A Systematic Literature Review of Recommender Systems for Requirements Engineering

Nejood Hashim Al-walidi
Department of Information Systems and Technology Faculty of Graduate Studies for Statistical Research

Abdelaziz Khamis
Department of Information Systems and Technology Faculty of Graduate Studies for Statistical Research

Nagy Ramadan
Department of Information Systems and Technology Faculty of Graduate Studies for Statistical Research

ABSTRACT
Requirements Engineering (RE) is the first phase of a software project development. This phase aims to help project stakeholders discover, analyze, and specify the needs for a software project. In complex projects, requirement engineers deal with substantial requirements specifications and a large number of stakeholders. Studies have shown that poorly implemented RE processes are one of the primary causes of project failures including cost/schedule overruns, and failure to deliver promised functions. On the other hand, Recommender Systems (RSs) are software tools that support users in the recognition of appropriate items in a context where the amount of an assortment exceeds their capability to reach a decision. Therefore, RSs are needed to support several processes in requirements engineering. In this paper, the utilization of RSs in RE is examined by the use of a Systematic Literature Review (SLR) through the years 2010 - 2019. The results show how recommender systems can support several processes in requirements engineering. Finally, the utilization of recommender systems in requirements traceability is suggested as future work.

Keywords
Recommender Systems; Software Requirements; Requirements Engineering; Requirements Engineering Processes; Systematic Literature Review

1. INTRODUCTION
In his book "Software Engineering", Sommerville defined software requirements as "the descriptions of the services provided by the system and its operational constraints" [1]. There are two classes of software requirements namely, functional requirements and non-functional requirements. Functional requirements refer to the services offered by the system, while non-functional requirements refer to the constraints on the services offered by the system.

The term, requirements engineering, refers to the process of discovering, analyzing, documenting, and checking software requirements. Requirements engineering is the first phase of a software project development. This phase includes four high-level RE sub-processes: feasibility study, elicitation and analysis, validation, and management [1].

Requirements elicitation and analysis is an iterative process that includes four activities: Requirements discovery, Requirements classification and organization, Requirements prioritization and negotiation, and Requirements documentation. Requirements classification and organization is concerned with identifying overlapping requirements from different stakeholders and grouping related requirements. Stakeholders may have different views on the importance and priority of requirements, and sometimes these views conflict. Therefore, one needs to organize stakeholder negotiations to reach compromises [1].

Requirements validation is an important RE sub-process because errors in a requirements document can lead to extensive rework costs when one discover them in later phases of the software project development. One may use a number of requirements validation techniques in conjunctions or individually. These techniques include Requirements reviews, Prototyping, and Test-case generation. In a formal requirements review, the reviewers check the requirements for consistency, completeness, traceability and adaptability [1].

Requirements management is the process of understanding and controlling changes to requirements. This process should start as soon as a draft version of the requirements document is available. During the requirements management stage, one has to decide on Requirements identification, Change management process, Traceability policies, and CASE tool support [1].

One needs to apply the requirements change management to all proposed changes to the requirements. There are three principal stages to a change management process. First, the change proposal is analyzed to check that it is valid. Second, the effect of proposed change is assessed using traceability information, and the cost of making the change is estimated in terms of the modifications to the requirements document. Finally, if it was decided to proceed with the requirements change, the requirements document and, where necessary, the system design and implementation are modified [1].

Requirements traceability is important as it can be used to measure progress, manage change and assess risks. Trace links allow you to follow the life of a requirement both forward and backward, from origin through implementation.

Requirements trace information documents the dependencies between individual requirements and other system elements such as design components, source code modules, and tests. Trace information facilitates impact analysis by helping you identify all the work systems you might have to modify to implement a proposed requirement change [2].

Studies have shown that poorly implemented RE processes are still one of the primary causes of project failure. Consequently, one might notice an increasing demand for intelligent software tools that support stakeholders in the completion of RE processes [3]. On the other hand, recommender systems are software tools that support users in the recognition of appropriate items in a context where the

amount of an assortment exceeds their capability to reach a decision. Therefore, RSs are needed to support several processes in requirements engineering, and help software developers to work more effectively.

The process of building an effective recommender includes five steps: framing the problem, determining the inputs, building the recommender, delivering the recommendations, and evaluation. In [4], the authors described each of these steps using an example of a recommender system. However, for the context of this paper, framing the problem step will be briefly described.

The first step in building a recommender system mainly aims to determine what problem the recommender system will solve and what value the recommender system will provide [4]. This involves many activities that enable answering the following questions: "(1) Who will be the user of the recommender system? (2) What problem the recommender system will solve? (3) Which solution the recommender system will offer? (4) What is the value proposition of the recommender system?"

The content of the remaining parts of this paper are as follows. Section 2 discusses some of the related literature reviews. The methodology used for this study is described in Section 3. Section 4 presents the results from the analysis of this SLR. Section 5 discusses the results in relation to the addressed research questions. Finally, section 6 contains the conclusions and future work.

2. RELATED SLRs

This section contains an overview the literature related to recent research directions in requirements engineering, recommender systems, and their intersection area. The focus will be on the addressed problems and the techniques used to solve these problems.

2.1 Requirements Engineering

In the paper titled "A systematic mapping study on crowdsourced requirements engineering using user feedback", the authors addressed the lacking of a structured account on a promising form of crowdsourcing in RE by the use of a systematic mapping study. They investigated four research questions. "(1) What sources of implicit and explicit crowdsourced user feedback have been reported in RE activities? (2) What metadata of crowdsourced user feedback are reported as being useful for RE? (3) In which RE activities, has the crowdsourced user feedback been applied? (4) What are the demographics of the research on applying crowdsourced user feedback for crowd- based RE?" Regarding the use of explicit and implicit feedback, the results showed that more than three quarters of the included studies on explicit user feedback did concentrate on app reviews, and call for more investigation on employing implicit user feedback in Requirements Engineering. Regarding the attributes of crowdsourced user feedback, the results identified six pieces of feedback metadata. Concerning the activities employing user feedback, the results indicated that requirements elicitation and analysis are the two most popular RE activities, which employ crowdsourced user feedback. Finally, regarding the demographics of the studies, the results showed that the topic of these studies has received the most attention from researchers; most of them are working at universities spreading in 12 countries, mostly located in Europe and Asia [5].

In another paper titled "Requirements Engineering Techniques: A Systematic Literature Review", the authors addressed the problem of software projects failure in spite of the proposed numerous techniques to be used in RE, by identifying the gaps in these techniques. They investigated three research questions: "(1) What are the techniques used in requirements engineering? (2) What are the limitations of the existing techniques used in requirements engineering? (3) How do changing software requirements affect requirements analysis?" Regarding the techniques used in RE, the review identified forty-three techniques used in RE, and indicated that there is no RE technique that can best solve all the issues in the software requirements domain. Concerning the limitation of the existing RE techniques, the results showed that the techniques used in RE usually addresses one activity of the RE process, and do not provide support for dynamic environment. Finally, regarding the effect of changing requirements, one needs to carry out a proper change impact analysis so that the change will not cost the project in terms of the delivery time and in monetary terms [6].

In the paper titled "A systematic literature review of stakeholder identification methods in requirements elicitation", the authors addressed the problem of Stakeholder Identification (SI) in the area of requirements elicitation. In this area, it is critical to describe the SI process in order to provide correct, consistent, and complete requirements specification. The review investigated four research questions: "(1) What methods or techniques are currently used to carry out Stakeholder Identification (SI) in requirements elicitation? (2) What are the recommended effective practices for performing SI? (3) What are the consequences of incorrect SI on the quality of Software Requirements? (4) What aspects of IS are necessary to use as advisable practices?" The results showed that the current SI approaches have limitations in terms of covering all aspects of SI. However, through correctly identifying and understanding the stakeholders, it is possible to develop high quality software. Finally, the obtained findings provided strong evidence to encourage further research in the development of a new methodology to perform SI adequately [7].

2.2 Recommender Systems

In their paper titled "Approaches, Issues and Challenges in Recommender Systems: A Systematic Review" the authors provided a comprehensive and systematic review of the state-of-the-art recommender systems. The literature review process was divided into six research questions: (1) What kinds of approaches are used for generating recommendations by RSs? (2) What are the strengths and weaknesses of recommendation approaches practiced in RSs field? (3) What kinds of issues and challenges encounter in deployment of RSs? (4) What are the various application domains where RSs being adopted? (5) Which evaluation methods one may use to measure the quality of RSs? (6) What are the different gaps exist in the present RSs research?." The results showed that collaborative filtering and content-based filtering approaches got wide acceptance and extensive usage by the research community over other techniques available in the RSs field. Regarding the pros and cons of recommendation approaches, the authors identified a list of strengths and weaknesses for each approach. In addition, the paper described the most common issues and challenges that encounter in deploying RSs. Regarding recommender application domains, the results showed that there are certain application fields that need to grab the attention of scientific and research communities. Concerning the evaluation measures of RSs, the results showed that many RS research studies apply different evaluation methods especially the ranking measures. Finally,

regarding the gaps that exist in the present RSs research, one may conclude that the new RSs research will focus on advancing the existing approaches and algorithms to enhance the quality of recommendations [8].

In another study titled "A recommendation Systems in Education: A Systematic Mapping Study", the authors addressed the impact of RSs in the education domain by performing a systematic mapping study. The study investigated five research questions: "(1) What are the educational areas covered by RSs? (2) What are the approaches used to generate recommendations within educational scenarios? (3) Which platform is used for the RS deployment?" (4) Which evaluation or validation strategies one may apply to RSs? (5) What are the challenges addressed by adopting RSs in the educational context?" The results showed significant interest in the use of RSs in educational scenarios, which include helping on academic choices, assisting in suggesting courses, and e-learning. About the approaches used to generate recommendations, the study revealed that the most frequent approach used in education is the Hybrid approach. Regarding the RS development platform, the Web is the most adopted platform for using RS in education; other platforms are the desktop-based and mobile-based platforms. Regarding RSs evaluation, the results showed that about third of the primary studies did not conduct any type of validation, and the applied evaluation strategies to RSs include experimentation, survey, and case study. Finally, the issues addressed by adopting RSs include how to provide personalized recommendations, prediction accuracy and efficiency, and improve educational practices [9].

The paper titled "Hybrid Recommender Systems: The Review of State-of-the-Art Research and Applications" addressed the weaknesses of individual RSs using hybrid RSs. The authors investigated three research questions: "(1) What are the actual statements of the field of hybrid RSs? (2) What are the presented types of hybrid RSs? (3) What kind of input data is usually used in RS?" The results showed that one might use hybrid RSs for different problem solutions in different areas including e-commerce, tourism, and medicine. About the types of hybrid RSs, the results showed seven different types of hybrid RSs, and the most common type is the one that combines the collaborative filtering and content-based filtering approaches. Finally, regarding the kind of input data, the most popular input date is the information about users' preferences and item features. However, to date, all types of data are used [10].

2.3 Recommender Systems for Requirements Engineering

In their paper titled "Systematic Mapping of Recommendation Systems for Requirements Engineering", the authors investigated three research questions about what existing recommender systems are used to support RE activities, and what are their characteristics and state of validation?. The results showed that the mostly used technique to generate recommendations is the collaborative filtering technique, and the majority of the studies address requirements elicitation and analysis activities, and few consider requirements validation. In addition, the results indicated that there is none or only limited evaluation of recommender systems. Evaluating recommender systems is necessary to truly understand the effectiveness of the system and help with the adoption of the systems in real life projects. Finally, the results pointed out some concepts and techniques that are of use in assisting requirements engineers in their task, but not within the context of recommendation systems. However, one might utilize and incorporate these concepts and techniques in RSs for RE [11].

3. RESEARCH METHODOLOGY

This research has been conducted as a systematic literature review which is a mean of collecting evidence-based data about a research topic and its related issues [12]. The objective of this research is to identify the state-of-the-art on recommender systems for requirements engineering through the years 2010- 2019.

The SLR methodology is based on the guidelines by Kitchenham and Charters [13]. According to these guidelines, an SLR process is composed of three main phases:

- Planning the review (See Section 3.1)
- Conducting the review (See Section 3.2)
- Reporting the review (See Sections 4 and 5)

3.1 Planning the Review

This phase aims at developing a review protocol. The steps involved in planning the review include specifying the research questions that the review is intended to answer (See Section 3.1.1), the strategy that will be used to search for primary studies (See Section 3.1.2), and the study selection criteria (See Section 3.1.3).

3.1.1 Specifying the Research Questions

Generally, specifying research questions is considered as one of the most important steps of the SLR because the research questions manage the entire literature review methodology [14]. The principal goal of this SLR is to examine the utilization of RSs in RE. In order to achieve this goal, one needs to address the following research questions (RQs):

RQ1: What are the problems that were addressed by RSs in RE?

RQ2: In what contexts the problems found in **RQ1**were addressed?

RQ3: What are the solutions offered by RSs to the problems found in **RQ1**?

RQ4: What types of RSs were utilized in solving the problems found in **RQ1**?

3.1.2 Search Strategy

The search strategy should specify the *search terms* and the *sources* to be searched, so that every relevant publication has a very good chance to appear in the research results [14].The sources that were used during the SLR are specified in Table 1.

Table 1. Digital Libraries used in the SLR

Name	Website
Google Scholar	https://scholar.google.com/
IEEE Xplorer	https://ieeexplore.ieee.org/
World Scientific	https://www.worldscientific.com/
Dogpile	https://www.dogpile.com/
Research Gate	https://www.researchgate.net/
ACM Digital Library	https://dl.acm.org/

Semantic Scholar	https://www.semanticscholar.org/
Google	https://www.google.com/

The following search terms were used to extract primary studies:

1) Software requirements
2) Requirements engineering
3) Recommender systems
4) Recommendation systems
5) Recommender systems in requirements engineering
6) (2) AND Problems
7) (2) AND Issues
8) (2) AND Challenges
9) (5) AND Types
10) (5) AND Techniques
11) (5) AND Approaches
12) (5) AND Methods
13) (5) AND Tasks
14) (5) AND Context
15) (5) AND Solutions

3.1.3 Study Selection Criteria

Study selection criteria are intended to identify those primary studies that provide direct evidence about the research questions. They are used to determine which primary studies are to be included in, or excluded from an SLR. The inclusion and exclusion criteria for this SLR are summarized in Table 2.

Table 2. Study Selection Criteria

Inclusion Criteria	Exclusion Criteria
• The time span for primary studies is 2010-2019. • Primary studies that provide evidence about the research questions. • Primary studies that propose solutions offered by RSs in RE. • Primary studies that propose effective RSs to be utilized in RE. • Primary studies that are published in refereed journals or conferences.	• Primary studies that are out of the defined time span. • Primary studies that are not relevant to the research questions. • Every study that is not published in refereed journals or conferences. • Primary studies having more than one published sources are included only once using the latest versions of the studies.

3.2 Conducting the Review

This phase starts after the approval of the review protocol. The steps involved in conducting the review include selection of primary studies (See section 3.2.1), study quality assessment (See section 3.2.2), and data extraction and synthesis (See Section 3.2.3).

3.2.1 Selection of Primary Studies

Based on the serch strategy and the study selection criteria presented above, an initial pool of 208 studies was created. Then, 33 studies were selected after applying inclusion and exclusion criteria. Finally, based on this research objective, 18 studies are identified as primary studies as shwon in Table 3.

3.2.2 Study Quality Assessment

After selecting the primary studies, one should assess their quality using Quality Assessment (QA) questions. These questions are used in performing analysis of each study with the aim of giving a judgment about the credibility of the research it present. In this SLR, one may adopt the following QA questions [15]:

"QA1.How clear and coherent is the work? QA2. How clear is the research goal defined? QA3. How well can the route from the research goal to any conclusion are seen?" QA4. How clearly is the search process established? QA5. how good is the work in comparison to other related works? QA6. How clearly are the work limitations documented?"

Table 3. Primary Studies used in SLR

Ref.	Title
[16]	Semi-Automated Feature Traceability with Embedded Annotations
[17]	Recommender Systems for Software Requirements Negotiation and Prioritization
[18]	A Novel Recommender System Based on Apriori Algorithm for Requirements Engineering
[19]	Utilizing Recommender Systems to Support Software Requirements Elicitation
[20]	Context-Aware Recommender Systems for Non-functional Requirements
[21]	Requirement Elicitation Based Collaborative Filtering Using Social Networks
[22]	A Novel Method for Large Scale Requirement Elicitation
[23]	Recommendation and Decision Technologies For Requirements Engineering
[24]	Group Decision Support for Requirements Negotiation
[25]	REQAnalytics: A Recommender System for Requirements Maintenance
[26]	Functional Requirements Identification Using Item-to-Item Collaborative Filtering
[27]	An Automated Approach to Requirement Elicitation Using Stakeholder Recommendation and Prediction Analysis
[28]	A New Approach to Requirement Elicitation Based on Stakeholder Recommendation and Collaborative Filtering
[29]	INTELLIREQ: Intelligent Techniques for Software RE

[30]	Personal Recommendations in Requirements Engineering: The OpenReq Approach
[31]	A Non-Functional Requirements Recommendation System for Scrum-based Projects
[32]	An experimental study on Collaborative Filtering for RE
[33]	A Proposed RS for Eliciting Software Sustainability Requirements

3.2.3 Data Extraction and Synthesis

This step defines how the information required from each primary study will be obtained and synthesized. The obtained information will enable us to answer the research questions identified in section (1). To organize the data extraction and synthesis process, researchers often use a form or table to capture the data they will then synthesize. Table 4 contains the elements of data extraction and synthesis for this SLR.

Table 4. Elements of Data Extraction Synthesis

Item #	Description	Details
1	Bibliographic Information	Name of authors, year of publication, etc.
Extraction of Data		
2	Overview	The basic objective of the selected study is finding out the problems addressed by RSs in RE, the contexts in which these problems were addressed, the solutions offered by RSs to these problems, and the types of RSs that were utilized.
3	Results	Results achieved in the selected study.
Synthesis of Data		
4	Problems	The problems addressed by RSs in RE (See Section IV)
5	Context	The context in which these problems were addressed (See Section IV)
6	Solutions	The solutions offered by RSs these problems (See Section IV)
7	Types	The types of RSs that were utilized (See Section 4)

4. RESULT

This section contains a summary of the obtained results after analyzing the selected primary studies. One may define the four-step analysis process for each primary study in this SLR as follows. The first step defines the problems in requirements engineering that have been addressed using recommender systems. The second step defines the context in which the identified problems were addresses. The solutions of the defined problems are identified in the third steps. Finally, the fourth step identifies the types of the utilized RSs. Table 5 contains the summary of these results together with their references and Table 6 contains the experimental results for the primary studies.

5. DISCUSSION

The results in section 4 show how recommender systems can support several processes in requirements engineering through the years 2010 - 2019. This section contains an analysis of these results with aim of answering the four research questions defined in Section 3.

5.1 What are the problems that were addressed by RSs in RE?

Requirements engineering in large-scale projects can be a highly complex process that result in massive amounts of data that must be analyzed in order to extract useful requirements. As a result, the RE process can be supported through the use of RSs. The findings in section 4 show that the problems that were addressed by RSs in RE are mainly found during requirements elicitation, and requirements analysis and negotiation activities. Table 7 contains the classification of these problems.

Table 5. Results Summary

RE Problem	Context	Solution	RS Type	Papers
The accuracy of current feature-location techniques are too low to be useful in practice.	long-living or variant-rich software with many developers.	A semi-automated, machine-learning assisted feature-traceability technique based on a recommender system.	Collaborative Filtering	[16]
Handling requirements negotiation and prioritization.	Institute Examination System's requirements.	A method based on RS for requirements negotiation and prioritization.	Content Based Filtering	[17]
Lack of accuracy and completeness of gathered requirements	A synthesized dataset of requirements containing 4000 Records. (Averaging: 10 requirements per record).	A RS that improves the accuracy of the obtained requirements and produce more comprehensive results.	Collaborative Filtering	[18]
Data overload during the online requirements elicitation process.	Online forums and wikis.	Provide timely and useful recommendations to stakeholders, discussion group members, and project managers as they engage in requirements gathering activities.	Hybrid RS	[19]
Handling efficiency requirements during the software development process.	A sorting application for a range of problem sizes.	Suggesting the best-fit component variants for certain actual contexts which are later on used by a composition technique to improve	Context-Aware RS	[20]

		application runtime performance.		
Information overload, incomplete requirement and inadequate stakeholder.	Large scale software projects	A method called StakeRare is developed - using social networks and collaborative filtering - to identify and prioritize stakeholders and their requirements.	Collaborative Filtering	[21]
Handling difficulties in requirements elicitation in large scale software projects with many stakeholders.	A large-scale software project (RALIC project)	A system that that supports requirements elicitation in large-scale software projects.	Collaborative Filtering	[22]
Deciding by stakeholders on which requirements should be taken into account.	Industrial software projects.	A system that supports decision processes.	Hybrid RS	[23]
Resolving conflicts between requirements and deciding which requirements should be implemented.	Software development projects at Graz University of Technology.	An environment that supports group decision processes in requirements negotiation.	Hybrid RS	[24]
Frequent changes of software requirements.	Software as a Service (SaaS).	A recommender system that generates recommendations that can increase the quality of that service.	Knowledge Based Filtering	[25]
Identifying well-defined functional requirements.	Completed software project documentations.	A recommender model to recommend functional requirements.	Collaborative Filtering	[26]
Information overload, inadequate stakeholder input, and biased prioritization of requirements.	Large scale software projects.	A method that uses social networks and collaborative filtering to identify and prioritize requirements.	Collaborative Filtering	[27]
Current methods to identify and prioritize requirements do not scale well to large projects.	Large scale software projects.	A system that identifies and prioritizes SRs for large scale projects.	Collaborative Filtering	[28]
Handling low-quality requirements.	Software development projects at Graz University of Technology.	An environment that support stakeholders in requirements-related activities such as quality assurance.	Hybrid RS	[29]
Current applications that use RSs in RE focus on specific RE tasks and do not give a general coverage for the RE process.	Large and distributed systems	An open source tool and a set of APIs that integrate recommendation and decision technologies to support different phases of RE.	Hybrid RS	[30]
Neglecting non-functional requirements until the later stages of software development.	Scrum-based Projects.	A non-functional requirements recommendation system to support Scrum practitioners on their early identification.	Collaborative Filtering	[31]
Difficulties in the interactions between the stakeholders in requirements elicitation and management.	A real case study.	An optimization based system that identifies and prioritizes SRs.	Collaborative Filtering	[32]
The barriers of incorporating sustainability into the software engineering process.	Software Engineering for Sustainability (SE4S) projects.	A system that recommends the kinds of sustainability requirements that should be considered.	Hybrid RS	[33]

Table 6. Experimental Results of the Primary Studies

RE Problem	Experimental Results	Papers
The accuracy of current feature-location techniques are too low to be useful in practice.	An overall accuracy of about 50 % has been reached.	[16]
Handling requirements negotiation and prioritization	In a case study, the proposed recommender system applied to find Examination System's requirements and prioritize these requirements according to the stakeholders.	[17]
Lack of accuracy and completeness of gathered requirements	Experimental work showed that the proposed recommender system would improve the accuracy of the obtained requirements and	[18]

	produce more comprehensive results.	
Data overload during the online requirements elicitation process.	Experimental results showed using a K-Nearest Neighbor (KNN) collaborative recommender system provides the best recommendation results.	[19]
Handling efficiency requirements during the software development process.	The results showed how performance, a non-functional requirement, in a Sorting application was guaranteed by a context-aware recommender system.	[20]
Information overload, incomplete requirement and inadequate stakeholder.	Complete and accurate requirements are collected for projects and stakeholders are identified correctly though a method that is developed using social networks and collaborative filtering.	[21]
Handling difficulties in requirements elicitation in large scale software projects with many stakeholders.	The proposed method identified the requirements in a large-scale project with a high level of completeness, and with a 10 percent higher recall compared to the existing methods.	[22]
Deciding by stakeholders on which requirements should be taken into account.	In the proposed system, recommendation and decision technologies support two basic scenarios: decision processes of individual stakeholders and group decision processes.	[23]
Resolving conflicts between requirements and deciding which requirements should be implemented.	Experimental results showed that the effects of applying group recommendation technologies to requirements negotiation within the scope of software development projects at Graz University of Technology, Austria.	[24]
Frequent changes of software requirements	In a case study, the results show that the analysis of the use of a website provide recommendations that allow the website to meet the expectations of its customers and users.	[25]
Identifying well-defined functional requirements.	The result shows item-to-item collaborative filtering had been proven one of the most successful techniques in the development of a recommender system to recommend functional requirements.	[26]
Information overload, inadequate stakeholder input, and biased prioritization of requirements.	The proposed method used social networks and collaborative filtering to identify and prioritize a complete set of stakeholders and their requirements automatically and accurately.	[27]
Current methods to identify and prioritize requirements do not scale well to large projects.	The proposed approach uses social networks and collaborative filtering to identify and prioritize highly complete set of requirements compared to the existing method used.	[28]
Handling low-quality requirements.	The reported results of empirical studies showed in which way the recommendation approaches integrated in requirements engineering environment, could improve the quality of requirements.	[29]
Current applications that use RSs in RE focus on specific RE tasks and do not give a general coverage for the RE processes.	The proposed approach is intended to support different phase of RE in software projects. It assists stakeholders with personal recommendations during the RE process.	[30]
Neglecting non-functional requirements until the later stages of software development.	The proposed solution showed a recall rate of up to 81%, which indicates that it is a promising approach to recommend non-functional requirements.	[31]
Difficulties in the interactions between the stakeholders in requirements elicitation and management.	The proposed RS could be employed for the problem of the requirements elicitation and management. The results showed its validation against a real case study.	[32]
The barriers of incorporating sustainability into the software engineering process.	The results showed that the proposed RS lessens the workload of eliciting appropriate sustainability requirements though enhancing the knowledge of sustainability and related types of requirements.	[33]

5.2 What contexts the problems found in RQ1 were addressed?

The notion of the context of a problem refers to the environment in which the problem is being addressed. The findings in section 4 show that contexts in which the problems found in RQ1 were addressed may be identified as follows:

- Scrum-based projects
- Online forums and wikis
- Industrial software projects
- Software as a Service (SaaS)
- Large scale software projects

- Large and distributed systems
- Synthesized datasets of requirements
- Completed software project documentations
- Specific applications for a range of problem sizes
- Software Engineering for Sustainability (SE4S) projects
- Software development projects at Educational Institutions
- Long-living or variant-rich software with many developers

Table 7. Classifying the Problems Addressed by RSs

RE Activity	Problem	Ref.
Elicitation	Data overload during the online requirements elicitation process	[19]
	Handling efficiency requirements during the software development process	[20]
	Handling difficulties in requirements elicitation in large scale software projects with many stakeholders	[22]
	Information overload, incomplete requirement and inadequate stakeholder	[21]
	The barriers of incorporating sustainability into the software engineering process	[33]
	Identifying well-defined functional requirements	[26]
	Information overload, inadequate stakeholder input, and biased prioritization of requirements	[27]
	Neglecting non-functional requirements until the later stages of software development	[31]
Analysis and Negotiation	Deciding by stakeholders on which requirements should be taken into account	[23]
	Resolving conflicts between requirements and deciding which requirements should be implemented	[24]
	Current methods to identify and prioritize requirements do not scale well to large projects	[28]
	Current methods to identify and prioritize requirements do not scale well to large projects	[29]
	Handling requirements negotiation and prioritization	[17]
	Lack of accuracy and completeness of gathered enrequiremts	[18]
	Frequent changes of software requirements	[25]
Management	The accuracy of current feature-location techniques are too low to be useful in practice	[16]
Different Activities	Difficulties in the interactions between the stakeholders in requirements elicitation and management	[32]
	Current applications that use RSs in RE focus on specific RE tasks and do not give a general coverage for the RE process	[30]

5.3 What are the solutions offered by RSs to the problems found in RQ1?

For pages other than the first page, start at the top of the page, and continue in double-column format. The two columns on the last page should be as close to equal length as possible.

The requirements engineering problems identified in RQ1 have been solved by developing recommender systems, models, environments, techniques or methods that use recommendation techniques. These systems, models, environments, techniques or methods may be classified according to requirements engineering activities as shown in Table 8.

5.4 What types of RSs were utilized in solving the problems found in RQ1?

There are many types of recommender systems based on: how these systems analyze and filter the available information to provide the user with the information that he/she is interested in. A variety of techniques have been proposed for performing recommendation, resulting in different types of RSs that include: Content-Based RSs, Collaborative Filtering RSs, Context-Aware RSs, Knowledge-Based RSs, and Hybrid RSs [27],[26],[19],[29],[33],[20],[25] In the following paragraphs, These types are described in the following paragraphs. Then, Table 9 contains a classification of the primary studies according to the types of the utilized RSs in solving the problems found in RQ1.

Table 8. Classifying the Solutions offered by RSs

RE Activities	Solutions	Ref.
Elicitation	A RS that provides timely and useful recommendations to stakeholders, discussion group members, and project managers as they engage in requirements gathering activities.	[19]
	A RS that Suggests the best-fit component variants for certain actual contexts which are later on used by a composition technique to improve application runtime performance.	[20]
	A method that supports requirements elicitation in large-scale software projects.	[22]
	A method called StakeRare is developed - using social networks and collaborative filtering - to identify and prioritize stakeholders and their requirements.	[21]

	A RS that recommends the kinds of sustainability requirements that should be considered in a given system.	[33]
	A validated recommender model to recommend functional requirements.	[26]
	A method that uses social networks and collaborative filtering to identify and prioritize requirements.	[27]
	A RS that supports Scrum practitioners on their early identification of non-functional requirements.	[31]
Analysis and Negotiation	A system that supports stakeholders on deciding which requirements should be taken into account.	[23]
	An environment that supports group decision processes in requirements negotiation.	[24]
	A RS that identifies and prioritizes software requirements for large scale projects.	[28]
	An environment that support stakeholders in requirements-related activities such as quality assurance.	[29]
	A method based on recommender systems for requirements negotiation and prioritization	[17]
	A RS that improves the accuracy of the obtained requirements and produce more comprehensive results	[18]
Management	A RS that collects information on the usage of a web service, relates that information back to the requirements, and generates recommendations that can increase the quality of that service	[25]
	A semi-automated, machine-learning assisted feature-traceability technique that allows developers to continuously record feature-traceability information while being supported by recommendations about missed locations	[16]
Different Activities	An optimization based recommender system that handles difficulties in the interactions between the stakeholders in requirements elicitation and management	[32]
	An open source tool and a set of APIs that integrate recommendation and decision technologies to support different phases of RE.	[30]

- *Content-Based Recommender Systems*: These systems recommend items similar to those a given user has liked in the past. In order to recommend a new interesting item, the recommender system match up the attributes of that item with the attributes of a user profile in which preferences are stored. The user profiles are updated automatically based on their feedback.
- *Collaborative Filtering Recommender Systems:* There are two types of these systems, item-based and user-based collaborative filtering systems. The item-based collaborative filtering systems recommend items depending on the rating given by the same user to other items with high correlations. The user-based collaborative filtering systems recommend items depending on the opinion of other similar minded user for these items.
- *Knowledge-Based Recommendation Systems*: These systems recommend advices about decisions to make or actions to take. In order to generate recommendations, they rely on encoded knowledge provided by human experts in the product domain and user's requirements.
- Context-Aware Recommender Systems: These systems recommend items to users in certain circumstances. Therefore, they take the relevant contextual information, such as time, place and weather, into account when providing recommendations. Traditional RSs deal with applications having only two types of entities, users and items.
- *Hybrid Recommender Systems*: These systems combine two or more recommendation techniques in different ways to benefit from their corresponding advantages.

Table 9. Primary Studies Classification According to the Types of the Utilized RSs

RSs Types	Primary Stydis
Content-Based Recommender Systems	[17]
Collaborative Filtering Recommender Systems	[32] [28] [21] [22] [26] [27] [16] [18] [31]
Knowledge-Based Recommendation Systems	[25]
Context-Aware Recommender Systems	[20]
Hybrid Recommender Systems	[19] [23] [24] [33] [29] [30]

6. CONCLUSION AND FUTURE WORK

This study presents a Systematic Literature Review to investigate the latest development in utilizing recommender systems in requirements engineering with the aim of identifying any gaps in current research that need further

investigation. Based on this research objective, 18 studies are identified as primary studies and investigated. This leads to identify the problems in different requirements engineering activities, which have been addressed by utilizing recommender systems. This paper has presented a classification scheme for the addressed problems and their solutions based on the involved requirements engineering activities. In addetion, the utilized recommender systems have been classified according to their types. A noticeable gap in the invesigated primary studies is the lack of utilizing recommender systems in requirements traceability.

Requirements Traceability (RT) enables software engineers to trace a requirement from its emergence to its fulfillment. RT is recognised as a concern in the requirements engineering process. One may attribute this to inadequate tool support. Future research work aims to making use of the recommendation techniques to address some of the problems in requirements traceability.

7. REFERENCES

[1] Sommerville, I. (2007). Software Engineering–Eight Edition. United States of America: Pearson Education Limited.

[2] Wiegers, K. & Beatty, J. (2013). Software Requirements. Washington. Microsoft Press.

[3] Felfernig, A., Jeran, M., Ninaus, G., Reinfrank, F., & Reiterer, S. (2013). Toward the next generation of recommender systems: applications and research challenges. In Multimedia services in intelligent environments, (pp. 81-98).

[4] Proksch, S., Bauer, V., & Murphy, G. C. (2014). How to build a recommendation system for software engineering. In Software Engineering., (pp. 1-42). Springer, Cham.

[5] Wang, C., Daneva, M., van Sinderen, M., & Liang, P. (2019). A systematic mapping study on crowdsourced requirements engineering using user feedback. Journal of Software: Evolution and Process, 31(10), e2199.

[6] Matyokurehwa, K., Mavetera, N., & Jokonya, O. (2017). Requirements Engineering Techniques: A Systematic Literature Review. International Journal of Soft Computing and Engineering,, 7(1), 14-20.

[7] Pacheco, C., & Garcia, I. (2012). A systematic literature review of stakeholder identification methods in requirements elicitation. Journal of Systems and Software, 85(9), 2171-2181.

[8] Kumar, B., & Sharma, N. (2016). Approaches, issues and challenges in recommender systems: a systematic review. Indian J. Sci.Technol, 9(47), 1-12.

[9] Rivera, A. C., Tapia-Leon, M., & Lujan-Mora, S. (2018). Recommendation Systems in Education: A Systematic Mapping Study. In International Conference on Information Theoretic Security, (pp. 937-947). Springer, Cham.Spain.

[10] Danilova, V., & Ponomarev, A. (2017). Hybrid Recommender Systems: The Review of State-of-the-Art Research and Applications. PROCEEDING OF THE 20TH CONFERENCE OF FRUCT ASSOCIATION., 1-7.Saint-Petersburg, Russia.

[11] Mohebzada, J. G., Ruhe, G., & Eberlein, A. (2012). Systematic mapping of recommendation systems for requirements engineering. In 2012 International Conference on Software and System Process (ICSSP), (pp. 200-209). IEEE.

[12] Babar, M. I., Ghazali, M., & Jawawi, D. N. (2014). Systematic reviews in requirements engineering: A systematic review. In 2014 8th. Malaysian Software Engineering Conference (MySEC) , (pp. 43-48). IEEE.

[13] Kitchenham, B.A., Charters, S. (2007). Guidelines for Performing Systematic Literature Reviews in Software Engineering. Technical Report EBSE-2007-01., 1-65.

[14] Javed, M. A., & Zdun, U. (2014). A systematic literature review of traceability approaches between software architecture and source code. In Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering (p. p. (p. 1)). United Kingdom. ACM.

[15] Santiago, I., Jiménez, A., Vara, J. M., De Castro, V., Bollati, V. A., & Marcos, E. (2012). Model-Driven Engineering as a new landscape for traceability management: A systematic literature review. Information and Software Technology. , 54(12), 1340-1.

[16] Abukwaik, H., Burger, A., Andam, B. K., & Berger, T. (2018). Semi-automated feature traceability with embedded annotations. International Conference on Software Maintenance and Evolution (ICSME), (pp. 529-533). IEEE.USA.

[17] Ahmad, S., & Sadiq, M. (2015). Recommender Systems for Software Requirements Negotiation and Prioritization. International Journal of Computer Applications,117(13).

[18] AlZu'bi, S., Hawashin, B., EIBes, M., & Al-Ayyoub, M. (2018). A novel recommender system based on apriori algorithm for requirements engineering. Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS), (pp. 323-327) Irbid, Jordan. IEEE.

[19] Castro-Herrera, C., & Cleland-Huang, J. (2010). Utilizing recommender systems to support software requirements elicitation. In Proceedings of the 2nd International Workshop on Recommendation Systems for Software Engineering, (pp. 6-10). ACM.

[20] Danylenko, A., & Löwe, W. (2012). Context-aware recommender systems for non-functional requirements. In Proceedings of the Third International Workshop on Recommendation Systems for Software Engineering, (pp. 80-84). IEEE Press.

[21] Deepika, P., & Smitha, P. S. . (2013). Requirement elicitation based collaborative filtering using social networks. Journal, 3., 1-4.

[22] Dheepa, V., Aravindhar, D. J., & Vijayalakshmi, C. (2013). A novel method for large scale requirement elicitation. International Journal of Engineering and Innovative Technology, 2(7), 375-379.

[23] Felfernig, A., Schubert, M., Mandl, M., Ricci, F., & Maalej, W. (2010). Recommendation and decision technologies for requirements engineering. In Proceedings of the 2nd International Workshop on Recommendation Systems for Software Engineering, (pp. 11-15). ACM.

[24] Felfernig, A., Zehentner, C., Ninaus, G., Grabner, H., Maalej, W., Pagano, D., & Reinfrank, F. (2011). Group decision support for requirements negotiation. In International Conference on User Modeling, Adaptation, and Personalization., (pp. 105-116)., Springer, Berlin, Heidelberg.

[25] Garcia, J. E., & Paiva, A. C. (2016). REQAnalytics: a recommender system for requirements maintenance. International Journal of Software Engineering and Its Applications, pp. 129-140.

[26] Hidalgo, R. J. F., & Fernandez, P. L. (2015). Functional Requirements Identification Using Item-to-Item Collaborative Filtering. . International Journal of Information and Education Technology,, 5(10), 758.

[27] Latheef, N., & Nithya, A. A. . (2013). An Automated Approach to Requirement Elicitation using Stakeholder Recommendation and Prediction Analysis. In International Conference on Engineering and Technology , (p. 77).

[28] Mulla, N., & Girase, S. (2012). A new approach to requirement elicitation based on stakeholder recommendation and collaborative filtering. International Journal of Software Engineering & Applications,, 3(3), 51.

[29] Ninaus, G., Felfernig, A., Stettinger, M., Reiterer, S., Leitner, G., Weninger, L., & Schanil, W.(2014). INTELLIREQ: Intelligent Techniques for Software Requirements Engineering. In ECAI , (pp. 1161-1166).

[30] Palomares, C., Franch, X., & Fucci, D. (2018, March). Personal recommendations in requirements engineering: the OpenReq approach. In International Working Conference on Requirements Engineering: Foundation for Software Quality, (pp. 297-304). Springer, Cham.

[31] Ramos, F. B. A., Costa, A. A. M., Perkusich, M., Almeida, H. O., & Perkusich, A. (2018). A Non-Functional Requirements Recommendation System for Scrum-based Projects. In SEKE, (pp. 149-148).

[32] Roda, F. (2012). :An experimental study on collaborative filtering for requirements engineering. In 2012 IEEE 17th International Conference on Engineering of Complex Computer Systems, (pp. 23-28). IEEE.

[33] Roher, K., & Richardson, D. (2013). A proposed recommender system for eliciting software sustainability requirements. In 2013 2nd International Workshop on User Evaluations for Software Engineering Researchers (USER), (pp. 16-19). IEEE.