Information and Software Technology 56 (2014) 568–585

Contents lists available at ScienceDirect

Information and Software Technology

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

A systematic literature review of software requirements prioritization research

Philip Achimugu *, Ali Selamat, Roliana Ibrahim, Mohd Naz'ri Mahrin

Faculty of Computing, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia

ARTICLE INFO

Article history:
Received 5 November 2013
Received in revised form 4 February 2014
Accepted 5 February 2014
Available online 11 February 2014

Keywords:
Stakeholders
Requirements
Prioritization
Software systems
Requirement engineering

ABSTRACT

Context: During requirements engineering, prioritization is performed to grade or rank requirements in their order of importance and subsequent implementation releases. It is a major step taken in making crucial decisions so as to increase the economic value of a system.
Objective: The purpose of this study is to identify and analyze existing prioritization techniques in the context of the formulated research questions.
Method: Search terms with relevant keywords were used to identify primary studies that relate requirements prioritization classified under journal articles, conference papers, workshops, symposiums, book chapters and IEEE bulletins.
Results: 73 Primary studies were selected from the search processes. Out of these studies; 13 were journal articles, 35 were conference papers and 8 were workshop papers. Furthermore, contributions from symposiums as well as IEEE bulletins were 2 each while the total number of book chapters amounted to 13.
Conclusion: Prioritization has been significantly discussed in the requirements engineering domain. However, it was generally discovered that, existing prioritization techniques suffer from a number of limitations which includes: lack of scalability, methods of dealing with rank updates during requirements evolution, coordination among stakeholders and requirements dependency issues. Also, the applicability of existing techniques in complex and real setting has not been reported yet.

Contents

* Corresponding author. Tel.: +60 143143251; fax: +60 7 5565044.
E-mail addresses: check4philo@gmail.com (P. Achimugu), aselamat@utm.my (A. Selamat), roliana@utm.my (R. Ibrahim), mdnazrim@utm.my (M.N. Mahrin).

http://dx.doi.org/10.1016/j.infsof.2014.02.001
0950-5849/

1. Introduction

Software engineering is more than just programming. It consists of all associated documentations, design principles or philosophies required to make these programs function as expected. Software requirements prioritization (SRP) is one of the design principles that enable software under consideration for development to function as expected. Requirements prioritization is considered to be a complex multi-criteria decision making process [1]. It has long been established that, for software systems to be acceptable by users or stakeholders, its requirements must be well captured, analyzed and prioritized [2–4]. SRP has to do with the identification of important requirements as perceived by relevant stakeholders [5]. Its essence is to implement the core requirements of stakeholders with respect to cost, quality, available resources and delivery time [6,7]. Therefore, the unambiguous ranking of software requirements is a critical success factor for ensuring efficient requirements engineering process.

There are so many advantages of prioritizing requirements before architecture design or coding. Prioritization aids the implementation of a software system with preferential requirements of stakeholders [8]. Also, the challenges associated with software development such as limited resources, inadequate budget, insufficient skilled programmers among others makes requirements prioritization really important. It can help in planning software releases since not all the elicited requirements can be implemented in single release due to some of these challenges [9]. Perini et al. [1] also opined that, prioritizing requirements plays an integral role in software development process as it enhances software release planning, budget control and scheduling. Therefore, determining which, among a pool of requirements to be implemented first and the order of implementation is known as requirements prioritization. Furthermore, software products that are developed based on prioritized requirements can be expected to have a lower probability of being rejected. To prioritize requirements, stakeholders will have to compare them in order to determine their relative importance through a weighting or scoring system [10]. These comparisons becomes complex with increase in the number of requirements [11]. Therefore, the aim of this study is to chronologically select and review published literature and present a holistic overview of existing techniques used in prioritizing software requirements.

Existing literature reviews can be classified as follows: traditional literature review (TLR) and systematic literature review (SLR). The TLR is mainly executed to establish the current research trends or paradigm shifts of a subject of discussion while the SLR attempts to address precise set of formulated research questions. To the best of the authors' knowledge, there is no existing SLR that focuses on prioritization techniques with respect to their explicit limitations, taxonomies and processes. Summarily, the essence of this SLR is to abridge and clarify the available evidences regarding (1) the existing prioritization techniques, (2) their limitations, (3) processes and (4) taxonomies. This SLR will therefore provide insight for both practitioners and researchers in the industries and academia in their quest to develop and utilize improved techniques.

The remainder of the article is structured as follows. Section 2 describes the research method used in this review. Section 3 discusses the threats to validity. Section 4 presents the results and discussions and Section 5 enumerates the research findings. Section 6 document the related works while Section 7 provides the general discussion of the entire review processes. Section 8 presents the limitations of this review while Section 9 concludes the study.

2. Research method

The approach proposed by [12] was adopted in executing this research (Fig. 1).

Referring to Fig. 1, the review protocols consist of six phases enumerated as follows: research questions, search strategy design, data extraction results, scrutiny, quality assessment criteria and data synthesis. In the first phase, a set of research questions were formulated based on the aim of this study. In the second phase, search strategies were designed in line with the formulated research questions which consisted of the identification of search terms and the choices of literature resources. The third phase dealt with the collation of extracted data while the fourth phase concentrated on the refinement of extracted data (studies) by scrutinizing the titles of collated studies to ensure relevance. In the fifth phase, the scrutinized studies were evaluated by applying the quality assessment criteria and the sixth phase dwelled on the selection of final studies for analysis and subsequent actions.

2.1. Research questions

The aim of this SLR is to understand and summarize the empirical proofs as regard the state-of-the-art prioritization techniques and identify areas for further research in order to complement the performances of existing techniques. To achieve this aim, 4 research questions (*RQs*) were formulated as presented below:

- *RQ1:* What are the existing techniques used for prioritizing requirements?
- *RQ2:* What are the descriptions and limitations of existing prioritization techniques?
- *RQ3:* What taxonomy of prioritization scales does each technique exhibit?
- *RQ4:* What are the processes involved in software requirements prioritization?

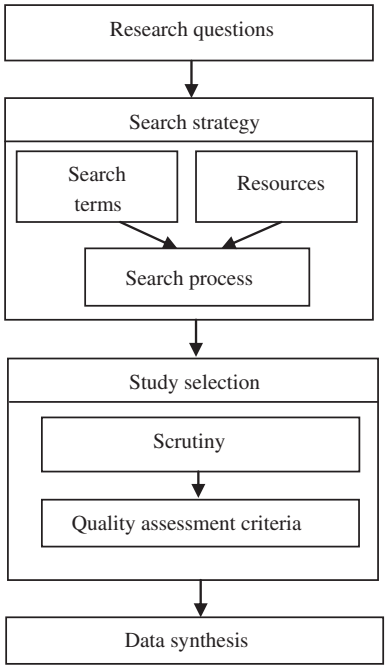

Fig. 1. Phases of the review protocol.

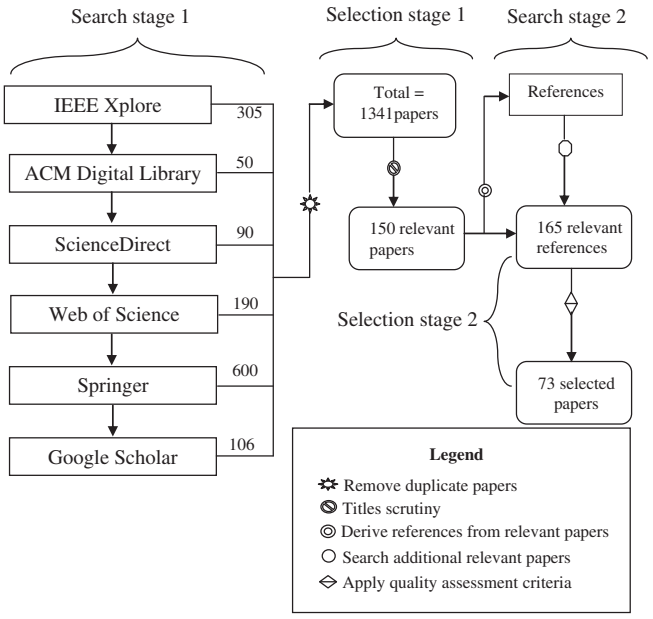

Fig. 2. Search and selection process.

These questions, which forms the basis for undertaking this research are intertwined and were simultaneously investigated.

2.2. Search strategy

The detailed description of the search strategies utilized in this research consisted of search terms, literature resources and search process as explained below:

2.2.1. Search strings

The following steps were used to build the search terms [12]:

(a) Derivation of major terms from the research questions.
(b) Identification of alternative spellings and synonyms for major terms.
(c) Identification of keywords in relevant papers or books.
(d) Usage of the Boolean OR to incorporate alternative spellings and synonyms.
(e) Usage of the Boolean AND to link the major terms.

The resulting search terms are described as follows: Requirements AND (prioritization/OR technique/) AND (“categories” OR “taxonomies” OR “classifications” OR techniques OR “activities” OR “”processes /” OR “limitations/” OR “shortcomings/” OR “practice/” OR “methods” OR “practices” OR “significance/”).

2.2.2. Literature resources

Six electronic database resources were used to primarily extract data for synchronizations in this research. These include: IEEE Xplore, ACM Digital Library, ScienceDirect, Web of Science, Springer, and Google Scholar. Title, abstract and index terms were used to conduct search for published journals papers, conference proceedings, workshops, symposiums, books chapters and IEEE bulletins.

2.2.3. Search process

Systematic literature review has to do with a comprehensive search of all relevant sources about a subject of discussion. However, the search processes employed in this research consisted of the following enumerated steps and depicted in Fig. 2.

Search stage 1: A thorough search was launched on the six electronic database sources and the returned results (papers) were assembled as sets of prospective papers.
Search stage 2: The reference lists of all relevant papers were perused to detect additional relevant papers and then, if any, combine them with the ones in stage 1.

2.3. Study selection

From the first search stage, 1341 prospective studies were realized. Next, the titles of these studies were used to scrutinize and collate relevant studies. This task was necessary to eliminate duplicate and irrelevant studies. Consequently, 150 relevant studies were selected. Thereafter, the references of each selected study were perused to identify important studies that might have been missed out during the initial search process. This effort led to the identification of 15 additional studies which took the tally of the selected studies to 165. Finally, the quality assessment question or criteria were applied to these 165 studies. At the end of the exercise, 73 studies were selected and deemed capable of providing answers to the formulated research questions.

2.3.1. Scrutiny

From Fig. 2, 1341 prospective studies were obtained during the first search process. Therefore, scrutiny was necessary to streamline these studies to relevant ones. First, the title of each study was considered; then their contents were briefly studied. Hence, all the papers that do not reflect the topic of discussion or are incapable of addressing any of the formulated research questions were excluded from the relevant studies list. Also, only studies written and published in English language from peer-reviewed journals, refereed conference proceedings, workshops, symposiums, book chapters and IEEE bulletins were considered for inclusion in the list of relevant studies. However, when multiple copies of the same paper appeared, the most complete, recent and improved one is included in the search process while the others are excluded. Specifically, we performed a systematic literature review for

Table 1
Inclusion and exclusion criteria.

Inclusion criteria	Exclusion criteria
a. All papers published in English language	a. Papers that are not published in English language
b. Papers that focuses on requirements prioritization	b. Papers that do not have any link with the research questions
c. Relevant papers that are published from 1996 to 2013	c. Gray papers; i.e. papers without bibliographic information such as publication date/type, volume and issue numbers were excluded
d. All published papers that have the potential of answering at least, one research question	d. Duplicate papers (only the most complete, recent and improved one is included). The rest are excluded

Table 2
Quality assessment questions.

S/No.	Questions
1	Are the aims of the research clearly articulated?
2	Is the proposed technique clearly described?
3	Is the experimental design appropriate?
4[a]	Is the experiment applied on adequate project data sets or case study?
5	Does the research add value to the academia or industrial community?

[a] "Yes" (2 or more datasets or case studies); "Partly" (1 dataset or case study); "No" (None).

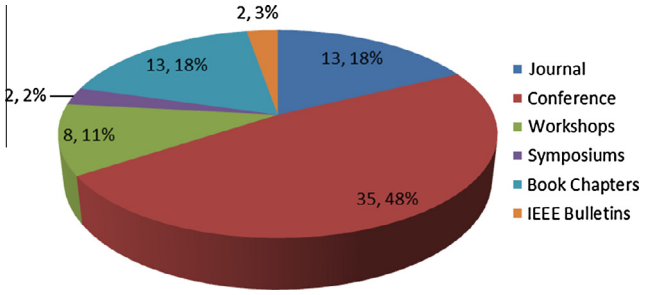

Fig. 3. Numbers and percentages of collated studies.

requirements prioritization techniques on articles published from 1 January 1996 to 31 December 2013. This is because; the earliest published paper that addressed at least one of the research questions after scrutiny was in 1996. A summary of the criteria used for scrutiny are shown in Table 1.

2.3.2. Quality assessment of selected studies

The quality assessment (QA) of selected studies was achieved by weighting or scoring technique so as to obtain relevant studies capable of addressing each research question. We formulated a number of quality assessment questions to evaluate the credibility, completeness and relevance of the selected studies. These questions are presented in Table 2. Each question has only three optional answers: "Yes", "partly" or "No". These three answers are scored as follows: "Yes" = 1, "Partly" = 0.5 and "No" = 0. Consequently, the quality score for a particular study is computed by finding the sum of all the scores of the answers to the QA questions. The authors meticulously executed quality assessment of the selected studies. All discrepancies on the quality assessment results were discussed among the authors with the aim of reaching consensus. The reliability of the findings of this review was accomplished by considering only the relevant studies with acceptable quality rate, i.e., with quality score greater than 2.5 (50% of the percentage score). As a result, 93 papers were excluded from the initially collated studies giving rise to 73 finally selected relevant studies (see Fig. 2). The quality scores of these selected studies are depicted in Table A1 of the Appendix section.

These metrics guided the interpretation of findings of the selected studies and determined the validity of the inferences proffered. It also helped in ascertaining the credibility and coherent synthesis of results.

2.4. Data synthesis

The essence of data synthesis is to summarize proofs from the selected studies in order to address or answer the research questions. To synthesize data, the 73 selected studies were further perused to assess the detailed contents of each study with respect to the criteria defined in Table 3.

The aim of this exercise is to synchronize selected studies in order to enhance clarity. This will also aid the identification of precise answers to the research questions. The extracted data obtained in this research consisted of both quantitative data (e.g., values of prioritization accuracy or results) and qualitative data (e.g., strengths and weaknesses of existing prioritization techniques). The comprehensive explanations of how data syntheses were executed are enumerated below:

Data related to *RQ1* were organized in a coherent manner. Visualization tools such as bar chart, pie chart, and colon charts were also used to present the distribution of various prioritization techniques. In *RQ2*, the limitations of existing techniques were identified from selected studies and the outcome was displayed in a tabular form. In *RQ3*, the taxonomies of the various prioritization techniques were identified and visualized using a descriptive diagram while the processes involved in prioritizing requirements was the focus in *RQ4*. The results were also displayed in a tabular form.

Table 3
Contents assessment criteria.

Selected study	Description
Identification of study bibliographic references	Unique identification number for the study, publication year, title and source
Type of study	Journal and conference papers, IEEE bulletins and book chapters
Study focus	Domain topic, problems, scope, motivation and objectives
Research methodology	Case study, survey, experiment, interview, observation and questionnaire
Data analysis	Quantitative/qualitative analysis
Application domain	Description of the context and application domain of the study. For example, academic or industrial settings
Constriants	Identification of the study's shortcomings and areas for future research
System development life cycle phase	Identification of the life cycle phases in which the study falls under

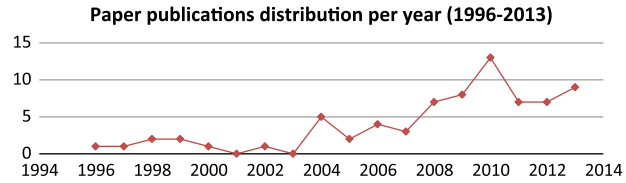

Fig. 4. Number of papers by year of publication.

3. Threats to validity

The publications biasness and inaccurate extraction of data were considered to be the major threats militating against this review protocol. The studies were chosen based on the search strategy described previously which include (a) various literature databases, (b) selection criteria, and (c) quality criteria. The index terms corresponding or relating to the specified research questions was used to detect relevant studies that were utilized in this review. However, there exists the possibility of missing important studies because, not all studies can be extracted using the terms that is related to the research questions in their titles, abstracts or keywords. To curb this threat, a manual scrutiny of the references of all the extracted studies was patiently carried out to identify those studies that were missed out during the initial search. Additionally, a precise definition of the selection criteria that complied with the research questions was enforced to avoid incorrect exclusion of desired studies. The studies were selected via a meticulous application of the quality assessment criteria and where discrepancies exist; inconsistencies were immediately resolved. This way, a wide range of further studies were detected. However, the second category of envisaged threat is known as publication biasness. This is a situation where positive results on prioritization techniques are more likely to be reported than negative results, or scholars claiming that, their technique outpaced others. Consequently, this can lead to an overestimation of the performances of existing techniques. To avert this threat, publications that dealt with comparisons of various existing techniques were searched for and included in the selected studies in order to obtain an objec-

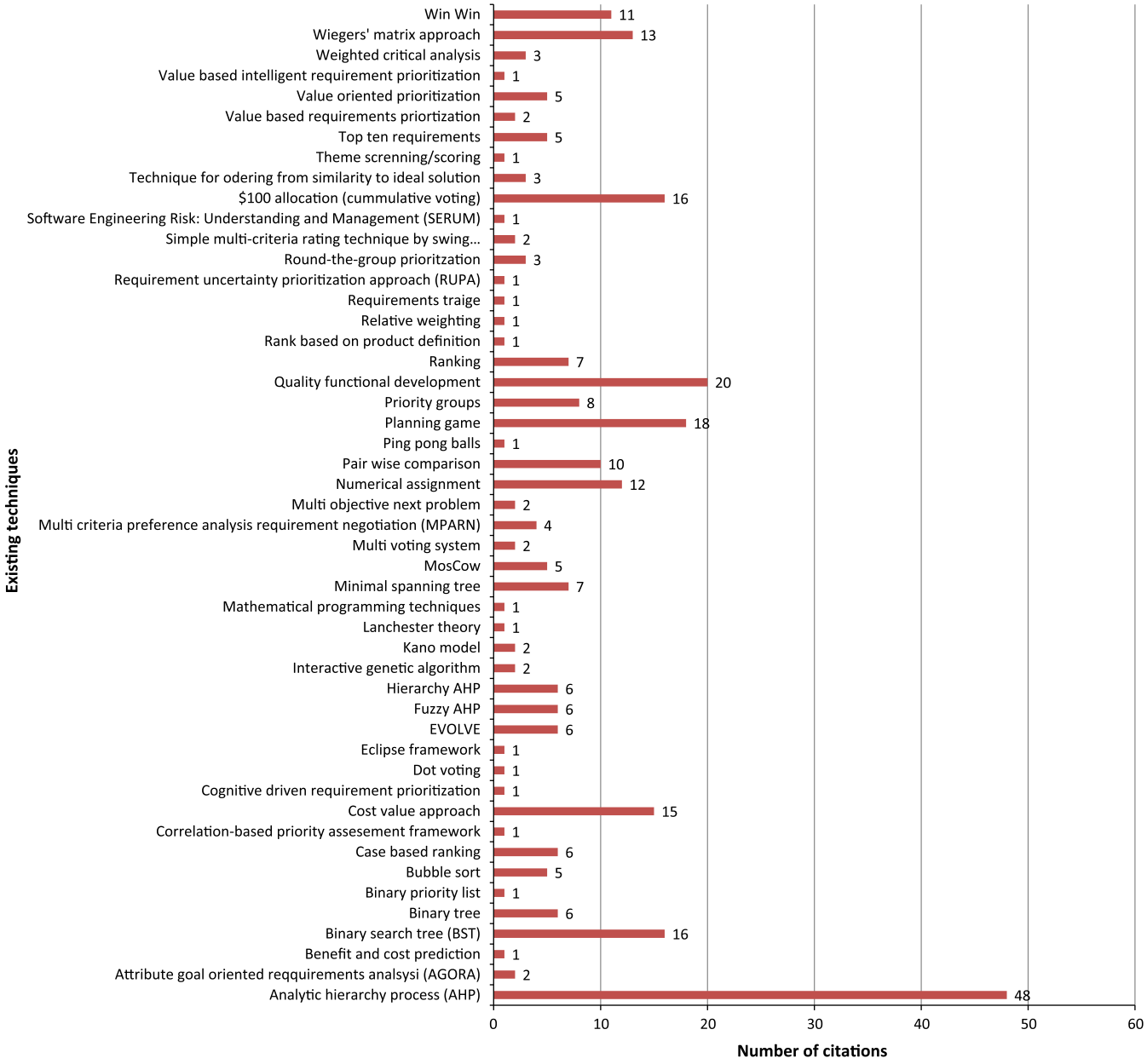

Fig. 5. Existing prioritization techniques.

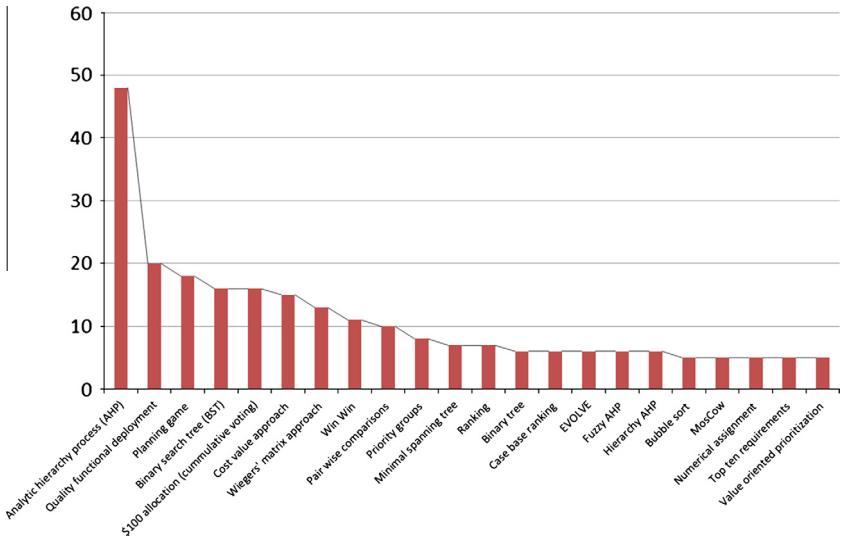

Fig. 6. Most cited and utilized techniques.

tive evaluation result across the various techniques. This is because, in most cases; these comparative studies present unbiased reports. Finally, to minimize the threat associated with inaccurate extraction of data, all the selected studies were re-evaluated to identify the true positives, a situation where the title of a study could connote relevance but the contents do not contain answers to any of the research questions.

To reduce the inaccuracy of extracted data, the authors carried out independent valuation using the assessment questions in Table 2 on the selected studies and later engaged in an inter-rater agreement to resolve the discrepancies and obtain similarities in the ordering of ratings executed by the authors.

4. Results and discussion

This section presents and discusses the findings of this review. We start by presenting an overview of the selected studies. Secondly, we present a detailed description of the findings of this review in line with the research questions in separate sub-sections. The review results are also interpreted in this section.

4.1. Overview of selected studies

73 studies were selected for this research. Among them, 13 papers were published in journals, 35 papers appeared in conference proceedings, 8 papers came from workshops, 2 papers were extracted from symposiums, 13 papers were from book chapters and 2 papers came from IEEE bulletin. The respective numbers and percentages of the selected studies are represented in Fig. 3; while the number of papers by year of publication is depicted in Fig. 4. However, the detailed descriptions of the selected studies are shown in the Table A2 of the Appendix section.

4.2. Requirements prioritization techniques (RQ1)

49 prioritization techniques were identified from the selected studies. Fig. 5 shows the plot of the various techniques and their citation rates; while Fig. 6 depicts the most cited and utilized techniques. So many efforts have been invested in requirements prioritization research [84,85]. This is evidenced with the number of techniques available in the literature. These techniques are used to determine requirements with greater value to business successes. The most cited prioritization techniques are detailed as follows: Numerical assignment is based on grouping requirements into different categories; that is, high, medium and low [86]. The pairwise comparison technique [59] was developed based on analytical hierarchy process. This technique determines the relative importance of requirements by pair-wisely comparing each requirement to obtain weighted scores across all the requirements. Cost-value prioritization technique was proposed by [83] which also have its root in analytical hierarchy process. This technique determines top requirements by graph plots to visualize the requirements value against its implementation cost. $100-Test (Cumulative voting) was proposed by [1] to determine prime requirements by disbursing fabricated $100 notes across requirements according to their degree of importance; after which, requirements are sorted in ascending order with respect to the number of dollar notes each requirement has earned. Techniques like bubble sort, binary search tree, minimal spanning tree, priority groups, MosCow, win-win, planning game, quality functional deployment, top ten and binary priority listing have also gained adequate attention [25]. However, using these techniques pose some significant challenges in terms of evaluating the relative priority differences among requirements [17]. In Karlsson's work [59]; they posited that, their technique (pairwise comparison) performs better in terms of accuracy of prioritization results than numeral assignment since the latter just deals with the assignment of numbers over specified requirements and the former adjudge requirements by pair-wisely comparing each to determine their relative importance. From Fig. 6, it is clear that, the analytical hierarchy process (AHP) is the most cited and cherished technique among others. Furthermore, Karlsson [25] affirmed that, AHP is one of the few techniques that provide reliable prioritization results because it possesses the ability to compute consistency ratios across requirements to enhance clarity. Wiegers [87] postulated that, the cost-value framework is also rooted in AHP. The EVOLVE technique [22] attempted to apply machine learning approach in numerous iterations in order to maximize the weighted benefits over all relevant stakeholders. The essence of iteration is to support evolution of requirements so as to curb the rank reversal problems. The value-oriented prioritization technique was proposed by [82]. In this technique, requirements are prioritized based on their business value to the organization or stakeholders. Again, this technique have its root in AHP since the business value of each requirement is determined by weights which eventually leads to the construction of prioritization matrix to reflect these valued

requirements. More recent machine learning technique like case base ranking, hierarchy AHP and fuzzy AHP also utilizes weighting scales or linguistic weights to prioritize requirements based on predefined criteria using learning algorithms.

The learning algorithm accepts weights or scores of rated requirements to compute the final ranks of requirements. From the literature reviewed, techniques like analytical hierarchy process (AHP), binary search tree (BST), game planning (GP), cumulative voting (CV), value cost approach (CVA) and quality functional deployment (QFD) are the most used and eminent techniques. The detailed citation indexes of each prioritization technique are shown in Table A3 of the Appendix section. The remaining techniques are considered to be of low significances with just one or two citations.

4.3. Descriptions and Limitations of existing prioritization techniques (RQ2)

The descriptions and limitations of existing prioritization techniques are enumerated in Table A4 of the Appendix section. These limitations serve as the basis for any improvement hoped for. The descriptions of these techniques are necessary to proffer a glimpse of how each prioritization technique function. While frantic efforts have been invested in developing and improving prioritization techniques, some limitations still exist which requires urgent research attention. These limitations were gained from the selected studies.

From the literature; most authors concluded that, AHP suffer scalability problems. This is due to the fact that, AHP executes ranking by considering the criteria that are defined through an assessment of the relative priorities between pairs of requirements. This becomes impracticable as the number of requirements increases. It also does not support requirements evolution or rank updates but provide efficient or reliable results [25,33]. Also, from this research, it is observed that, most existing machine learning techniques suffer from rank updates issue. Prominent machine learning techniques that suffer from this limitation are case base ranking [13]; interactive genetic algorithm prioritization technique [14]; Binary search tree [25] and EVOLVE [22]. Furthermore, existing techniques also lack approach for classifying prioritized requirements according to their respective ranks. Karlsson et al. [25] conducted some researches where certain prioritization techniques were empirically evaluated. From their research, they reported that, most of the prioritization techniques apart from AHP and bubble sorts produce unreliable or misleading results while AHP and bubble sorts were also time consuming. The authors then posited that; techniques like hierarchy AHP, spanning tree, binary search tree, priority groups produce unreliable results and are difficult to implement. Babar et al. [33] were also of the opinion that, techniques like requirement triage, value intelligent prioritization and fuzzy logic based techniques are also error prone due to their reliance on experts and are time consuming too. Planning game has a better variance of numerical computation but suffer from rank updates problem. Wieger's method and requirement triage are relatively acceptable and adoptable by practitioners but these techniques do not support rank updates in the event of requirements evolution as well. It is worthy to note that, some of the existing techniques are manual. Example of such techniques include, round the group prioritization, cumulative voting, win-win, ping pong balls approach, multi voting system, dot voting system and top ten prioritization techniques. These techniques work very well for small scale projects with only 10–30 requirements across 5–6 stakeholders [65].

In summary, the limitations of existing prioritization techniques can be described as follows:

(4.3.1) *Scalability:* Techniques like AHP, pairwise comparisons and bubblesort suffer from scalability problems because, requirements are compared based on possible pairs causing $n(n-1)/2$ comparisons [25]. For example, when the number of requirements is doubled in a list, other techniques will only require double the effort or time for prioritization while AHP, pairwise comparisons and bubblesort techniques will require four times the effort or time. This is bad scalability.

(4.3.2) *Computational complexity:* Most of the existing prioritization techniques are actually time consuming in the real world [25,33]. The author of the work documented in [8] executed a comprehensive experimental evaluation of five different prioritization techniques namely; AHP, binary search tree, planning game, $100 (cumulative voting) and a new method which combines planning game and AHP (PgcAHP), to determine their ease of use, accuracy and scalability. The author went as far as determining the average time taken to prioritize 13 requirements across 14 stakeholders with these techniques. At the end of the experiment; it was observed that, planning game was the fastest while AHP was the slowest. Planning game prioritized 13 requirements in about 2.5 min while AHP prioritized the same number of requirements in about 10.5 min. In other words, planning game technique took only 11.5 s to compute the priority scores of one requirement across 14 stakeholders while AHP consumed 48.5 s to accomplish the same task due to pair comparisons.

(4.3.3) *Rank updates:* Perini et al. [13] defined rank update as 'anytime' prioritization; that is, the ability of a technique to automatically update ranks anytime a requirement is included or excluded from the list. This situation has to do with requirements evolution. Therefore, existing prioritization techniques are incapable of updating or reflecting rank status whenever a requirement is introduced or deleted from the rank list. Therefore, it does not support iterative updates. This is very critical because, decision making and selection processes cannot survive without iterations. Therefore, a good and reliable prioritization technique will be one that supports rank updates. This limitation seems to cut across most existing techniques.

(4.3.4) *Communication among stakeholders:* Most prioritization techniques do not support communication among stakeholders. One of the most recent works in requirement prioritization research reported communication among stakeholders as part of the limitations of their technique [13]. This can lead to the generation of vague results. Communication has to do with the ability of all relevant stakeholders to fully understand the meaning and essence of each requirement before prioritization commences.

(4.3.5) *Requirements dependencies:* This is a crucial attribute that determines the reliability of prioritized requirements. These are requirements that depend on another to function. Requirements that are mutually dependent can eventually be merged as same; since without one, the other cannot be implemented. Prioritizing such requirements may lead to erroneous or redundant results. This attribute is rarely discussed among prioritization research authors. However, the author of the work documented in [57] asserted that, dependencies can be detected from mapping the pre and post conditions from the whole set of requirements, based on the contents of each requirement. Therefore, a good prioritization technique should cater or take requirements dependences into cognizance before initiating the process.

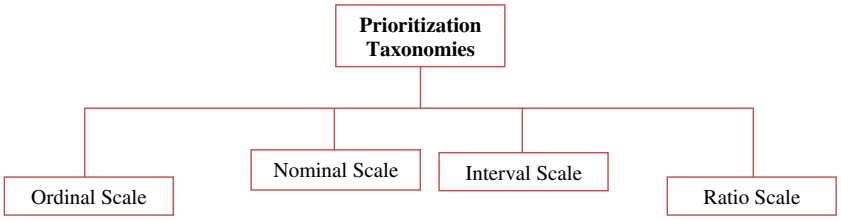

Fig. 7. Prioritization taxonomies.

(4.3.6) *Error proneness:* Existing prioritization techniques are also prone to errors [18]. This could be due to the fact that, the rules governing the requirements prioritization processes in the existing techniques are not robust enough. This has also led to the generation of unreliable prioritization results because; such results do not reflect the true ranking of requirements from stakeholder's point of view or assessment after the ranking process. Therefore robust algorithms are required to generate reliable prioritization results.

(4.3.7) *Lack of fully implemented requirements prioritization systems:* From this research, it was observed that most existing prioritization techniques have not been really implemented for real-life scenarios probably because of the complexities associated with prioritizations and the time required for generating prioritized requirements. In the future, there is need to implement algorithms that will improve or support requirements prioritization at commercial or industrial level [96–98]. Before these algorithms can work efficiently, the methods for capturing requirements in an unambiguous way must be well thought of [99] since the output of prioritization processes depend on the input and the aim is to plan for software releases [100] as well as the successful development of software products in line with negotiated or prioritized requirements [101].

In summary, for requirements engineers to efficiently execute prioritization, there is need to urgently address these shortcomings; which can aid developers produce good quality software products that meet user's requirements.

4.4. Taxonomies of prioritization techniques (RQ3)

Four major taxonomies of prioritization techniques have been identified as shown in Fig. 7. Voola and Babu [16]; Aasem et al. [36]; Berander and Andrews [80] Karlsson et al. [102] identified nominal, ordinal, interval and ratio scales or taxonomies of prioritization techniques. Techniques exhibiting any of these taxonomies order requirements into groups or sub-groups and numbers or inscriptions are assigned across all the requirements to reflect their relative importance. Here, no requirement is expected to belong to more than one group or subgroup at a time. Consequently, the frequencies across all requirements are computed and the requirement with the highest frequency is taken to be the prime requirement.

The ordinal scale can be used to complement the performance of the nominal scale since the former present information about the ordering of requirements [59]; while the latter has the capacity of showing the ranks of various requirements but cannot further indicate the extent at which, one requirement is considered to be more important than the other. The interval scale possesses information about the size of the intervals between the ordered set of requirements so as to enhance the computation of the discrepancy that exist among requirements. An interval scale preserves order, just like the ordinal scale. The ratio scale is considered to be the most viable of all the scales or taxonomies because, it has the capacity of ordering, determining intervals or relative distances

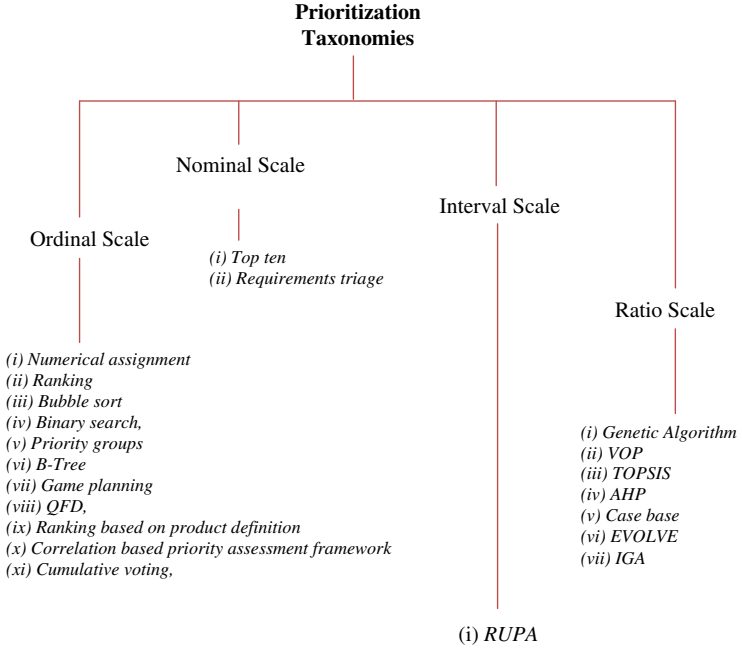

Fig. 8. Prioritization taxonomies with their respective techniques.

Table 4
Processes involved in requirements prioritization.

S/No	Source	Processes
1.	Perini et al. [13]	(a) Requirements gathering (b) Pair sampling (c) Priority elicitations (d) Priority learning (e) Ranked requirements
2.	Vola and Babu [16]	(a) Identify relevant stakeholders for the proposed project (b) Obtain weights of each requirement from the stakeholders (c) Aggregate weights by applying IER algorithm (d) Compute ranks with utility theory (e) Distribute output to the stakeholders
3.	Perini et al. [19]; Avesani et al. [54]	(a) Specification of requirements (b) Pair sampling (c) Elicitation of preferences (d) Rank learning (e) Prioritized requirements
4.	Firesmith [23]	(a) Convince stakeholders (b) Train stakeholders (c) Categorize raw potential requirements (d) Prioritize the actual requirements (e) Publish priorities (f) Estimate efforts (g) Schedule development (h) Maintain priorities
5.	Dabbagh and Lee [28]	(a) Requirements elicitation (b) Collection of preference values for each pair of requirements (c) Application of AHP's pairwise comparison method on the candidate list of requirements (d) Insertion of the results into an array called initial set (e) Selection of requirements that poses the greatest value from the initial set (f) Removal of the selected requirements from the initial set (g) Removal of requirements with negative impact in the priority list (h) Prioritized requirements
6.	Iqbal et al. [44]	(a) Requirements elicitation (b) Modified AHP engine (c) Consistency check (d) Requirements prioritization
7.	Daneva and Herrmann [50]	(a) Primary and secondary requirements specification (b) Determination of relative importance based on benefit/cost (c) Visualization of prioritized requirements
8	Bebensee et al. [73]	(a) Pile all requirements that have been collected from various sources (b) Take one element from the pile, and use it as the root requirement (c) Take another requirement and compare it in terms of priority to the root requirement (d) If the requirement has a lower priority than the root requirements, compare it to the requirement below the root and so forth. If it has a higher priority than the root, compare it to the requirement above the root and so forth. This is done until the requirement can finally be placed as sub-requirement of a requirement without an appropriate sub-requirement (e) Steps b to d are repeated for all requirements (f) Finally, traverse the list from top to down to obtain the prioritized order of the requirements
9.	Duan et al. [74]	(a) Requirements specification (b) Grouping of various requirements into clusters (c) Prioritization of clusters by human analyst (d) Computation of weights for each clustered requirements (e) Generation of prioritized requirements
10.	Saaty [84]	(a) Specification of requirements (b) Construction of requirements matrix (c) Pair sampling (d) Elicitation of preferences (e) Computation of ranks (f) Ranks display
11.	Wiegers [87]	(a) Elicitation of requirements (b) Estimation of relative benefit of each requirement (c) Estimation of the relative consequences of not including a particular requirement (d) The overall estimation of the relative benefit and consequences (e) Estimation of relative cost of implementing each requirement (f) Estimation of the relative degree of technical or other risk associated with each requirement (g) Computation of the relative priorities of each requirement in a spreadsheet using the formula priority = value%/(cost% $*$ cost weight + risk% $*$ risk weight) (h) The requirements are sorted in descending order according to the priority scores
12	Kaiya et al. [90]	(a) Establishing initial goals as customers' needs (b) Decomposing and refining goals into sub-goals (c) Choosing and adopting the goals from the alternatives of decomposed goals (d) Detecting and resolving conflicts on goals
13.	Barney et al. [100]	(a) Requirement elicitation (b) Stakeholder's requirements ranking (c) Calculation of ranking values

Table 4 (*continued*)

S/No	Source	Processes
		(d) Prioritized requirements
14.	Sivzittian and Nuseibeh [103]	(a) Selection of one or more prioritization criteria among business goals and technical features (b) Acquisition of requirements ordering according to a specific criterion from one or more stakeholders (c) Composition of the acquired orderings into a final one based on an appropriate composition schema

and ratios between requirements. Example of requirements prioritization techniques that falls under each scale is shown in Fig. 8.

The nominal scale-based prioritization techniques have the capacity of calculating the mode and chi square of ranked requirements, while the ordinal scale-based ones can calculate the median and percentile. The interval scale-based techniques computes the mean, standard deviation, correlation, regression, analysis of variance and the ratio scale-based techniques perform all forms of statistical computations including geometric mean, harmonic mean, coefficient of variations and also utilizes algorithms in the prioritization process [36].

4.5. Processes involved in requirements prioritization (RQ4)

The major processes involved in prioritizing requirements are described in Table 4.

5. Research findings

Prioritizing requirements is an important activity in software development [91–95]. When customer expectations are high, delivery time is short and resources are limited, the proposed software must be able to provide the desired functionalities as early as possible. Many projects are challenged with the fact that, not all the requirements can be implemented because of limited time and resource constraints. This means that, it has to be decided which of the requirements can be removed for the next release. According to [87], information about priorities is needed, not just to ignore the least important requirements but also to help the project manager resolve conflicts, plan for staged deliveries, and make the necessary trade-offs. Therefore, the impact of requirements prioritization in requirements engineering domain cannot be over-emphasized. From the literature reviewed, the following findings were discovered:

(5.1) *Requirements prioritization determines the relative necessity of elicited requirements:* Whereas all requirements are mandatory, some are more important than others. For example, failure to implement certain requirements may have grievous business consequences that can make the entire system valueless, while others although contractually binding, would have far less grievous business consequences if they were not implemented or not implemented correctly.

(5.2) *Negotiation of precise requirements:* Requirements prioritization help software engineers through negotiation and consensus eliminates unnecessary potential "requirements" (i.e., goals, desires, and concerns that do not merit the mandatory nature of true requirements).

(5.3) *Implementation schedule:* Requirements prioritization aid project managers and customers in modifying project schedules in order to deal with the project realities with available resources and expected delivery date. It also improves customer satisfaction by increasing the likelihood of implementing customers' preferential requirements. Therefore, with requirement prioritization, software projects are less likely to be rejected after development because, the software would have been developed based on unambiguous or precise software requirements.

(5.4) *Judicious utilization of fund:* Requirements prioritization provides stakeholders with a rough estimate of the financial implication of implementing each requirement in order to determine where best to expend project funds with excellent output.

6. Related work

There are few systematic reviews in this research domain. Kaur and Bawa [15] presented a systematic literature review in software prioritization research. In their work, 7 prioritization techniques were identified and analyzed. These techniques include; analytic hierarchy process, value oriented prioritization, cumulative voting, numerical assignment, binary search tree, planning game and B tree prioritization. In [60], the authors reported a systematic mapping study in software requirements prioritization with specific emphasis on empirical studies. Most of the studies documented in their work were about the validation of various researches or solution proposals. Also, the authors reported the prevalence of studies on techniques or methodologies and confirmed that, there is a scarce interest in the strict evaluation of tools that could be beneficial to industry. Furthermore, the authors in [69] carried out another systematic literature review on software requirements selection and prioritization research with particular focus on search-based software engineering (SBSE). Their search strategy yielded 30 articles which were evaluated with 18 established quality criteria. The review did not only identify the techniques and methods most frequently used in the experiments, but also analyzed some quality criteria that must be taken into consideration when prioritizing requirements. Similarly, a systematic review of requirements prioritization techniques is documented in [104]. The review concentrated on comparisons of prioritization techniques, where 8 relevant studies were selected. The author asserted that, requirements prioritization domain is yet to witness substantial amount of research and most studies in this area dealt with prioritizing small number of requirements.

7. Discussion

This review underscores the philosophy behind requirement prioritization research and presents a synopsis of existing prioritization techniques as it relates requirements engineering phase of the system development life cycle. Prioritization is a vital activity that should hold if accurate decisions regarding product releases are to be taken [105,106]. Decision-making cuts across almost every facets of human endeavor or discipline. In computer science and software engineering precisely, complex decision making experiences are inevitable. Notable disciplines that are confronted with complex decision making processes include psychology, organizational behavior, education, medicine and engineering to mention a few. As a result, many decision making models have been

proposed in the past to help decision makers arrive at profitable conclusions.

Software system's acceptability level is frequently determined by how well the developed system has met or satisfied the specified user's or stakeholder's requirements. Hence, eliciting and prioritizing the appropriate requirements and scheduling right releases with the correct functionalities are a critical success factor for building formidable software systems. In other words, when vague or imprecise requirements are implemented, the resulting system will fall short of user's or stakeholder's expectations. Many software development projects have enormous prospective requirements that may be practically impossible to deliver within the expected time frame and budget [13,14]. It therefore becomes highly necessary to source for appropriate measures for planning and rating requirements in an efficient way in order to avoid breach of trust, contract or agreement. This process is known as requirement prioritization. It is also the process of identifying the most valuable requirements in their order of importance or preference.

The significances of requirements prioritization are many. A lot of techniques have been proposed in the literature by authors and scholars, yet many areas of improvement have also been identified to optimize the prioritization processes. With the advent of Internet and quest for software that can service distributed organizations, the number of stakeholders in large scale projects have drastically increased with requirements possessing the attributes of being changed due to innovation, technological advancement or business growth. However, whatever prioritization technique is been selected for modifications, the essence of requirements prioritization must not be lost. Automated requirements prioritization techniques should incorporate all the factors engendering efficient prioritization process.

8. Conclusion

The aim of this research was to identify and examine the status quo of software requirements prioritization techniques. The method utilized in this research was a systematic literature review. With this method, some pertinent research questions were formulated based on the aim of the study to identify and ascertain existing prioritization techniques, their limitations, taxonomies and processes. This was carried out through identifying, assessing and interpreting relevant studies. The essence of this research was to identify areas for possible improvement or enhancement via systematic evaluation of relevant and current studies in requirements prioritization techniques as reported in the literature. In this research, a review protocol was developed and some quality evaluation questions or criteria were developed and applied to the studies to ensure relevance and correctness. The review protocol describes the aim of this research and how it was achieved. The research identification, study selection, study quality assessment, data extraction and data synchronization processes were executed in line with the review protocol. During these processes, relevant primary studies were identified and the quality of these studies was assessed. Data were extracted from these primary studies and the extracted data were synchronized. The research objectives were considered to have been met and the formulated research questions were considered to have been addressed. Some studies that dealt with the method of enhancing existing prioritization techniques were identified and synthesized. It was discovered that, although a lot of prioritization techniques exist; improvements are still required. Some of these improvement borders on scalability, computational complexities, ease of use, reliability of results, vali-

Table A1
Results of quality scores of selected studies.

Paper ID	QA1	QA2	QA3	QA4	QA5	Score
S13	1	1	1	0.5	1	4.5
S14	1	1	1	0.5	1	4.5
S15	1	1	0.5	0	1	3.5
S16	1	1	1	0.5	1	4.5
S17	1	1	1	0.5	1	4.5
S18	1	1	1	0	1	4
S19	1	1	1	0.5	1	4.5
S20	1	1	1	0.5	1	4.5
S21	1	1	1	0.5	1	4.5
S22	1	1	1	0.5	1	4.5
S23	1	1	1	0	1	4
S24	1	1	1	0.5	1	4.5
S25	1	1	1	0.5	1	4.5
S26	1	1	1	0	1	4
S27	1	1	1	0	1	4
S28	1	1	1	0	1	4
S29	1	1	1	0.5	1	4.5
S30	1	1	0.5	0	1	3.5
S31	1	0.5	1	0	1	3.5
S32	1	0.5	1	0	1	3.5
S33	1	1	0.5	0	1	3.5
S34	1	1	1	0.5	1	4.5
S35	1	1	1	0	1	4
S36	1	1	1	0	1	4
S37	1	1	1	0	1	4
S38	1	1	1	0	1	4
S39	1	1	1	0	1	4
S40	1	1	0.5	0.5	1	4
S41	1	1	1	0.5	1	4.5
S42	1	1	0.5	0	1	3.5
S43	1	1	1	0	1	4
S44	1	1	1	0	1	4
S45	1	1	1	0	1	4
S46	1	1	1	0	1	4
S47	1	1	0.5	0.5	1	4
S48	1	1	0.5	0.5	1	4
S49	1	1	0.5	0.5	1	4
S50	1	1	0	0	1	3
S51	1	1	1	0	1	4
S52	1	1	0.5	0.5	1	4
S53	1	1	1	0.5	1	4.5
S54	1	1	0.5	0	1	3.5
S55	1	1	1	0.5	1	4.5
S56	1	1	1	0.5	1	4.5
S57	1	1	0.5	0.5	1	4
S58	1	1	0.5	0.5	1	4
S59	1	1	0.5	0.5	1	4
S60	1	1	0.5	0.5	1	4
S61	1	1	0	0	1	3
S62	1	1	0.5	0	1	3.5
S63	1	1	0.5	0	1	3.5
S64	1	1	0.5	0.5	1	4
S65	1	1	0.5	0.5	1	4
S66	1	1	0.5	0.5	1	4
S67	1	1	1	0.5	1	4.5
S68	1	1	0	0	1	3
S69	1	1	0.5	0	1	3.5
S70	1	1	1	0.5	1	4.5
S71	1	1	0.5	0.5	1	4
S72	1	1	1	0.5	1	4.5
S74	1	1	1	0.5	1	4.5
S75	1	1	1	0.5	1	4.5
S76	1	1	1	0.5	1	4.5
S77	1	1	0	0	1	3
S78	1	1	1	0.5	1	4.5
S79	1	1	0.5	0.5	1	4
S80	1	1	1	0.5	1	4.5
S81	1	1	0.5	0.5	1	4
S82	1	1	0.5	0.5	1	4
S83	1	1	0.5	0	1	3.5
S84	1	1	0.5	0.5	1	4
S85	1	1	1	0.5	1	4.5

Table A2
Overview of selected studies with their respective reference numbers.

Publication type	Publication year	Publication name	Refs.
Journal	2013	IEEE Transactions on Software Engineering	[13]
Journal	2013	Information and Software Technology	[14]
Journal	2013	International Journal of Engineering Research & Technology	[15]
Journal	2012	Software Engineering: An International Journal	[16]
Journal	2012	Software Quality Journal	[17]
Journal	2011	International Journal of Innovative Computing, Information and Control	[18]
Journal	2009	Information and Software Technology	[19]
Journal	2006	Journal of Systems and Software	[20]
Journal	2006	International Journal of Production Research	[21]
Journal	2004	Information and Software Technology	[22]
Journal	2004	Journal of Object Technology	[23]
Journal	1999	International Journal of Quality & Reliability Management	[24]
Journal	1998	Information and Software Technology	[25]
Conference	2013	Conference on Systems Engineering Research	[26]
Conference	2013	International Conference on Software Engineering	[27]
Conference	2013	International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing IEEE	[28]
Conference	2013	European Conference on Software Maintenance and Reengineering IEEE	[29]
Conference	2012	International Conference on Information Systems Design and Intelligent Applications	[30]
Conference	2012	Open Systems (ICOS) Conference, IEEE	[31]
Conference	2012	Computing Technology and Information Management (ICCM), International Conference	[32]
Conference	2011	Computer Networks and Information Technology (ICCNIT), International Conference IEEE	[33]
Conference	2011	Computer Science and Automation Engineering (CSAE), IEEE International Conference	[34]
Conference	2011	International Requirements Engineering Conference IEEE	[35]
Conference	2010	Information and Emerging Technologies (ICIET), International Conference IEEE	[36]
Conference	2010	Research Challenges in Information Science (RCIS), International Conference IEEE	[37]
Conference	2010	International Conference on Mathematical/Analytical Modeling and Computer Simulation IEEE	[38]
Conference	2010	International Conference on Cognitive Informatics IEEE	[39]
Conference	2010	International Conference on Data Storage and Data Engineering IEEE	[40]
Conference	2010	International Conference on Software Engineering Advances	[41]
Conference	2010	Computational Intelligence and Software Engineering (CiSE) International Conference IEEE	[42]
Conference	2010	EUROMICRO Conference on Software Engineering and Advanced Applications IEEE	[43]
Conference	2010	International Conference on Data Storage and Data Engineering IEEE	[44]
Conference	2010	International Conference on Data Storage and Data Engineering IEEE	[45]
Conference	2009	International Advance Computing Conference IEEE	[46]
Conference	2009	International Conference on Information Management and Engineering IEEE	[47]
Conference	2008	Australian Conference on Software Engineering IEEE	[48]
Conference	2008	International Conference on Automated Software Engineering IEEE/ACM	[49]
Conference	2008	Euromicro Conference on Software Engineering and Advanced Applications IEEE	[50]
Conference	2008	International Conference on Emerging Trends in Engineering and Technology IEEE	[51]
Conference	2008	Agile Conference IEEE	[52]
Conference	2007	IEEE International Requirements Engineering Conference	[53]
Conference	2006	Conference on Software Engineering Research and Practice ACM	[9]
Conference	2005	IEEE International Conference on Requirements Engineering	[54]
Conference	2004	SEKE Conference	[55]
Conference	2004	International Computer Software and Applications Conference IEEE	[56]
Conference	2002	Systems Engineering, Test & Evaluation Conference	[57]
Conference	1999	Asian Pacific Software Engineering Conference IEEE	[58]
Conference	1996	Requirements Engineering International Conference IEEE	[59]
Workshop	2013	Empirical Requirements Engineering (EmpiRE), 2013 International Workshop IEEE	[60]
Workshop	2012	Recommendation Systems for Software Engineering (RSSE), 2012 Third International Workshop	[61]
Workshop	2010	Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises IEEE	[62]
Workshop	2009	International Workshop on Requirements Engineering and Law	[63]
Workshop	2009	International Workshop on Software Product Management	[64]
Workshop	2008	International Workshop on Software Product Management IEEE	[65]
Workshop	2007	International Workshop on Comparative Evaluation in Requirements Engineering IEEE	[4]
Workshop	2000	Database and Expert Systems Applications, International Workshop IEEE	[66]
Symposiums	2012	International Symposium on Empirical software engineering and measurement ACM	[67]
Symposiums	2010	International Symposium on Search Based Software Engineering IEEE	[68]
Book chapter	2013	Search Based Software Engineering	[69]
Book chapter	2011	Requirements engineering: Foundation for software quality	[70]
Book chapter	2011	Search Based Software Engineering	[71]
Book chapter	2011	Computer Networks and Intelligent Computing	[72]
Book chapter	2010	Requirements Engineering: Foundation for Software Quality	[73]
Book chapter	2009	Requirements Engineering	[74]
Book chapter	2009	Software & Systems Design	[75]
Book chapter	2009	Empirical Software Engineering	[76]
Book chapter	2008	Software Engineering Research and Practice	[77]
Book chapter	2006	Empirical Software Engineering	[78]
Book chapter	2006	Software Process Improvement and Practice	[79]
Book chapter	2005	Engineering and Managing Software Requirements	[80]
Book chapter	2004	Software Process Improvement	[81]
IEEE bulletin	2007	Software, IEEE	[82]
IEEE bulletin	1997	IEEE Software	[83]

Table A3
Existing techniques and their citation indexes.

S/No.	Techniques	Source
1.	Analytic hierarchy process (AHP)	[4,9,13,15,17–21,25,27,28,30,31,33,36,38,40–45,47,48,51,53,54–62,64,68,70,74–77,80–82,84,85]
2.	Attribute goal-oriented requirement analysis (AGORA)	[75,90]
3.	Benefit and cost prediction	[50]
4.	Binary search tree (BST)	[4,9,15,19,25,28,33,34,37,51,53,60,64,68,74,75]
5.	Binary Tree	[15,33,36,46,51,70]
6.	Binary Priority List	[73]
7.	Bubblesort	[4,25,28,33,68]
8.	Case Base Ranking	[13,19,20,31,54,68]
9.	Correlation-based priority assessment framework (CBPA)	[20]
10.	Cost value approach	[4,9,13,17,28,35,36,41,42,44,49,52,57,70,85]
11.	Cognitive driven requirement prioritization	[39]
12.	Dot voting	[37]
13.	Eclipse process framework	[37]
14.	EVOLVE	[14,17,22,52,68,81]
15.	Fuzzy AHP	[21,31,44,52,72,73]
16.	Hierarchy AHP	[25,28,31,33,51,4]
17.	Interactive requirement prioritization	[14,68]
18.	Kano model	[37,65]
19.	Lanchester theory	[65]
20.	Mathematical programming techniques	[37]
21.	Minimal spanning tree	[19,25,28,37,51,9,4]
22.	MosCow	[28,37,48,62,73]
23.	Multi voting system	[23,37]
24.	Multi criteria preference analysis requirement negotiation (MPARN)	[41,54,55,76]
25.	Multi-objective next release problem	[14,68]
26.	Numerical assignment	[9,15,18,28,33,35,36,42,47,59,63,81]
27.	Pair wise analysis	[9,17,28,37,47,59,77,79,80,82]
28.	Ping pong balls	[37]
29.	Planning game	[4,9,13,18,28,33,36,37,42,53,60,61,64,74,75,77,79,81]
30.	Priority groups	[4,9,19,25,37,51,80,82]
31.	Quality functional deployment	[13,20,21,23,24,27,37,38,40,41,47,54–57,64,65,71,73,76]
32.	Ranking	[18,33,44,36,61,63,81]
33.	Rank based on product definition	[37]
34.	Relative weighting	[37]
35.	Requirement triage	[33,53]
36.	Requirement uncertainty prioritization approach (RUPA)	[16]
37.	Round the group prioritization	[23,37,64]
38.	Simple multi criteria rating techniques by swing (SMART)	[54,55]
39.	Software Engineering Risk: Understanding and Management (SERUM)	[34]
40.	$100 Allocation (Cumulative voting)	[15,17,18,33–36,43,48,9,61,63,64,68,74,81]
41.	Technique for ordering from similarity to ideal solution (TOPSIS)	[26,27,73]
42.	Theme screening/scoring	[37]
43.	Top ten requirements	[18,36,63,68,81]
44.	Value based requirement prioritization	[26]
45.	Value oriented prioritization	[15,17,53,75,84]
46.	Value based intelligent requirements prioritization	[18]
47.	Weighted criteria analysis	[23,37,64]
48.	Weiger's method	[4,9,13,17,33,37,57,64,74,75,80–82]
49.	WinWin	[13,18,23,36,53,55,56,66,75,76,81]

dation of techniques in industrial settings and requirements evolvability and dependency issues. Therefore, further studies on prioritization techniques can dwell on the aforementioned limitations. Some threats to validity were eminent in this study; most of which have been identified and curbed at the early stage of this research. Six major online database sources were used to extract data that was utilized in addressing the research questions.

9. Limitations

9.1. Completeness

We have within the confines of the formulated research questions completed a rigorous review exercise on requirement prioritization research. Consequently, 73 studies were finally identified and selected to possess the capacity of adequately addressing at least one of the formulated research questions. From the selected studies, the earliest publication was in 1996 and the most recent ones were published in 2013. However, with the paradigm shift and dynamic nature of requirements prioritization research, we cannot fully guarantee to have captured all the available studies in this research area. Another issue of concern borders on the fact that, important or relevant studies must have been missed out in the non-English published articles since only English published articles were considered in this review.

9.2. Publication biasness

Publication biasness refers to a situation where positive research results are more likely to be reported than negatives ones [12]. This is regarded as a threat. However, to curb this threat, review and evaluatory studies were sorted for, because of the high tendencies of reporting the correct strengths and limitations of existing techniques. However, in this review, gray literature (tech-

Table A4
Descriptions and limitations of existing prioritization techniques.

S/No.	Name of techniques	Description	Limitations/source
1.	Analytic hierarchy process	This technique uses a pair-wise comparison matrix to calculate the relative importance of each requirements	Time consuming at higher values of number of requirements. Not scalable [25,74,83]
2.	Attribute goal-oriented requirement analysis	Prioritization achieved by attributing preference values to requirements computed in a decision matrix form and represented in a goal graph	Inefficient management of the complexities of the goal graphs [90]
3.	Benefit and cost prediction	This technique prioritize requirements based on the value of each requirement to the organization or customer as well as the cost perceived to be incurred by implementing such requirements	Do not cater for evolution of requirements; do not also weight requirements based on a scale [50]
4.	Binary Search Tree	This technique analyzes all the elicited requirements and ranks them in a hierarchical order (parent-child relationship)	Provides only a simple ranking of requirements without assigning any priority values [74]
5.	Binary-tree	This technique consist of an algorithm that is capable of searching for requirements in nodes and compare them to determine relative importance using a weighting scale	Complex and do not scale well, not implemented and tested in a collaborating environment [36,46]
6.	Binary priority list	This technique sort or rank requirements based on their perceived benefits to the application domain	Do not cater for requirement dependencies, the ranking was based on just one criterion (benefit), other criteria such as costs, penalty, and risk were missing [73]
7.	Bubble sort	This technique prioritize requirements based on the following steps: (a) Preparation and arrangement of requirements in a vector. (b) Execution of the comparisons of requirements (c) Sorting the requirements in their order of rating from bottom to top	Not scalable for large number of requirements [33]
8.	Case based ranking	This technique uses a machine learning approach to reduce the amount of information required from stakeholders, for achieving rankings of a given quality degree	Un-scalable; inability to support coordination among different stakeholders through negotiation [13]
9.	Correlation-based priority assessment framework (CBPA)	Prioritize requirements by incorporating inter-perspective relationships (correlations) of requirements using relationship matrix	This technique did not consider how to prioritize requirements with negative correlations [20]
10.	Cost-value ranking	Prioritize requirements based on their perceived value and implementation cost	Time consuming and un-scalable [83]
11.	Cognitive driven requirements prioritization	This technique combines the use of AHP, self-organizing maps (SOM) and Cognitive Psychology methods to build a Conceptual Framework that is capable of assessing the knowledge level of stakeholders with respect to the proposed software before prioritization commences	This technique lacks methods of aggregating and globalizing preference weights during the ranking process [39]
12.	Dot voting	This technique implores sticky dots to rank requirements. The higher the numbers of sticky dots, the most valued a requirement will be	NI
13.	Eclipse process framework	This framework rank requirements with a weighting scale	NI
14.	EVOLVE	This technique applies genetic algorithm in a number of iterations to maximize the weighted benefit over all the different stakeholders	Computational complexity [17] and needed to be tested in a more complex industrial setting for effectiveness [22].
15.	Fuzzy AHP	This technique uses triangular fuzzy numbers and weights for ranking requirements	Not scalable, do not cater for requirements interdependencies [71]
16.	Hierarchy AHP	In this technique, requirements are prioritized in hierarchical order based on the accumulated scores of each requirement across relevant stakeholders	Produces a lot of judgemental errors due to its inability to address consistency like in the case of AHP [25]
17.	Interactive requirements prioritization	This technique utilizes a weighting scale to prioritize requirements and minimize their disagreement divergences	Did not conduct more experiments on other case studies to corroborate their findings [14]
18.	Kano model	Kano technique prioritize requirements based on the comparison of customer satisfaction with technical Excellence	NI
19.	Lanchester theory	This technique prioritizes requirements based on business objective and market share. It is very similar to quality functional deployment technique	Do not define relative values for the linguistic terms that can aid the calculation of relative weights across all the requirements [65]
20.	Mathematical programming techniques	Use machine learning algorithms to rank requirements	NI
21.	Minimal spanning tree	This technique utilize a weighting scale to rank requirements in a hierarchical form	More sensitive to judgemental errors due to inconsistencies in rank result [25]
22.	MosCow	It is an acronym that stand for; "**MUST have**": Requirements are not negotiable; the failure to deliver these requirements would result in the failure of the entire project. "**SHOULD have**": Features that would be nice to have if at all possible "**COULD have**" Features that would be nice to have if at all possible but slightly less advantageous than the "**S**" "**WON'T have**" (also known as "wish list"). These requirements are not unimportant, but they will definitely not be implemented in the current software project. They may, at a later stage, be created.	NI
23.	Multi voting system	This technique allows stakeholders to vote for requirements in line with their perceived importance of each requirement	NI
24.	Multi-criteria Preference Analysis Requirements Negotiation (MPARN)	MPARN model guides stakeholders to make negotiated agreements using multi-criteria preference analysis techniques, cooperating with the WinWin artifacts of win conditions, issues, and options	It does not detect inconsistencies amongst ranking values [54,92]
25.	Multi-objective next release problem	This technique utilizes the main objectives or rationale behind the development of software systems to prioritize requirements	This technique does not produce a total ordering of requirements. Rather, they group requirements for the planning of the next

(*continued on next page*)

Table A4 (*continued*)

S/No.	Name of techniques	Description	Limitations/source
			release [68]
26.	Numerical assignment	Numerical assignment is based on grouping requirements into different categories, viz. high, medium, and low	A criticism of the technique is that use of categories like high, medium, and low may confuse the stakeholders since different stakeholders [25,80,87,93]
27.	Pair wise analysis	Requirements are ranked by comparing them in pairs until the top requirements emerge at the top of the stack	Tedious, complicated and provide unreliable results [83]
28.	Ping Pong Balls	A particular numbers of ping balls which represent each requirement are given to stakeholders to cast their lot in order of requirements priority	NI
29.	Planning game	This technique has to do with a situation where clients categorize requirements into three classes: i.e.; Essential, conditional and optional. The process is based on two criteria: business value judged by the clients and technical risk judged by the developers	Do not scale well with large number of requirements [74]
30.	Priority groups	This technique has to do with the formation of priority groups where requirements are classified based on their importance by stakeholders	Does not possess the ability of indicating consistency in the decision maker's judgment [25]
31.	Quality Functional Deployment (QFD)	This technique utilizes matrices to chronologically represent client's expectations and how these expectations are to be met by the developers	Typically applied to small subsystems [89]; does not cater for inconsistencies and not scalable [54]
32.	Ranking	This technique implores numbers to rank requirements in ascending order starting from 1 to *n*	Do not scale with many requirements [18]; do not show relative difference between ranked items, suitable for single stakeholder because this technique finds it difficult to align several different stakeholders' views [80]
33.	Rank based on product definition	This prioritization technique accounts for three important perspectives on product definition: the business, users, and technology	NI
34.	Relative weighting	This technique rank requirements using a weighting scale to reflect the relative importance of requirements	NI
35.	Requirements triage	This technique determines requirements priorities by educating and convincing stakeholders to understand each requirement before ranking commences. The requirements are then sorted	Prone to errors and the results do not recall [33,107]
36.	Requirement uncertainty prioritization approach (RUPA)	This technique prioritize requirements by aggregating weights using internal evidential reasoning (IER) algorithm	Not scalable, applied to small scale project [16]
37.	Round-the-Group Prioritization	Elicited requirements are displayed on cards and positioned in a haphazard manner where stakeholders are requested to reposition the cards according to their order of preferences	Cannot cater for large number of requirements [48,83]
38.	Simple multi-criteria rating technique by swing (SMARTS)	This technique weighs requirements based on some defined criteria to prioritize them	It does not detect inconsistencies amongst ranking values, do not also scale [54]
39.	Software engineering risk understanding and management (SERUM)	Uses estimations for cost, benefit, development risk, and operational risk reduction to execute prioritization process	The limitation of this approach is that it does not handle dependencies between requirements [40]
40.	$100 Allocation (Cumulative Voting)	Stakeholders are given a fabricated $100 note to expense on the elicited requirements. After which, the total expended money for each requirement is divided by the total numbers of stakeholders to prioritize requirements	Not suitable for large number of requirements [80,48,88]
41.	Technique for ordering from similarity to ideal solution (TOPSIS)	Requirements are valued based on a weighting scale	Inability of hierarchically organizing requirements; Poor prerequisite handling and inability to update ranks whenever requirements evolves [26,27]
42.	Theme screening/scoring	This technique prioritize requirements scoring them using a pre-defined criteria for the scoring process	NI
43.	Top ten requirements	In this technique, stakeholders are asked to choose their top-ten requirements from the pool of requirements	This technique can be conflicting or ambiguous since weights are not utilized in the ranking process [94]
44.	Value based requirements prioritization	This technique extracts individual utility functions from each stakeholder which provides a natural method of addressing requirements interdependences	Inability to hierarchically organize goals or requirements, poor report generation and poor pre-requisite handling [26]
45.	Value oriented prioritization	Requirements are linked to identified business values and prioritized based on the stakeholder ratings	Ignores requirement dependencies in the computation procedure [17]. Not suitable for larger project [74]
46.	Value based intelligent requirement prioritization	This technique involves the use of stakeholder's, experts and automated fuzzy logic based system to iteratively prioritize the requirements	Do not classify the ranked requirements in various categories such as "most important", "medially important" "less important" etc. [18]
47.	Weighted critical analysis	Criteria are established and weights are allotted to each requirement to prioritize them	Ignores requirement dependencies
48.	Wiegers' matrix approach	This technique describes a semi-quantitative analytical approach that uses a simple spreadsheet model to help estimate the relative priorities for a set of product feature	This technique can be easily manipulated by stakeholders seeking to accomplish their own objectives [74]
49.	WinWin	Stakeholders express their goals as win conditions and if everyone concurs, the win conditions become agreements. If not, iteration occurs	Difficulty in reaching consensus especially when biased stakeholders are involved; inconsistencies in prioritized requirements [54,66]

NI = not indicated.

nical reports, work in progress, unpublished or non-peer reviewed publications) which may also possess the capacity of answering any of the research questions were not included in the selected studies. This is one of the limitations of this review as some techniques or limitations or taxonomies or processes must have been missed as a result.

9.3. Data synthesis

Different domains in requirements engineering and prioritization were sorted for in order to identify titles capable of answering our research questions. The content assessment criteria described in Table 3 was used to conduct data synthesis. However, we cannot again guarantee whether these criteria were sufficient enough to undertake this task.

Acknowledgements

The authors appreciate the efforts of the Ministry of Science, Technology and Innovation Malaysia (MOSTI) under Vot 4S062 and Universiti Teknologi Malaysia (UTM) for supporting this research.

Appendix A

See Tables A1–A4.

References

[1] D. Leffingwell, D. Widrig, Managing Software Requirements: A Unified Approach, Addison-Wesley Longman Inc., 2000.
[2] A. Davis, The art of requirements triage, IEEE Comput. 36 (3) (2003) 42–49.
[3] N.R. Mead, Requirements Prioritization Introduction, Software Engineering Institute Web Publication, Carnegie Mellon University, Pittsburgh, 2006.
[4] A. Perini, F. Ricca, A. Susi, C. Bazzanella, An empirical study to compare the accuracy of AHP and CBRanking techniques for requirements prioritization, in: Proceedings of the Fifth International Workshop on Comparative Evaluation in Requirements Engineering, IEEE, 2007, pp. 23–35.
[5] G. Ruhe, A. Eberlein, D. Pfahl, Trade-off analysis for requirements selection, Int. J. Softw. Eng. Knowl. Eng. 13 (4) (2003) 345–366.
[6] A. Finkelstein, M. Harman, S. Mansouri, J. Ren, Y. Zhang, A search based approach to fairness analysis in requirement assignments to aid negotiation, mediation and decision making, Requir. Eng. 14 (2009) 231–245.
[7] S. Barney, A. Aurum, C. Wohlin, A product management challenge: creating software product value through requirements selection, J. Syst. Architect. 54 (2008) 576–593.
[8] V. Ahl, An experimental comparison of five prioritization methods – investigating ease of use, accuracy and scalability, Master's Thesis, School of Engineering, Blekinge Institute of Technology, Sweden, August 2005.
[9] P. Berander, K.A. Khan, L. Lehtola, Towards a research framework on requirements prioritization, in: Proceedings of Sixth Conference on Software Engineering Research and Practice in Sweden (SERPS'06), October 2006.
[10] M. Kobayashi, M. Maekawa, Need-based requirements change management, in: Proc. ECBS 2001 Eighth Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems, 2001, pp. 171–178.
[11] N.W. Kassel, B.A. Malloy, An approach to automate requirements elicitation and specification, in: Proc. of the 7th IASTED International Conference on Software Engineering and Applications, Marina Del Rey, CA, USA, 3–5 November 2003.
[12] B.A. Kitchenham, S. Charters, Guidelines for Performing Systematic Literature Reviews in Software Engineering, Tech. Rep. EBSE-2007-01, Keele University and University of Durham, 2007.
[13] A. Perini, A. Susi, P. Avesani, A machine learning approach to software requirements prioritization, IEEE Trans. Softw. Eng. 39 (4) (2013) 445–460.
[14] P. Tonella, A. Susi, F. Palma, Interactive requirements prioritization using a genetic algorithm, Inf. Softw. Technol. Inf. Softw. Technol. 2012 (55) (2013) 173–187.
[15] G. Kaur, S. Bawa, A survey of requirement prioritization methods, Int. J. Eng. Res. Technol. 2 (5) (2013) 958–962.
[16] P. Voola, A. Babu, Requirements uncertainty prioritization approach: a novel approach for requirements prioritization, Softw. Eng.: Int. J. (SEIJ) 2 (2) (2012) 37–49.
[17] R. Thakurta, A framework for prioritization of quality requirements for inclusion in a software project, Softw. Qual. J. 21 (2012) 573–597.
[18] M. Ramzan, A. Jaffar, A. Shahid, Value based intelligent requirement prioritization (VIRP): expert driven fuzzy logic based prioritization technique", Int. J. Innovat. Comput. 7 (3) (2011) 1017–1038.
[19] A. Perini, F. Ricca, A. Susi, Tool-supported requirements prioritization. Comparing the AHP and CBRank method, Inf. Softw. Technol. 51 (2009) 1021–1032.
[20] X. Liu, Y. Sun, C. Veera, Y. Kyoya, K. Noguchi, Priority assessment of software process requirements from multiple perspectives, J. Syst. Softw. 79 (11) (2006) 1649–1660.
[21] H. Raharjo, M. Xie, A. Brombacher, Prioritizing quality characteristics in dynamic quality function deployment, Int. J. Prod. Res. 44 (23) (2006) 5005–5018.
[22] D. Greer, G. Ruhe, Software release planning: an evolutionary and iterative approach, Inf. Softw. Technol. 46 (4) (2004) 243–253.
[23] D. Firesmith, Prioritizing requirements, J. Object Technol. (2004). 3(S).
[24] F. Franceschini, A. Rupil, Rating scales and prioritization in QFD, Int. J. Qual. Reliab. Manage. 16 (1) (1999) 85–97.
[25] J. Karlsson, C. Wohlin, B. Regnell, An evaluation of methods for prioritizing software requirements, Inf. Softw. Technol. 39 (14) (1998) 939–947.
[26] N. Kukreja, S. Payyavula, B. Boehm, S. Padmanabhuni, Value-based requirements prioritization: usage experiences, Proc. Comput. Sci. 16 (2013) 806–813.
[27] N. Kukreja, Decision theoretic requirements prioritization: a two-step approach for sliding towards value realization, in: Proceedings of the 2013 International Conference on Software Engineering, IEEE Press, 2013, pp. 1465–1467.
[28] M. Dabbagh, S. Lee, A consistent approach for prioritizing system quality attributes, in: Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2013 14th ACIS International Conference on, IEEE, pp. 317–322.
[29] M. Asghar, A. Marchetto, A. Susi, G. Scanniello, Maintainability-based requirements prioritization by using artifacts traceability and code metrics, in: Software Maintenance and Reengineering (CSMR), 2013 17th European Conference on, IEEE, pp. 417–420.
[30] P. Voola, A. Babu, Interval evidential reasoning algorithm for requirements prioritization, in: Proceedings of the International Conference on Information Systems Design and Intelligent Applications 2012 (INDIA 2012) Held in Visakhapatnam, India, January 2012. Springer, Berlin Heidelberg, pp. 915–922.
[31] A. Ejnioui, C. Otero, A. Qureshi, Software requirement prioritization using fuzzy multi-attribute decision making, in: Open Systems (ICOS), 2012 IEEE Conference on, IEEE, pp. 1–6.
[32] S. Forouzani, R. Ahmad, S. Forouzani, N. Gazerani, Design of a teaching framework for software requirement prioritization, in: Computing Technology and Information Management (ICCM), 2012 8th International Conference on, IEEE, vol. 2, 2012, pp. 787–793.
[33] M. Babar, M. Ramzan, S. Ghayyur, Challenges and future trends in software requirements prioritization, in: Computer Networks and Information Technology (ICCNIT), 2011 International Conference on, IEEE, pp. 319–324.
[34] A. Ahmad, A. Shahzad, V. Padmanabhuni, A. Mansoor, S. Joseph, Z. Arshad, Requirements prioritization with respect to geographically distributed stakeholders, in: Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on, IEEE, vol. 4, 2011, pp. 290–294.
[35] R. Svensson, T. Gorschek, B. Regnell, R. Torkar, A. Shahrokni, R. Feldt, A. Aurum, Prioritization of quality requirements: state of practice in eleven companies, in: Requirements Engineering Conference (RE), 2011 19th IEEE International, IEEE, pp. 69–78.
[36] M. Aasem, M. Ramzan, A. Jaffar, Analysis and optimization of software requirements prioritization techniques, in: Information and Emerging Technologies (ICIET), 2010 International Conference on, IEEE, pp. 1–6.
[37] Z. Racheva, M. Daneva, A. Herrmann, R. Wieringa, A conceptual model and process for client-driven agile requirements prioritization, in: Research Challenges in Information Science (RCIS), 2010 Fourth International Conference on, IEEE, 2010, pp. 287–298.
[38] C. Otero, E. Dell, A. Qureshi, L. Otero, A quality-based requirement prioritization framework using binary inputs, in: Mathematical/Analytical Modelling and Computer Simulation (AMS), 2010 Fourth Asia International Conference on, IEEE, pp. 187–192.
[39] N. Carod, A. Cechich, Cognitive-driven requirements prioritization: a case study, in: Cognitive Informatics (ICCI), 2010 9th IEEE International Conference on, IEEE, pp. 75–82.
[40] V. Gaur, A. Soni, P. Bedi, An agent-oriented approach to requirements engineering, in: Advance Computing Conference (IACC), 2010 IEEE 2nd International, pp. 449–454.
[41] N. Carod, A. Cechich, Cognitive profiles in understanding and prioritizing requirements: a case study, in: Software Engineering Advances (ICSEA), 2010 Fifth International Conference on, IEEE, pp. 341–346.
[42] S. Marjaie, V. Kulkarni, Recognition of hidden factors in requirements prioritization using factor analysis, in: Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on, pp. 1–5.
[43] P. Chatzipetrou, L. Angelis, P. Rovegard, C. Wohlin, Prioritization of issues and requirements by cumulative voting: a compositional data analysis

framework, in: Software Engineering and Advanced Applications (SEAA), 2010 36th EUROMICRO Conference on, pp. 361–370.
[44] M. Iqbal, A. Zaidi, S. Murtaza, A new requirements prioritization model for market driven products using AHP, in: 2010 International Conference on Data Storage and Data Engineering, IEEE.
[45] M. Sadiq, J. Ahmed, M. Asim, Q. Suman, More on elicitation of software requirements and prioritization using AHP, in: International Conference on Data Storage and Data, Engineering, IEEE, 2010.
[46] M. Beg, R. Verma, A. Joshi, Reduction in number of comparisons for requirement prioritization using B-Tree, in: Advance Computing Conference, 2009. IACC 2009. IEEE International, IEEE, pp. 340–344.
[47] A. Danesh, R. Ahmad, Study of prioritization techniques using students as subjects, in: Information Management and Engineering, 2009, ICIME'09. International Conference on, IEEE, pp. 390–394.
[48] S. Hatton, Choosing the right prioritisation method, in: Software Engineering, 2008. ASWEC 2008. 19th Australian Conference on, IEEE, pp. 517–526.
[49] D. Port, A. Olkov, T. Menzies, Using simulation to investigate requirements prioritization strategies, in: Proceedings of the 2008 23rd IEEE/ACM International Conference on Automated Software Engineering, IEEE Computer Society, pp. 268–277.
[50] M. Daneva, A. Herrmann, Requirements prioritization based on benefit and cost prediction: a method classification framework, in: EUROMICRO-SEAA, IEEE, 2008, pp. 240–247.
[51] R. Beg, Q. Abbas, R.P. Verma, An approach for requirement prioritization using b-tree, in: Emerging Trends in Engineering and Technology, 2008. ICETET'08. First International Conference on, IEEE, pp. 1216–1221.
[52] K. Logue, K. McDaid, Agile release planning: Dealing with uncertainty in development time and business value, in: Engineering of Computer Based Systems, 2008. ECBS 2008, 15th Annual IEEE International Conference and Workshop on the IEEE, pp. 437–442.
[53] P. Laurent, J. Cleland-Huang, C. Duan, Towards automated requirements triage, in: 15th IEEE International Requirements, Engineering Conference, RE'07, 2007, pp. 131–140.
[54] P. Avesani, C. Bazzanella, A. Perini, A. Susi, Facing scalability issues in requirements prioritization with machine learning techniques, in: RE 2005, pp. 297–306.
[55] P. Avesani, C. Bazzanella, A. Perini, A. Susi, Supporting the requirements prioritization process. A machine learning approach, in: Proceedings of 16th International Conference on Software Engineering and Knowledge Engineering (SEKE 2004), KSI Press, Banff, Alberta, Canada, 2004, pp. 306–311.
[56] X. Liu, S. Chandra, Y. Sun, K. Noguchi, Y. Kyoya, Priority assessment of software requirements from multiple perspectives, in: Proceedings of the 28th Annual International Computer Software and Applications Conference, IEEE, 2004, pp. 410–415.
[57] F. Moisiadis, The Fundamentals of Prioritizing Requirements, Proceedings of Systems Engineering Test and Evaluation Conference (SETE2002).
[58] J. Park, D. Port, B. Boehm, Supporting distributed collaborative prioritization, in: Software Engineering Conference, 1999 (APSEC'99) Proceedings, Sixth Asia Pacific, IEEE, pp. 560–563.
[59] J. Karlsson, Software requirements prioritizing, in: Proceedings of 2nd International Conference on Requirements Engineering (ICRE'96), April 1996, pp. 110–116.
[60] M. Pergher, B. Rossi, Requirements prioritization in software engineering: a systematic mapping study, in: Empirical Requirements Engineering (EmpiRE), 2013 IEEE Third International Workshop on, IEEE, pp. 40–44.
[61] A. Felfernig, G. Ninaus. Group recommendation algorithms for requirements prioritization, in: Recommendation Systems for Software Engineering (RSSE), 2012 Third International Workshop on, IEEE, pp. 59–62.
[62] P. Fitsilis, V. Gerogiannis, L. Anthopoulos, I. Savvas, Supporting the requirements prioritization process using social network analysis techniques, in: Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE), 2010 19th IEEE International Workshop on, IEEE, pp. 110–115.
[63] K.M. Aaron, N. Paul, A.I. Anton, Prioritizing legal requirements, in: Second International Workshop on Requirements Engineering and Law, 2009 (relaw'09), IEEE, pp. 27–32.
[64] M. Svahnberg, A. Karasira, A study on the importance of order in requirements prioritisation, in: Software Product Management (IWSPM), 2009 Third International Workshop on, IEEE, pp. 35–41.
[65] T. Fehlmann, New lanchester theory for requirements prioritization, in: Software Product Management, 2008, IWSPM'08. Second International Workshop on, IEEE, pp. 35–40.
[66] P. Gruenbacher, Collaborative requirements negotiation with easy WinWin, in: Proc. 2nd International Workshop on the Requirements Engineering Process, Greenwich London, September 2000.
[67] H. Benestad, J. Hannay, Does the prioritization technique affect stakeholders' selection of essential software product features?, in: Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ACM, 2012, pp. 261–270.
[68] P. Tonella, A. Susi, F. Palma, Using interactive GA for requirements prioritization, in: Search Based Software Engineering (SSBSE), 2010 Second International Symposium on, IEEE, pp. 57–66.
[69] A. Pitangueira, R. Maciel, M. Barros, A. Andrade, A systematic review of software requirements selection and prioritization using SBSE approaches, in: Search Based Software Engineering, 2013, Springer, Berlin Heidelberg, pp. 188–208.
[70] Z. Bakalova, M. Daneva, A. Herrmann, R. Wieringa, Agile requirements prioritization: what happens in practice and what is described in literature?, in: Requirements Engineering: Foundation for Software Quality, Springer, Berlin Heidelberg, 2011, pp 181–195.
[71] D. Lima, F. Freitas, G. Campos, J. Souza, A fuzzy approach to requirements prioritization, in: Search Based Software Engineering, Springer, Berlin Heidelberg, 2011, pp. 64–69.
[72] V. Gaur, A. Soni, Evaluating degree of dependency from domain knowledge using fuzzy inference system, in: Trends in Computer Science, Engineering and Information Technology, Springer, Berlin Heidelberg, pp. 101–111.
[73] T. Bebensee, I. van de Weerd, S. Brinkkemper, Binary priority list for prioritizing software requirements, in: Requirements Engineering: Foundation for Software Quality, Springer, Berlin Heidelberg, 2010, pp. 67–78.
[74] C. Duan, P. Laurent, J. Cleland-Huang, C. Kwiatkowski, Towards automated requirements prioritization and triage, Requir. Eng. 14 (2) (2009) 73–89.
[75] N. Carod, A. Cechich, Requirements Prioritization Techniques, 2001.
[76] A. Herrmann, B. Paech, Practical challenges of requirements prioritization based on risk estimation, Emp. Softw. Eng. 14 (6) (2009) 644–684.
[77] S. Mohamed, I. ElMaddah, A. Wahba, Towards value-based requirements prioritization for software product management, Int. J. Softw. Eng. 1 (2) (2008) 35–48.
[78] L. Karlsson, T. Thelin, B. Regnell, P. Berander, C. Wohlin, Pair-wise comparisons versus planning game partitioning-experiments on requirements prioritisation techniques, Emp. Softw. Eng. 12 (1) (2006) 3–33.
[79] L. Lehtola, M. Kauppinen, Suitability of requirements prioritization methods for market-driven software product development, Softw. Process: Improve. Pract. 11 (1) (2006) 7–19.
[80] P. Berander, A. Andrews, Requirements prioritization, in: Engineering and Managing Software Requirements, Springer, Berlin Heidelberg, 2005, pp. 69–94.
[81] L. Lehtola, M. Kauppinen, S. Kujala, Requirements prioritization challenges in practice, in: Product Focused Software Process Improvement, Springer, Berlin Heidelberg, 2004, pp. 497–508.
[82] J. Azar, R.K. Smith, D. Cordes, Value-oriented requirements prioritization in a small development organization, IEEE Softw. (2007) 32–73.
[83] J. Karlsson, K. Ryan, A cost-value approach for prioritizing requirements, IEEE Softw. 14 (1997) 67–74.
[84] T.L. Saaty, The Analytic Hierarchy Process, McGraw-Hill, 1980.
[85] A. Herrmann, M. Daneva, Requirements prioritization based on benefit and cost prediction: an agenda for future research, in: RE, IEEE Computer Society, 2008, pp. 125–134.
[86] I. Sommerville, P. Sawyer, Requirements Engineering-A Good Practice Guide, Lancester University, Wiley, 1997.
[87] K.E. Wiegers, First Things First: Prioritizing Requirements, Software, Development, vol. 7, no. 9, September 1999. <http://www.processimpact.com/pubs.shtml#requirements>.
[88] B. Regnell, M. Host, J. Dag, An industrial case study on distributed prioritization in market-driven requirements engineering for packaged software, Requir. Eng. 6 (2001) 51–62.
[89] D. Edwin, Quality function deployment for large systems, in: International Engineering Management Conference '92, Eatontown, NJ, USA, October 25–28, 1992.
[90] H. Kaiya, H. Horai, M. Saeki, AGORA: Attributed goal-oriented requirements analysis method, in: Requirements Engineering, 2002. Proceedings, IEEE Joint International Conference, IEEE, pp. 13–22.
[91] J. Giesen, A. Volker, Requirements interdependencies and stakeholder's preferences, in: Requirements Engineering, 2002. Proceedings of IEEE Joint International Conference, 2002, pp. 206–209.
[92] H. P. In, D. Olson, T. Rodgers, Multi-criteria preference analysis for systematic requirements negotiation, in: Computer Software and Applications Conference, 2002. COMPSAC 2002, Proceedings of 26th Annual International, IEEE, pp. 887–892.
[93] D.M. Berry, The importance of ignorance in requirements engineering, J. Syst. Softw. 28 (2) (1995) 179–184.
[94] P. Berander, Prioritization of Stakeholder Needs in Software Engineering, Understanding and Evaluation. Licenciate Thesis, Blekinge Institute of Technology, Sweden, Licentiate Series, 2004, p. 12.
[95] J. Karlsson, S. Olsson, K. Ryan, Improved practical support for large scale requirements prioritizing, J. Requir. Eng. 2 (1997) 51–67.
[96] S. Peng, Sample Selection: An Algorithm for Requirements Prioritization, ACM, 2008.
[97] Z. Racheva, M. Daneva, L. Buglione, Supporting the dynamic reprioritization of requirements in agile development of software products, in: Software Product Management, IWSPM'08. Second International Workshop on, IEEE, 2008, pp. 49–58.
[98] M. Ramzan, J. Arfan, I. Allliad, S. Anwar, A. Shahid, Value based fuzzy requirement prioritization and its evaluation framework, in: 2009 Fourth International Conference on Innovative Computing, Information and Control, IEEE, 2009, pp. 1464–1468.
[99] P. Grunbacher, M. Halling, S. Biffl, H. Kitapci, B. Boehm, Repeatable quality assurance techniques for requirements negotiations, in: System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on, IEEE, pp. 9–17.

[100] S. Barney, A. Aurum, C. Wohlin, Quest for a silver bullet: creating software product value through requirements selection, in: Software Engineering and Advanced Applications, SEAA'06. 32nd EUROMICRO Conference on, IEEE, 2006, pp. 274–281.
[101] H. Olson, T. Rodgers, Multi-criteria preference analysis for systematic requirements negotiation, in: COMPSAC 2002, pp. 887–892.
[102] L. Karlsson, M. Höst, B. Regnell, Evaluating the practical use of different measurement scales in requirements prioritisation, in: Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering, ACM, pp. 326–335.
[103] S. Sivzittian, B. Nuseibeh, Linking the Selection of Requirements to Market Value: A Portfolio-based Approach, in: REFSQ 2001.
[104] K. Khan, A systematic review of software requirements prioritization, Master's Thesis, Blekinge Institute of Technology, Ronneby, Sweden, 2006.
[105] X. Liu, A quantitative approach for assessing the priorities of software quality requirements, J. Syst. Softw. (1998) 105–113.
[106] X. Liu, K. Noguchi, W. Zhou, Requirement acquisition, analysis, and synthesis in quality function deployment, Int. J. Concurr. Eng. 9 (1) (2001) 1–5.
[107] L. Karlsson, P. Berander, B. Regnell, C. Wohlin, Requirements prioritization: an experiment on exhaustive pair wise comparisons versus planning game partitioning, in: Proceedings of Empirical Assessment in Software Engineering (EASE2004), Edinburgh, Scotland.