OpenReq: Recommender Systems in Requirements Engineering

Alexander Felfernig
Institute for Software Technology
Inffeldgasse 16b/2
Graz A-8010
afelfernig@ist.tugraz.at

Martin Stettinger
Institute for Software Technology
Inffeldgasse 16b/2
Graz A-8010
mstettinger@ist.tugraz.at

Andreas Falkner
Siemens AG Austria
Siemensstrasse 90
Vienna A-1210
andreas.a.falkner@siemens.com

Müslüm Atas
Institute for Software Technology
Inffeldgasse 16b/2
Graz A-8010
muatas@ist.tugraz.at

Xavier Franch
Department of Service and
Information System Engineering
Calle Jordi Girona, 1-3
Barcelona ESP-08034
franch@essi.upc.edu

Cristina Palomares
Department of Service and
Information System Engineering
Calle Jordi Girona, 1-3
Barcelona ESP-08034
cpalomares@essi.upc.edu

ABSTRACT

The major focus of OpenReq is the development of recommendation and decision technologies that efficiently support requirements engineering processes in large and distributed software projects. Example scenarios thereof are the bid management in industrial systems, requirements engineering in cross-plattform open source software development, and requirements management in large user communities (telecommunications sector). The aim of this paper is to provide an overview of OpenReq and to provide insights into related application scenarios and research issues.

CCS CONCEPTS

•**Information Storage and Retrieval** → *Information Search and Retrieval;* •**Information Systems Applications** → *Decision Support Systems;* •**Information Interfaces and Presentation** → User Interfaces;

KEYWORDS

Recommender Systems, Requirements Engineering

ACM Reference format:
Alexander Felfernig, Martin Stettinger, Andreas Falkner, Müslüm Atas, Xavier Franch, and Cristina Palomares. 2017. OpenReq: Recommender Systems in Requirements Engineering. In *Proceedings of 2nd Workshop on Recommender Systems and Big Data Analytics, Graz, Austria, Oct. 2017 (RS-BDA'17),* 4 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

High-quality requirements engineering (RE) is among the most critical factors for successful software projects [14]. The overall goal of OpenReq[1] is to develop intelligent recommendation and decision technologies that support different phases of requirements engineering [1, 3, 4, 10, 20]. Based on existing work [11, 20, 21], the project focuses on AI-based techniques that pro-actively support stakeholders within the scope of requirements engineering. OpenReq entails different industrial RE scenarios (trials) that range from the bid management (identification and feasibility estimation of requirements in large-scale industrial development projects), community-driven cross-platform requirements engineering, and requirements engineering scenarios in telecommunication-related large user communities.

The *bid management* scenario (see also Section 5) focuses on Request For Proposals (RFPs) management for railway safety systems. These RFPs are issued by national railway providers and comprise natural language documents of several hundred pages with requirements of different levels of detail and of different types (domain specific, physical, non-functional, references to standards and regulations, etc.). At present, most of this management is done manually using an RE system, making most of the tasks time consuming. OpenReq will help in this scenario mainly to: 1) automatically extract requirements from RFPs, differentiating the information included in these RFPs that are not requirements; 2) reuse technical decisions made in previous projects; 3) automatically assign requirements to stakeholders; and 4) support group decision making.

In the case of the *cross-platform open source* scenario, the community consists of individuals contributing in their free time and professional developers working on projects. Here, OpenReq will help with advanced *user engagement* (e.g., by identifying users who potentially could contribute to issues related to topics of interest for other users), with *internal release planning* (e.g., by detecting "urgent" requirements using community information such as numerous complaints about the same issue or by supporting group decision making by, for example, highlighting relevant stakeholders who should be included in a discussion), with the *management of requirements knowledge* (e.g., by detecting requirements dependencies or by supporting the assessment of feature compatibilities and their relationships), and with the *detection of new requests* based on discussions identified in community sites.

[1]Intelligent Recommendation and Decision Technologies for Community-Driven Requirements Engineering (Horizon 2020 Project, www.openreq.org).

The main goal of the *telecommunication scenario*, apart from improving the RE process, is to react as fast as possible to the opinions of customers (stated in user communities). With this in mind, OpenReq will be used in this scenario to: 1) *identify and extract requirements* from user requests; 2) monitor the communities to *identify acute issues* to enable early risk assessment; 3) *propose prioritization indicators* for requirements derived from user discussions and/or usage behavior; and 4) support stakeholders in the *preparation for a group decision* (e.g., highlighting relevant topics/artifacts and relevant stakeholders).

In addition to the three mentioned trial scenarios, the OpenReq ShowCase will be developed which will integrate the core features (recommendation and decision technologies) of the OpenReq software components. We will provide these features in terms of an open-source component (HTML-5 application) with the goal to enable quality improvements in RE processes on a large scale. This component will consist of functionalities that support the automated identification of requirements from different knowledge sources (e.g., communities or natural language text documents), the recommendation of requirements and stakeholders in different RE phases, the support of group decision making in release planning, and the automated identification of (hidden) dependencies between requirements. Especially dependencies detection and (formal) release planning strongly depend on each other since release plans have to take into account existing dependencies. The later such dependencies are detected the higher the corresponding follow-up costs in a software project.

Major recommendation paradigms that will be integrated into OpenReq are the the following. First, *collaborative filtering* based recommendation [17] simulates word-of-mouth promotion of items where the opinion of family members and friends (also denoted as *nearest neighbors*) has an impact on choices taken by a person. In the context of recommender systems, nearest neighbors are system users with similar preferences often expressed in terms of item evaluations. Second, *content-based filtering* [22] focuses on the analysis of item descriptions, for example, news items which are content-wise similar to those already "consumed" by a person are recommendation candidates. Third, *constraint-based recommendation* is based on explicitly defined recommendation knowledge often represented in terms of constraints or rules – see [9]. Finally, group recommender algorithms [19, 25] focus on recommending items to groups of users. In this context, basic recommendation technologies such as collaborative filtering and content-based filtering are combined with social choice functions [19] that help to aggregate the preferences of individual users (stakeholders).

In the following, we will provide an overview of RE-related activities that can be supported by recommendation and decision technologies. In Section 2 we show in which way recommender algorithms for single users (stakeholders) can be applied in RE contexts. Section 3 focuses on scenarios where groups of stakeholders have to be supported in their decision processes. Section 4 is related to issues in the context of identifying and managing dependencies between requirements and resolve inconsistencies. Thereafter, we present one of the application scenarios of OpenReq (Section 5). Finally, we conclude the paper with Section 6.

2 RECOMMENDATIONS FOR SINGLE USERS

All of the recommendation approaches mentioned so far can be applied in the context of single-user recommendation scenarios, i.e., scenarios were a single user (stakeholder) interacts with a recommender application.

An example of a RE-related scenario where *collaborative filtering* [17] can be applied is the following: when trying to understand a given set of requirements (a requirements model), collaborative recommendation can recommend requirements that have already been analyzed by stakeholders with similar interests, i.e., stakeholders who analyzed similar sets of requirements. Furthermore, collaborative filtering can be applied in the context of identifying discussion forums for stakeholders, i.e., depending on the interests of the current stakeholder, further forums / discussion topics can be identified that might be of relevance for the stakeholder [5–7].

When defining requirements, *content-based recommenders* [22] can recommend requirements that have already been defined in previous projects and could be of relevance for the current project, i.e., support requirements reuse. Similarly, content-based recommendation can be used to support the recommendation of stakeholders relevant for a new software project. Depending on the requirements defined for a software project, content-based recommendations can indicate persons who could be engaged as stakeholders in a software project due to their tasks already completed in previous projects. Finally, content-based recommendation technologies can be applied in the context of new requirements development, for example, to filter requirements of high relevance due to the fact that these cover issues/topics included in many discussion threads.

Constraint-based recommendation [9] is applied in scenarios where constraints are used to define restrictions on the possible outcomes of a decision process. An example of the application of constraint-based recommendation technologies in RE is *release planning* [24], i.e., the decision making on when (in which software release) to implement a specific requirement. In this scenario, requirements have to be assigned to releases. Existing dependencies between requirements have to be taken into account by the search component (e.g., constraint solver [26]) in charge of identifying/proposing solutions for a release planning task. A major issue in this context is to assure that all existing dependencies between requirements are taken into account. This requires the integration with corresponding dependency detection functionalities (see Section 4).

Integrations of basic recommendation approaches into requirements engineering environments already exist – for an overview see, for example, [11, 20, 21]. A major goal of OpenReq in this context is to focus on a more in-depth integration of these technologies with related social factors, for example, by taking into account decision styles of stakeholders [8] and exploiting recommenders for achieving persuasive effects to increase the amount of information exchange [18].

All the examples mentioned so far are related to single stakeholder (user) scenarios, i.e., scenarios where single stakeholders are in charge of taking a decision (e.g., which stakeholders should be invited to join the project or which requirements should be reused). In the following sections, we will focus on scenarios where groups of stakeholder have to be supported by recommendation technologies.

3 RECOMMENDATIONS FOR GROUPS

In contrast to single user scenarios, many choice tasks are defined in group contexts. One example thereof is the already mentioned release planning scenario. Very often, decisions regarding the assignment of requirements to releases are taken in groups, i.e., a group as a whole has to develop agreement regarding the planned releases. In such scenarios, inconsistencies between the preferences of individual stakeholders can occur. For example, two stakeholders have different opinions regarding the assignment of a specific requirement to a release. Another related example is *requirement evaluation* where a group of stakeholders is in charge of evaluating a requirement with regard to different dimensions such as *effort in MMs*, *risk level*, *potential turnover*, and *importance of implementation.* The group (of stakeholders) as a whole has to decide on how to further proceed with this requirement. Since different stakeholders often evaluate requirements differently, a major task in this context is to achieve consensus regarding the overall evaluation. In this context, diagnosis techniques [12, 23] play a major role since they are able to indicate possible (minimal) changes of the current stakeholder preferences in order to identify a solution (e.g., a release plan). In the context of group recommendation scenarios, *social choice functions* play a major role [19]. These functions (also denoted as *aggregation functions*) help to identify recommendations acceptable for the whole group. For example, *least misery* prefers recommendations that do not ignore negative evaluations of stakeholders, i.e., if a requirement has been evaluated negatively by ”only” one member of a group of stakeholders, this requirement has to be evaluated further (and discussed in the group) before being acceptable as a release candidate.

Integrations of group recommendation technologies into RE processes already exist [10, 21]. A major focus of OpenReq is to gain in-depth insights into RE related decision processes and how (group) recommendation technologies best help to improve the overall quality of decision processes. For example, different types of decision biases will be analyzed with regard to their occurrence and possibilities to counteract. An example thereof is *GroupThink* [15] in a discussion forum [7] where strong influences of opinion makers can result in suboptimal outcomes of related decision processes.

Furthermore, existing theories of group dynamics [13] and related social choice functions [19] are evaluated with regard to their applicability and new variants will be developed to optimize recommendation support in RE related group decisions. We are also working on extending the application of group recommendation technologies to scenarios where recommendations are used to persuade users to change their behavior, for example, in terms of increasing their personal engagement in requirements engineering processes a.o. in terms of an increased frequency of knowledge sharing (see, for example, [2]).

4 DEPENDENCY DETECTION AND CONFLICT MANAGEMENT

A major issue in group decision making is how to deal with conflicting preferences of group members. Related examples in RE are group decisions in the context of release planning where stakeholders have conflicting preferences (e.g., regarding the assignment of a requirement to a release) but also different perceptions regarding the meta-properties of a requirements (e.g., effort, risk, etc.). If available in an explicit fashion, for example, in terms of stakeholder-specific evaluations or explicit assignments of requirements to releases, related conflicts can be resolved on the basis of conflict detection and diagnosis algorithms [12, 16, 23]. However, a major issue in this context is also to identify (hidden) dependencies between requirements, i.e., relationships (constraints) between requirements that are not represented in an explicit fashion, i.e., they are not contained in the requirement model due to the fact that the constraints/restrictions are simply not known to stakeholders.

Approaches to the automated detection of dependencies between requirements already exist [10, 21]. Existing approaches focus on the detection of dependencies using content-based recommendation based on similarity measures. A major focus of OpenReq is to advance the state of the art in dependency detection and to come up with new solutions (e.g., based on techniques from natural language processing) that help to significantly increase the overall quality of dependency detection. For example, existing content-based approaches provide indications of dependencies in terms of similarities between requirements. OpenReq will go beyond that a.o. in terms of approaches that also point out semantic properties of dependencies.

5 SCENARIO: BID MANAGEMENT

From the three OpenReq trial scenarios, we decided to discuss the Siemens trial in more detail. The corresponding use-case (trial) in the OpenReq project involves bid projects for large-scale industrial systems related to the Siemens Mobility division.

RFPs (Request For Proposal) for railway safety systems are issued by different national railway providers and comprise natural language documents (represented in MS Word format) consisting of several hundred pages with requirements of various kind (domain specific, physical, non-functional, references to standards and regulations, etc.) and level of detail. Typically, a complete bid (proposal) comprises several subsystems, such as signaling hardware, track indication, interlocking software, ETCS, SCADA, etc.

Proposals are delivered by the national sales departments of large enterprises such as Siemens. Typically, a bid project with a duration of a few months is necessary to answer a RFP. The team comprises several stakeholders such as a project manager, a requirements administrator, a system architect, and technical experts.

Requirements engineering is an important part of the bid process. Its main purpose is to ensure the technical compliance of the offer. After importing the (unstructured) natural language text into requirements management tool such as Polarion ALM and cutting it up into requirement candidates, recommendation technologies can support a.o. the following sub-tasks:

- Deciding which of the candidates are real requirements and which are just explanatory text. This classification is based on domain knowledge and experience from past bid projects.
- Assigning the requirements to one or more stakeholders (internal departments, external subcontractors) who shall evaluate them. Recommendation can suggest the corresponding stakeholder roles (not physical persons).

- Evaluating the requirements for technical compliance (yes, conditionally, no). Recommendation is based on similar requirements.
- Suggesting solution approaches to satisfy the requirement and deciding which approach should be supported in the given context.

6 CONCLUSIONS

In this paper, we provided a short overview of scenarios where recommendation and decision technologies can support requirements engineering processes. Within the scope of OpenReq, we will focus on the development of corresponding technologies that can be used as an extension for existing requirements engineering tools but also as a basis for the development of a new generation of requirements engineering solutions that focus on a systematic improvement of related development, maintenance, quality assurance, and decision processes. Finally, a basic version of OpenReq will be provided that acts as a showcase for demonstrating the basic capabilities of the OpenReq components.

ACKNOWLEDGMENTS

The work presented in this paper has been conducted within the scope of the Horizon 2020 project OpenReq (Intelligent Recommendation and Decision Technologies for Community-Driven Requirements Engineering) which is supported by the European Union under the Grant Nr. 732463.

REFERENCES

[1] B. Alenljung and A. Persson. 2005. Decision-making activities in the requirements engineering decision processes: A case study. In *ISD 2005*. Karlstad, Sweden, 707–718.

[2] M. Atas, A. Felfernig, M. Stettinger, and TNT. Tran. 2017. Beyond Item Recommendation: Using Recommendations to Stimulate Knowledge Sharing in Group Decisions. In *9th International Conference on Social Informatics (SocInfo 2017)*. Oxford, UK, 1–10.

[3] A. Aurum and C. Wohlin. 2003. The fundamental nature of requirements engineering activities as a decision-making process. *Information and Software Technology* 45, 14 (2003), 945–954.

[4] R. Burke, A. Felfernig, and M. Goeker. 2011. Recommender Systems - An Overview. *AI Magazine* 32, 3 (2011), 13–18.

[5] C. Castro-Herrera, C. Duan, J. Cleland-Huang, and B. Mobasher. 2008. Using Data Mining and Recommender Systems to Facilitate Large-Scale, Open, and Inclusive Requirements Elicitation Processes. In *16th IEEE Intl. Conf. on Req. Engineering (RE'08)*. Barcelona, Spain, 165–168.

[6] J. Cleland-Huang, H. Dumitru, C. Duan, and C. Castro-Herrara. 2009. Automated support for managing feature requests in open forums. *Commun. ACM* (2009), 68–74.

[7] J. Cleland-Huang and B. Mobasher. 2008. Using data mining and recommender systems to scale up the requirements process. In *2nd International Workshop on Ultra-large-scale Software-intensive Systems*. Leipzig, Germany, 3–6.

[8] A. Felfernig, M. Atas, TNT. Tran, M. Stettinger, S. Polat-Erdeniz, and G. Leitner. 2017. An Analysis of Group Recommendation Heuristics for High- and Low-Involvement Items. In *30th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2017)*. Arras, France, 335–344.

[9] A. Felfernig and R. Burke. 2008. Constraint-based Recommender Systems: Technologies and Research Issues. In *ACM International Conference on Electronic Commerce (ICEC08)*. Innsbruck, Austria, 17–26.

[10] A. Felfernig, W. Maalej, M. Mandl, M. Schubert, and F. Ricci. 2010. Recommendation and Decision Technologies For Requirements Engineering. In *ICSE 2010 Workshop on Recommender Systems in Software Engineering*. Cape Town, South Africa, 1–5.

[11] A. Felfernig, G. Ninaus, H. Grabner, F. Reinfrank, L. Weninger, D. Pagano, and W. Maalej. 2013. *Managing Requirements Knowledge Book* (1st ed.). Springer, Berlin Heidelberg, Chapter An Overview of Recommender Systems in Requirements Engineering, 315–332.

[12] A. Felfernig, M. Schubert, and C. Zehentner. 2012. An Efficient Diagnosis Algorithm for Inconsistent Constraint Sets. *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AIEDAM)* 26, 1 (2012), 175–184.

[13] D. Forsyth. 2006. *Group Dynamics*. Thomson Higher Education.

[14] H. Hofmann and F. Lehner. 2001. Requirements Engineering as a Success Factor in Software Projects. *IEEE Software* 18, 4 (2001), 58–66.

[15] I. Janis. 1972. *Victims of Groupthink*. Houghton-Mifflin.

[16] U. Junker. 2004. QUICKXPLAIN: Preferred Explanations and Relaxations for Over-Constrained Problems. In *19th National Conference on AI (AAAI04)*. San Jose, CA, 167–172.

[17] J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Riedl. 1997. GroupLens: applying collaborative filtering to Usenet news Full text. *Comm. of the ACM* 40, 3 (1997), 77–87.

[18] M. Stettinger M. Atas, A. Felfernig and TNT. Tran. 2017. Beyond Item Recommendation: Using Recommendations to Stimulate Information Exchange in Group Decisions. In *9th International Conference on Social Informatics (SocInfo 2017)*. Oxford, UK, 1–10.

[19] J. Masthoff. 2011. Group Recommender Systems. *Recommender Systems Handbook* (2011), 677–702.

[20] B. Mobasher and J. Cleland-Huang. 2011. Recommender Systems in Requirements Engineering. *AI Magazine* 32, 3 (2011), 81–89.

[21] G. Ninaus, A. Felfernig, M. Stettinger, S. Reiterer, G. Leitner, L. Weninger, and W. Schanil. 2014. IntelliReq: Intelligent Techniques for Software Requirements Engineering. In *European Conference on Artificial Intelligence, Prestigious Applications of Intelligent Systems (PAIS)*. 1161–1166.

[22] M. Pazzani and D. Billsus. 1997. Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learning* 27 (1997), 313–331.

[23] R. Reiter. 1987. A theory of diagnosis from first principles. *AI Journal* 23, 1 (1987), 57–95.

[24] G. Ruhe and M. Saliu. 2005. The Art and Science of Software Release Planning. *IEEE Software* 22, 6 (2005), 47–53.

[25] M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, and M. Jeran. 2015. Counteracting Serial Position Effects in the CHOICLA Group Decision Support Environment. In *20th ACM Conference on Intelligent User Interfaces (IUI2015)*. Atlanta, Georgia, USA, 148–157.

[26] E. Tsang. 1993. *Foundations of Constraint Satisfaction*. Academic Press, London.