Poster: Answering the Requirements Traceability Questions

Arushi Gupta
Department of EECS
University of Cincinnati
Cincinnati, Ohio
gupta2ai@mail.uc.edu

Wentao Wang
Department of EECS
University of Cincinnati
Cincinnati, Ohio
wang2wt@mail.uc.edu

Nan Niu
Department of EECS
University of Cincinnati
Cincinnati, Ohio
nan.niu@uc.edu

Juha Savolainen
Software and Control R&D
Danfoss Drives A/S
Gråsten, Denmark
juhaerik.savolainen@danfoss.com

ABSTRACT

To understand requirements traceability in practice, we present a preliminary study of identifying questions from requirements repositories and examining their answering status. Investigating four open-source projects results in 733 requirements questions, among which 43% were answered successfully, 35% were answered unsuccessfully, and 22% were not answered at all. We evaluate the accuracy of using a state-of-the-art natural language processing tool to identify the requirements questions and illuminate automated ways to classify their answering status.

CCS CONCEPTS

• **Software and its engineering → Requirements analysis**; **Traceability**;

KEYWORDS

Traceability, requirements questions, answering status

ACM Reference Format:
Arushi Gupta, Wentao Wang, Nan Niu, and Juha Savolainen. 2018. Poster: Answering the Requirements Traceability Questions. In *ICSE'18 Companion: 40th International Conference on Software Engineering Companion, May 27-June 3, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3183440.3195049

1 INTRODUCTION

The *requirements traceability problem* refers to the inability to describe and follow the life of a requirement [2]. What specific questions do practitioners actually ask regarding the life of a requirement? How many are successfully answered? What makes the remaining questions difficult to answer successfully or unanswered at all? These illustrate the motivations behind our research.

Malviya and her colleagues recently conducted an online survey with requirements professionals in the IT industry [6]. Their analysis of 29 survey responses grouped a total of 159 natural language queries into 9 purpose categories. Building on the previous research, we carried out a study to identify the questions that software practitioners asked in the requirements repositories. Table 1 lists the four open-source projects of our study. These projects tackle problems in different domains where different concerns such as security [13] and dependability [16] are important. For each project, we extracted the data from its initial release to its latest stable release. We also balanced factors such as group size [1] and release cycle [8, 14] in choosing the projects of Table 1.

In order to identify requirements traceability questions, it is crucial to recognize where such questions are likely to be recorded. The literature suggests that issue trackers are essential for open-source projects to manage requirements [9]. Common to all the 4 projects of Table 1 is the use of JIRA for requirements management. We relied on the issue type defined in each of the 4 projects to recognize requirements. The widely adopted requirements types included "feature request" and "enhancement", whereas "epic" and "story" [4, 11] reflected these projects' agile nature.

Our identification of requirements question from the JIRA repositories was performed at the comment level by leveraging the Stanford CoreNLP toolkit [7]. We removed duplicates (e.g., a comment reply repeated the content that had already been posted before) and filtered out three kinds of irrelevant information: XML, URL, and source code. The comments containing any question recognized by the Stanford CoreNLP formed the result set Q.

2 RESULTS AND ANALYSIS

We evaluate the results Q by comparing them with the manually created answer sets of the 4 projects. These comparisons are summarized in Table 2. For each open-source project of our study, two researchers built their answer sets individually and reached a substantial degree of agreement: average Cohen's kappa=0.67 on requirements traceability questions over 4 projects. The discrepancies were resolved in a joint meeting between the researchers.

Recent work by Liu *et al.* [5] investigated supervised learning methods for automatically detecting questions in microblogs. Their experiment with 8,465 Sina microblogs (1,278 questions) showed that the combination of lexical, syntax, and contextual features

Table 1: Four open-source projects from which requirements questions are uncovered.

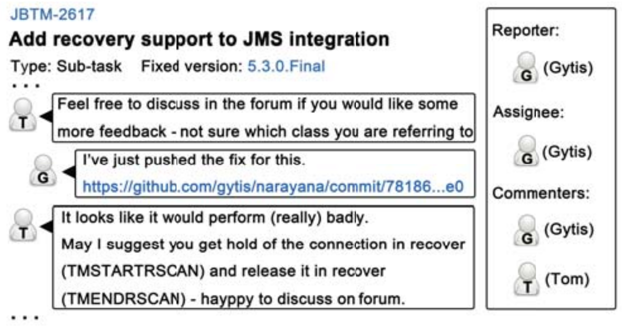
Project	Characteristics			Statistics			
	domain	written in	initial release	# of days per release	# of req.s	# of stakehold -ers per req.	# of code files per req.
AIRFLOW	workflow execution	Python	Oct 16, 2014	30.42	629	2.11	11.76
ANY23	RDF data extraction	Java	July 16, 2012	139.52	182	2.14	20.33
DROOLS	business rules	Java	Nov 13, 2012	29.36	486	1.78	31.26
JBTM	business process	Java, C++	Dec 5, 2005	21.23	1575	1.63	32.37

Table 2: Accuracy of requirements questions identification.

Project	|Answer Set|	$\|Q\|$	Recall	Precision	F_1
AIRFLOW	145	161	0.76	0.68	0.72
ANY23	98	93	0.69	0.73	0.71
DROOLS	96	112	0.81	0.70	0.75
JBTM	394	471	0.85	0.71	0.77

achieved the best performance: Recall=0.73, Precision=0.63, and F_1=0.68. Our results presented in Table 2 are comparable to this performance level, though our algorithm is unsupervised and mines questions from requirements repositories.

The following conversations extracted from [15] show a false negative (Tom's question beginning with "not sure ...") and a false positive (Tom's suggestion starting with "May I suggest ..."). Possible improvements include semantic parsing beyond the comment level and recognizing project-specific terms and concepts [10].


JBTM-2617

Add recovery support to JMS integration

Type: Sub-task Fixed version: 5.3.0.Final

...

T: Feel free to discuss in the forum if you would like some more feedback - not sure which class you are referring to

G: I've just pushed the fix for this. https://github.com/gytis/narayana/commit/78186...e0

T: It looks like it would perform (really) badly. May I suggest you get hold of the connection in recover (TMSTARTRSCAN) and release it in recover (TMENDRSCAN) - hayppy to discuss on forum.

...

Reporter: G (Gytis)

Assignee: G (Gytis)

Commenters: G (Gytis) T (Tom)

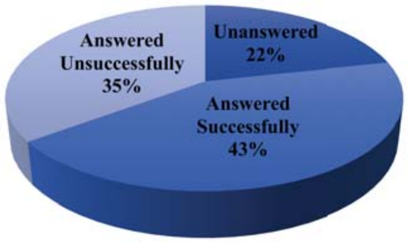


Figure 1: Classifying the 733 requirements questions.

3 SUMMARY

The answering status of the 733 requirements questions was analyzed manually and the classification results are shown in Figure 1. Based on our observations, automated methods of classifying the question's answering status could be developed according to features like how requirements traceability is used [12] and whether the diverse chunks of information are complementary [3].

Understanding practitioner questions helps to keep research grounded. We have presented our preliminary results of mining requirements repositories so as to identify questions and to classify whether and how well the questions are answered. Our ongoing work focuses on developing enhanced tools to further automation.

4 ACKNOWLEDGMENTS

The work is funded by the U.S. NSF Grant CCF-1350487.

REFERENCES

[1] T. Bhowmik, N. Niu, W. Wang, J.-R. C. Cheng, L. Li, and X. Cao. Optimal group size for software change tasks: a social information foraging perspective. *IEEE Transactions on Cybernetics*, 46(8):1784–1795, August 2016.

[2] O. Gotel and A. Finkelstein. An analysis of the requirements traceability problem. In *ICRE*, pages 94–101, Colorado Springs, CO, USA, April 1994.

[3] X. Jin, N. Niu, and M. Wagner. On the impact of social network information diversity on end-user programming productivity: a foraging-theoretic study. In *SSE*, pages 15–21, Seattle, WA, USA, November 2016.

[4] C. Khatwani, X. Jin, N. Niu, A. Koshoffer, L. Newman, and J. Savolainen. Advancing viewpoint merging in requirements engineering: a theoretical replication and explanatory study. *Requirements Engineering*, 22(3):317–338, September 2017.

[5] X. Liu, R. Xie, C. Lin, and L. Cao. Question microblog identification and answer recommendation. *Multimedia Systems*, 22(4):487–496, July 2016.

[6] S. Malviya, M. Vierhauser, J. Cleland-Huang, and S. Ghaisas. What questions do requirements engineers ask? In *RE*, pages 100–109, Lisbon, Portugal, September 2017.

[7] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *ACL*, pages 55–60, Baltimore, MD, USA, June 2014.

[8] N. Niu. RE in the age of continuous deployment. In *RE*, pages 568–569, Lisbon, Portugal, September 2017.

[9] N. Niu, T. Bhowmik, H. Liu, and Z. Niu. Traceability-enabled refactoring for managing just-in-time requirements. In *RE*, pages 133–142, Karlskrona, Sweden, August 2014.

[10] N. Niu and S. Easterbrook:. Extracting and modeling product line functional requirements. In *RE*, pages 155–164, Barcelona, Spain, September 2008.

[11] N. Niu, A. Koshoffer, L. Newman, C. Khatwani, C. Samarasinghe, and J. Savolainen. Advancing repeated research in requirements engineering: a theoretical replication of viewpoint merging. In *RE*, pages 186–195, Beijing, China, Sept 2016.

[12] N. Niu, W. Wang, and A. Gupta. Gray links in the use of requirements traceability. In *FSE*, pages 384–395, Seattle, WA, USA, November 2016.

[13] N. Niu, Y. Yu, B. González-Baixauli, N. A. Ernst, J. C. S. do Prado Leite, and J. Mylopoulos. Aspects across software life cycle: a goal-driven approach. *Transactions on Aspect-Oriented Software Development*, 6:83–110, 2009.

[14] J. Savolainen, N. Niu, T. Mikkonen, and T. Fogdal. Long-term product line sustainability with planned staged investments. *IEEE Software*, 30(6):63–69, November/December 2013.

[15] G. Trikleris. https://issues.jboss.org/browse/JBTM-2617.

[16] W. Wang, A. Gupta, N. Niu, L. D. Xu, J.-R. C. Cheng, and Z. Niu. Automatically tracing dependability requirements via term-based relevance feedback. *IEEE Transactions on Industrial Informatics*, 14(1):342–349, January 2018.