article discusses the validity of effectiveness measures). Finally, about external validity, Fig. 17 shows the results obtained. In this figure, one can identify that criterion 18, concerning the presentation of the parameter values used in the study, all the studies met this however, in Q15 (if the study clearly shows the strategy of selecting instances (real data or randomly generated) only 43.5% observed this and for Q16 (if the study deals with the variation in the complexity of the instances used in the experiment) 46% did this. About whether the study used real world instances in experiment or not (Q17), more than 20 papers used real world instance.

After the presentation of these results, the qualitative data extracted was analyzed, showing recent trends and possibilities, strengths and weaknesses in the field. Initially, the results indicated a growing number of publications from 2009 (Fig. 3). In this case, this is the result of the increase in the total publication number, due to the creation of the International Symposium on Search-Based Software Engineering, where the program adopted multiple areas, and specifically in 2010 when an area was dedicated to requirements selection and prioritization alone.

Overall, the quality criteria discussed here and found by the systematic review show which criteria should be best observed for more robust and reliable experiments. As for the qualitative aspect of the papers, some important observations can be summarized:

(a) A recent trend to use multi-objective modeling
(b) Incorporation of new search techniques
(c) Focus on randomly generated data
(d) Little comparison between real and experimental data
(e) Lack of generalization of the models
(f) Experiments with data on a small scale
(g) No categorization of requirements (functional or nonfunctional)
(h) Few studies consider interdependency between software requirements
(i) Modeling consideration only restriction related to budget or costs.

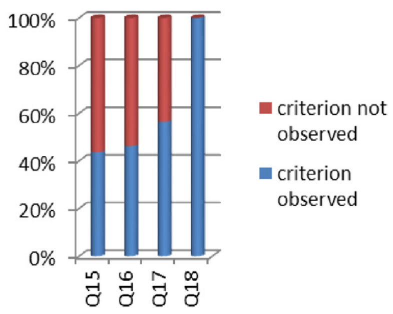


Fig. 17. External validity.

According to the qualitative and quantitative results obtained in our systematic review and mapping literature, we have identified some gaps and trends in the area and propose some points of interest that can be taken into consideration for new research undertaken by the scientific community.

Firstly, the usage of experiment in large scale is a good possibility for the experiments in the area because, normally in real world, many software projects use a quantity of data in large scale. Regarding experiment format, most studies use only experimental data on a small scale (Fig. 12), which restricts the application of results in real situations, on larger scales and generalization.

Some items are being researched, such as: (i) the adoption of multi-objective modeling and (ii) the usage of hybrid methods to get better solutions, for example, the hill climbing local search in conjunction with ant colony optimization (Jiang et al., 2010a). In the studies analyzed in this systematic review, the distribution of problem formulation (Fig. 7) showed that the use of multiple objectives is less than single objective. Since most problems in software requirements selection have multiple, conflicting objectives, the use of multi-objective formulation can be an option in obtaining some results which are closer to the reality of software engineers.

In relation to hybridization, Fig. 9 showed few uses of this method. For us, hybridization needs further research and can obtain some positive solutions. Therefore, there is the possibility of using exact optimization (e.g. mathematical programming and constraint programming) combined with several, parallel metaheuristics, such as evolutionary algorithms; to improve the results obtained in a search.

The interdependency between requirements is another point of interest, and when the studies selected are analyzed, the most common interaction types are precedence, value-related, cost-related, AND operator and exclusive OR. It is a good characteristic to explore in release planning and requirement interaction management and needs further research to analyze the effects of the interdependency in software development process.

Recently, a multilevel approach has been proposed to solve large-scale NRP instances, exploring iteratively the search spaces by multilevel reductions and refinements (Jiang et al., 2010a,b). It is a recent trend in the SBSE field and may be further explored in cases of experiments on a large-scale data.

In terms of innovation for the area, three factors may contribute to performing experiments closer to the reality of a software engineer:

(i) Conduct experiments in SBSE that categorize and differentiate functional and non-functional requirements given that there are differences between them, thus, further experiments can be performed and the results obtained may correspond to the reality of the software industry. Currently, there are no studies in NRP that achieve this differentiation and its consequences,
(ii) Add user judgment in modeling the problem of selecting and prioritizing requirements. This may lead to better results and experiments closer to the user requirements,
(iii) The inclusion of new restrictions could make the experiments closer to the reality of the software industry, where uncertainties, risks, factors related to project deadlines, stakeholders' influence factors, resource consumption factors, and quality constraints can have a significant impact on software development.

A promising area that can be observed for software requirements selection is robust optimization, where various aspects of risks and uncertainties are added to optimization models. According to Ben-Tal and Nemirovski (2002), Robust Optimization is a modeling methodology, combined with computational tools, to process optimization problems in which the data are uncertain and is only known to belong to some uncertainty set.

This method can be very effective in the engineering process requirements since human experts face a scenario with multiple decisions where there are several associated uncertainties and risks. With the inclusion of the new nine papers in this systematic review and mapping study, two selected studies used robust optimization (Paixão and Souza, 2013a,b). In our previous paper (Pitangueira et al., 2013) this approach was not used by any selected paper.

Another promising and challenging area is the addition of user judgment in the process of software requirements selection and prioritization. The involvement of the human judgment has drawn little attention in requirements selection using SBSE approaches. According to Harman (2010a,b), the search process in SBSE does not have to be fully automated and human contributions to fitness computation, incorporating human knowledge and judgment into the evolutionary search process may lead to better results.

6. Conclusions

Requirements engineering is a complex subject that covers multiple activities in the software development process. Some important tasks such as selecting and prioritization requirements can be an extremely arduous and exhaustive task throughout the process, depending on the size of the software features of the application domain and interdependence between requirements, among others factors.

In order to support the search for automated and less exhaustive solutions, Search-Based Software Engineering has emerged with the characteristic to address the problems of software engineering as a search based optimization problem.

Given the growth of this approach and obtaining meaningful results, the systematic review and mapping of the literature developed here studied and identified the state of the art applications of optimization in the process of software requirements selection and prioritization using SBSE approach.

The results obtained in mapping part can be very useful for a researcher in the software engineering field because it enables to identify the specific forums of publications, the papers desired, knowing the authors that have an important participation in the area. In the systematic review results was possible to identify the techniques and approaches which are already established in the area, such the distribution of formulation problem and the larger quantity of search technique used, as well as data type and scale used in the experiments.

This systematic review and mapping not only identified the techniques and methods most frequently used in the experiments, but also analyzed some quality criteria that must be taken into consideration when approaching the arduous task of selecting and prioritizing requirements. Some key points can be considered for further experiments:

- Insertion of statistical inference tests that can support the results obtained.
- Formal presentation of the hypothesis for a better understanding of what is the desired result.
- Adoption of real world instances and large data scale (for example, data from the software industry) to be closer to software industry reality.

From the qualitative results extracted from the analyzed papers, it can be seen that multi-goal modeling is a trend that is growing in current studies, however, there is much room to create models even closer to the reality of software engineers. Insertion of the user judgment and the inclusion of new restrictions, such as risks and uncertainties, constitute an open field for exploration. Consideration of the interdependence between requirements has already been studied; however, further elaboration is still required.

Future work includes studying the use of robust optimization to include new fitness functions and new restrictions in the modeling process, adoption of the user judgment in optimization process, new modeling take in account the type of software process and the usage of functional and non-functional requirements.

References

Aasem, M., Ramzan, M., Jaffar, A., Islamabad, E.S., 2010. Analysis and Optimization of Software Requirements Prioritization Techniques. Information and Emerging Technologies (ICIET).

Azmeh, Z., Mirbel, I., Crescenzo, P., 2013. Highlighting stakeholder communities to support requirements decision-making, Proceedings of the 19th International Conference on Requirements Engineering: Foundation for Software (REFSQ), pp. 190–205.

Bagnall, A.J., Rayward-Smith, V.J., Whittley, I.M., 2001. The next release problem. Inform. Softw. Technol. 43 (14), 883–890.

Baker, et al., 2006. Search based approaches to component selection and prioritization for the next release problem, In: Software Maintenance, 2006. ICSM '06, 22nd IEEE International Conference, pp. 176–185.

Barros, M.D.O., Dias-neto, A.C., 2011. Developing a systematic approach to evaluation of experimental studies on search-based software engineering, In: Brazilian Conference on Software: Theory and Practice (in Portuguese).

Ben-Tal, A., Nemirovski, A., 2002. Robust optimization – methodology and applications. Math. Program. 92 (3), 453–480.

Beyer, H., Sendhoff, B., 2007. Robust Optimization – A Comprehensive Survey. Computer Methods in Applied Mechanics and Engineering, pp. 3190–3218.

Boehm, B., Grunbacher, P., Briggs, R.O., 2001. Developing groupware for requirements negotiation: lessons learned. IEEE Softw. 18 (2).

Brasil, et al., 2012. A multiobjective optimization approach to the software release planning with undefined number of releases and interdependent requirements. In: Proceedings of the 13th Enterprise Information Systems Conference.

Cai, X., Wei, O., 2013. A hybrid of decomposition and domination based evolutionary algorithm for multi-objective software next release problem. In: Proceedings 0f the 10th International Conference on Control and Automation.

Cai, et al., 2012. Evolutionary approaches for multi-objective next release problem. Comput. Inform. 31, 847–875.

Carlshamre, P., Sandahla, K., Lindvallb, M., Regnellc, B., Natt, J., 2001. An industrial survey of requirements interdependencies in software product release planning. In: Fifth IEEE International Symposium on Requirements Engineering (RE'01), pp. 84–92.

Colares, F., et al., 2009. A new approach to the software release planning, XXIII Brazilian Symposium on Software Engineering, pp. 207–215.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol Comput. 6 (April (2)), 182–197.

Del Sagrado, J., Del Aguila, I.M., Orellana, F.J., 2010. Ant colony optimization for the next release problem: a comparative study. In: 2nd International Symposium on Search Based Software Engineering, pp. 67–76.

Del Sagrado, J., Águila, I.M., Orellana, F.J., 2011. Requirements interaction in the next release problem. In: GECCO 11, pp. 241–242.

Durillo, J.J., et al., 2009. A study of the multi-objective next release problem. In: 1st International Symposium on Search Based Software Engineering, pp. 49–58.

Durillo, J.J., et al., 2011. A study of the bi-objective next release problem. Empir. Softw. Eng. 16 (1), 29–60.

Dyba, T., Dingsoyr, T., Hanssen, G.K., 2007. Applying systematic reviews to diverse study types: an experience report. In: First International Symposium on Empirical Software Engineering and Measurement (ESEM-2007).

Felows, L., Hooks, I., 1998. A case for priority classifying requirements, In: 8th annual International Symposium on Systems Engineering. Seattle, Washington.

Firesmith, D., 2004. Prioritizing requirements. The Journal of Object Technology 3 (8), 35. doi:10.5381/jot.2004.3.8.c4.

Finkelstein, A., Harman, M., Mansouri, S.A., Ren, J., Zhang, Y., 2009. A search based approach to fairness analysis in requirement assignments to aid negotiation, mediation and decision making. Requir. Eng. 14 (4), 231–245.

Finkelstein, A., Harman, M., Mansouri, S., Ren, J., Zhang, Y., 2008. "Fairness Analysis" in requirements assignments, In: Proceedings of the 16th IEEE International Requirements Engineering Conference (RE '08). Barcelona, Catalunya, Spain, 8–12 September 2008. IEEE Computer Society, pp. 115–124.

Freitas, F., Coutinho, D.P., Souza, J.T., 2011. Software next release planning approach through exact optimization. Int. J. Comput. Appl. 22 (8), 1–8.

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. Adisson-Wesley, EUA.

Greer, D., Ruhe, G., 2004. Software release planning: an evolutionary and iterative approach. Inform. Softw. Technol. 46 (4), 243–253.

Grosan, C., Abraham, 2007. Hybrid Evolutionary Algorithms: Methodologies, Architectures, and Reviews. Studies in Computational Intelligence (SCI), vol. 75. Springer Verlag, pp. 1–17.

Harman, M., 2007a. The current state and future of search based software engineering, Techniques.
Harman, M., Jones, B.F., 2001. Search-based software engineering. Inform. Softw. Technol. 43 (14), 833–839.
Harman, M., 2007b. Search based software engineering for program comprehension, 15th IEEE International Conference on Program Comprehension (ICPC '07), pp. 3–13.
Harman, M., Krinke, J., Ren, J., Yoo, S., 2009a. Search based data sensitivity analysis applied to requirement engineering. In: Proceedings of the 11th Annual conference on Genetic and evolutionary computation, pp. 1681–1688.
Harman, M., Mansouri, S.A., Zhang, Y., 2009b. Search Based Software Engineering: A Comprehensive Analysis and Review of Trends Techniques and Applications. Technical Report TR-09-03.
Harman, M., 2010a. Why the virtual nature of software makes it ideal for search based optimization. In: 13th International Conference on Fundamental Approaches to Software Engineering (FASE 2010), pp. 1–12.
Harman, M., 2010b. The relationship between search based software engineering and predictive modeling. In: Proceedings of the 6th International Conference on Predictive Models in Software Engineering.
Harman, M., et al., 2012. Search based software engineering: techniques, taxonomy, tutorial. In: Mayer, B., Nordio, M. (Eds.), Empirical Software Engineering and Verification. Springer, pp. 1–59.
Heger, M., Dominique, A., 2004. A disquisition on the performance behavior of binary search tree data structures. Eur. J. Inform. Prof. 5, 67–75.
Jedlitschka, A., Pfahl, D., 2005. Reporting Guidelines for Controlled Experiments in Software Engineering Dietmar Pfahl. ISESE, pp. 95–104.
Jiang, H., Zhang, J., et al., 2010a. A hybrid ACO algorithm for the next release problem. In: Proceedings of 2nd International Conference on Software Engineering and Data Mining (SEDM 2010), pp. 166–171.
Jiang, H., Xuan, J., Ren, Z., 2010b. Approximate backbone based multilevel algorithm for next release problem. In: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation – GECCO '10, 1333.
Karlsson, J., Wohlin, C., Regnell, B., 1998. An evaluation of methods for prioritizing software requirements. Inform. Softw. Technol. 39 (14/15), 939–947.
Karlsson, J., Ryan, K., 1997. A cost-value approach for prioritizing requirements. IEEE Softw. 14 (5), 67–74.
Kheirkhah, E., Deraman, A., 2008. Important factors in selecting requirements engineering techniques. In: Information Technology Symposium (ITSim), vol. 4, pp. 1–5.
Kitchenham, B., 2004. Procedures for Performing Systematic Reviews. Technical Report TR/SE-0401. Keele University, Australia, pp. 28, ISSN 1353-776.
Kitchenham, B., Sjøberg, D.I.K., Brereton, O.P., Budgen, D., Dybå, T., Höst, M., Pfahl, D., et al., 2010. Can we evaluate the quality of software engineering experiments?. In: Proceedings of the ACM–IEEE International Symposium on Empirical Software Engineering and Measurement – ESEM '10.
Kumari, A.C., Srinivas, K., Science, C., 2012. Software requirements selection using quantum inspired elitist multi-objective evolutionary algorithm, IEEE ICAESM.
Li, C., Akker, J.M., Van Den, Brinkkemper, S., Diepen, G., 2007. Integrated requirement selection and scheduling for the release planning of a software product, International Conference on Requirements Engineering: Foundation for Software (REFSQ, 2007). LNCS, pp. 93–108.
Li, C., et al., 2010. An integrated approach for requirement selection and scheduling in software release planning. Requir. Eng. 15 (4), 375–396.
Li, et al., 2012. Multi-objective optimization approaches to software release time determination. Asia-Pac. J. Oper. Res. 9, 1–19.
Liebchen, C., Lübbecke, M., Möhring, R., Stiller, S., 2009. Robust and Online Large-scale Optimization. Lecture Notes in Computer Science, vol. 5868. Springer.
Lima, et al., 2011. A fuzzy approach to requirements prioritization. In: Proceedings of the 3rd SSBSE.
Mausa, G., et al., 2013. Hill climbing and simulated annealing in large scale next release problem, In: Proceedings of the EuroCo. July, pp. 452–459.
Ngo-The, A., Ruhe, G., 2008. A systematic approach for solving the wicked problem of software release planning. Soft Comput. 12 (1), 95–108.
Paixão, M., Souza, J., 2013a. A scenario-based robust model for the next release problem. In: Proceedings of the Fifteenth Annual Conference on Genetic and Evolutionary Computation Conference, pp. 1469–1476.
Paixão, M., Souza, J., 2013b. A recoverable robust approach for the next release problem. In: Proceedings of the 5th SSBSE.
Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M., 2008. Systematic mapping studies in software engineering. In: EASE '08: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering. University of Bari, Italy.
Pitangueira, A., Maciel, R., Barros, M., Aline, A., 2013. A systematic of software requirements selection and prioritization using SBSE approaches. In: Proceedings of the 5th International Symposium of Search-Based Software Engineering (SSBSE).
Pressman, R., 2002. Software Engineering, 5th ed. McGraw-Hill.
Ruhe, G., Greer, D., 2003. Quantitative Studies in Software Release Planning under Risk and Resource Constraints University of Calgary. Empirical Software Engineering, pp. 1–10.
Saaty, T.L., 1980. The Analytic Hierarchy Process. McGraw-Hill, Inc..
Saliu, M., Ruhe, G., 2007. Bi-objective release planning for evolving software systems. In: Proceedings of the 6th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT International Symposium on Foundations of Software Engineering (ESEC/FSE), pp. 105–114.
Saliu, O., Ruhe, G., 2005a. Supporting software release planning decisions for evolving systems. In: 29th Annual IEEE/NASA Software Engineering Workshop, pp. 14–26.
Saliu, O., Ruhe, G., 2005b. Software release planning for evolving systems. Innov. Syst. Softw. Eng. 1 (2), 189–204.
Sim, W.W., Brouse, P.S., 2014. Empowering requirements engineering activities with personas. Proc. Comput. Sci. 28, 237–246.
Sjoberg, D.I.K., Hannay, J.E., Hansen, O., Kampenes, V.B., 2005. A Survey of Controlled Experiments in Software Engineering, vol. 31. ISESE, pp. 733–753.
Sommerville, I., 2007. Software Engineering, 8th ed. Pearson-Addison Wesley.
Souza, J.T.De, et al., 2011. An ant colony optimization approach to the software release planning with dependent requirements. In: Proceedings of the Third Symposium on Search-Based Software Engineering (SSBSE).
Svahnberg, M., Gorschek, T., Feldt, R., Torkar, R., Saleem, S., Shafique, M.U., 2010. A systematic review on strategic release planning models. Inform. Softw. Technol. 52, 237–248.
SWEBOK, 2014. Guide to the Software Engineering Body of Knowledgment Available from: http://computer.centraldesktop.com/swebokv3review/ (accessed 28.11.13).
Talbi, El-Ghazali, 2009. Metaheuristics: From Design to Implementation. John Wiley & Sons.
Tonella, P., Susi, A., Palma, F., 2013. Interactive requirements prioritization using a genetic algorithm. Inform. Softw. Technol., 1–15.
Tonella, P., Susi, A., Palma, F., 2010. Using interactive GA for requirements prioritization. In: 2nd International Symposium on Search Based Software Engineering, (Section II), pp. 57–66.
Van Den Akker, J.M., et al., 2005a. Determination of the next release of a software product: an approach using integer linear. In: Proceedings of CAISE'05, 03018, pp. 119–124.
Van Den Akker, J.M., et al., 2005b. Flexible release planning using integer linear programming. In: Proceedings of the 11th International Workshop on Requirements Engineering for Software Quality (RefsQ '05).
Van den Akker, M., et al., 2008. Software product release planning through optimization and what-if analysis. Inform. Softw. Technol. 50 (1/2), 101–111.
Wiegers, K., 1999, September. First Thing First: Prioritizing Requirements. Software Development.
Wohlin, C., Runeson, P., Host, m., Ohlsson, M., Regnell, B., Wesslen, 2000. Experimentation in Software Engineering: An Introduction. Kluwer Academic Publishers.
Xuan, J., et al., 2012. Solving the large scale next release problem with a backbone-based multilevel algorithm. IEEE Trans. Softw. Eng. 38 (5), 1195–1212.
Zave, P., 1997. Classification of research efforts in requirements engineering. ACM Comput. Stud. 29 (4), 315–321.
Zhang, Y., et al., 2011. Comparing the performance of metaheuristics for the analysis of multi-stakeholder tradeoffs in requirements optimisation. Inform. Softw. Technol. 53 (7), 761–773.
Zhang, Y., Harman, M., 2010. Search based optimization of requirements interaction management. In: 2nd International Symposium on Search Based Software Engineering, pp. 47–56.
Zhang, Y., 2010. Multi-Objective Search-based Requirements Selection and Optimisation. King's College London, UK.
Zhang, Y., Harman, M., Lim, S.L., 2013. Empirical evaluation of search based requirements interaction management. Inform. Softw. Technol. 55, 126–152.
Zhang, Y., Harman, M., Mansouri, S.A., 2007. The multi-objective next release problem. In: Proceedings of the 9th Annual Conference on Genetic and evolutionary computation – GECCO '07, 1129.
Zhang, Y., Finkelstein, A., Harman, M., 2008. Search based requirements optimisation: existing work & challenges. In: Proceeding REFSQ'08 Proceedings of the 14th international conference on Requirements Engineering: Foundation for Software Quality, pp. 88–94.

Antônio Maurício da Silva Pitangueira completed his Masters in Business Administration from the Federal University of Rio Grande do Sul in 2004 (Concentration Area: Information and Decision Support Systems). He is currently Professor at the Federal Institute of Bahia-IFBA and participates in a research project involving simulation models. He has worked in Information Technology projects, with emphasis on methodologies of IT projects and systems modeling.

Rita Suzana Pitangueira Maciel has a Ph.D. in Computer Science from the Federal University of Pernambuco (2005) and Masters in Computer Science from the Federal University of Rio Grande do Sul (1995). She is currently an Associate Professor at the Federal University of Bahia. Her main researcher interests in Software Engineering area includes Model Driven Development, Requirements Engineering, Service Oriented Architecture and Collaborative Systems. Additionally, she has practiced application software engineering for ten years in industry.

Márcio Barros Márcio de Oliveira Barros is an associate professor at the Applied Informatics Department of the Federal University of Rio de Janeiro State (UNIRIO), Brazil. He holds a DSc degree in System Engineering and Computer Science from COPPE/UFRJ. His research concentrates on using optimization and simulation to model Software Engineering problems and find (close to) optimal solutions for them. He also uses optimization and simulation as tools to bring forth insight about Software Engineering problems. His main research interests include software design, software requirements, and project management.