Towards Explainable RE Tools

Andreas Vogelsang*, Jonas Paul Winkler*, Henning Femmer†
* Technische Universität Berlin, DCAITI, Germany
{andreas.vogelsang,jonas.winkler}@tu-berlin.de
† Technische Universität München, Germany
femmer@in.tum.de

I. Motivation and Problem

Many Requirements Engineering tasks are nowadays supported by tools that either check the quality of manual RE work or perform RE tasks completely automatic. Examples are requirements categorization [1], prioritization [2], trace link recovery [3], or detection of language weaknesses [4]. The increasing abilities of these tools is driven by the availability and accessibility of complex technologies. RE tools make use of advanced natural language processing techniques [4], information retrieval mechanisms [3], and machine learning (e.g., by artificial neural nets [1]).

Despite the complex technologies used, RE tools are very appealing to practitioners because most of the technology is hidden from the user. However, when tools produce results that a user finds strange or that a user cannot explain, tools often fail to give evidence or hints *why* it made this decision and what the consequences are. Moreover, for some of the complex technologies used it may even be impossible to provide reasons for some decisions. For example, it is very hard to explain why a neural net makes a specific decision.

A special property of RE tools is that they are almost never used in a fully automated context. Most of the times, RE tools are part of processes, where they support a human analyst in performing tasks or reviewing work products. Therefore, we argue in this paper that more research is needed towards *explainable* RE tools. An explainable RE tool is able to provide rationales or indication for the decisions that it makes. Moreover, we argue that RE tools should also provide *actionable* results. A result is actionable if the tool can provide hints or recommendations on what could be done to improve or change a situation.

We use the following informal definitions of these terms:

Explainable The tool provides hints or indication on the rationale why the tool made a decision.

Actionable The tool provides hints or indication on how the user can influence the decision by changing the processed data.

In our experience, most tools and approaches reported in literature are not *explainable* or not *actionable*.

II. Example

One of the most cited papers of the RE conference is the paper on automated requirements tracing by Hayes et al. [3]. In the paper, the authors describe an approach for automatically finding trace links between high-level and low level requirements. In order to achieve this goal, the tool uses similarity analysis of textual requirements using information retrieval techniques. Given a requirement, the tool yields a set of other requirements that the tool identifies as "related" to the initial requirement. In their evaluation, they achieved a recall of 85% and a precision of 40%.

Let us imagine this tool "in action". A requirements analyst has to check whether and how a high-level requirement is realized in a complex system consisting of several components. Unfortunately, there is no tracing information between the high-level requirements and the low-level requirements in the component specifications. The analyst starts the mentioned tool with a requirement-under-analysis and gets a list of 10 low-level requirements. She may not be aware of it, but only 4 out of the 10 requirements are actually related to her initial requirement and 1 or 2 requirements that are actually related are not part of the list. In this paper, we do not want to discuss whether or not these numbers make the tool unusable (see [5], [6] for a discussion on that topic). We think that even imperfect tools can support a requirements engineer. We are more interested in the possibilities the tool provides for the user to comprehend the results. In that respect, the mentioned tool falls short. Let us assume that some of the requirements in the given list appear strange to the requirements analyst. She wonders: "Why is that requirement in the list related to my initial one?" The tool does not provide any answers to this question. It is not *explainable*. The requirement analyst may come to the conclusion that the tool is not working correctly because the results of the tool do not match her mental model.

In the meantime, the requirements analyst may have talked to a colleague who is working on one of the components of the system. The colleague remembers a requirement in her component specification that is actually related to the high-level requirement of the requirements analyst. The analyst, however, responds: "That's strange. Your requirement did not appear in the list of the tool." Both wonder why this is the case and what they could possibly do to make the requirements easier to be perceived as related. Again, the tool does not provide any guidance or hints for answering this question. The results are not *actionable*.

This example shows that RE tools have special requirements with respect to *explainability* and *actionability* because they are usually used in semi-automated settings, where a human has to comprehend and further process the results of the tools

Classes	requirement, information
requirement	the duration until the switch is recognized as hanging must be a configurable parameter .
information	the component conditionally drives an external fan . this fan is required for active ventilation of the headlight .

Fig. 1. Explanation of the tool decision

and because the results may be used to improve the current situation.

Other examples in RE are tools that evaluate the *readability* of requirements by calculating complex metrics, such as the Bog-Index[1]. As an analyst, you neither get feedback why the index is low for a requirement nor how you can improve the requirement.

III. Towards Explainable RE Tools

In the past, we made some efforts to make our RE tools explainable and actionable. Here, we provide two examples:

We have developed an automated approach to differentiate requirements from non-requirements (information) in requirements documents [1]. At one of our industry partners, it is the document author's task to manually label all elements of a requirements document as either *requirement* or *information*. Our approach uses an artificial neural net that is trained on a large set of well-labeled requirements documents. After the training, the neural net is able to classify text fragments as one of the two classes. We use this approach to check the quality of this classification in existing documents. To make the decisions of the tool explainable, we have developed a mechanism that traces back the decision through the neural net and highlights fragments in the initial text that influenced the tool to make its decision [7]. As shown in Fig. 1, it appears that the word "must" is a strong indicator for a requirement, whereas the word "required" is a strong indicator for an information. While the first is not very surprising, the latter could indicate that information elements often carry rationales (why something is *required*).

Our second example addresses an automated tool for finding language weaknesses in requirements—so-called requirements smells [4]. In our evaluation of the tool, practitioners considered the detection of requirements smells as valuable. One important aspect was that the tool does not only detect suspicious use of language in requirements but also provide explanations and recommendations how to improve the current situation. Therefore, instead of just issuing "This requirement is not testable", the tool outputs an explanation and suggestions how to improve the requirement: "Comparatives are hard to test. Use absolute values to ensure testability."

These two examples of explainable and actionable RE tools illustrate how acceptance in practice may be increased.

IV. Beyond Explaining: Insights through Tools

While we see increased acceptance of RE tools as the main benefit of an explainable and actionable focus, we also envision that research towards explainable RE tools may also increase our understanding on how we write and understand requirements. A good example is the use of neural networks. While it is hard to comprehend why a neural net makes specific decisions, recent research has shown that it is not impossible to analyze the inner structure of neural nets to get deeper insights into the characteristics of the learned instances. For example, Bacciu et al. [8] have used so called auto encoders to force a neural net to focus on the "essential" characteristics of jokes. By training the neural net behind the auto encoder on thousands of jokes given as texts, the inner structure of the neural net had to focus on those specifics in a text that are characteristic for jokes. By analyzing the inner structure, the authors were able to identify "regions" in which the net allocates specific text fragments that it classifies as the joke's punchline or "dirty" jokes. We think that it is an interesting area of research to apply unsupervised learning techniques to large sets of requirements to learn more about the characteristics of requirements by analyzing the inner structure of the resulting networks.

V. Conclusions

In this paper, we argue that more research and awareness is needed for *explainable* and *actionable* RE tools—tools that provide results together with explanations and suggestions for changes. We motivate this argument by the growing use of complex technology like natural language processing and artificial intelligence in RE tools. The performance and accessibility of these technologies has led to RE tools that are able to support RE tasks, which are usually conducted manually. However, many tools only focus on performing the task itself without giving reasons why the tool has made specific decisions or providing guidance how the situation could be changed to reach specific goals. We argue that explainable and actionable RE tools increase their acceptance in practice and facilitate their integration in a semi-automated RE process, where tools do not replace human activities but support them. Moreover, we argue that analyzing tooling decisions leads to deeper insights on how we write and understand requirements.

References

[1] J. Winkler and A. Vogelsang, "Automatic classification of requirements based on convolutional neural networks," in *AIRE*, 2016.

[2] A. Perini, A. Susi, and P. Avesani, "A machine learning approach to software requirements prioritization," *IEEE TSE*, vol. 39, no. 4, 2013.

[3] J. Hayes, A. Dekhtyar, and J. Osborne, "Improving requirements tracing via information retrieval," in *11th IEEE International Requirements Engineering Conference (RE)*, 2003.

[4] H. Femmer, D. Méndez Fernández, S. Wagner, and S. Eder, "Rapid quality assurance with requirements smells," *JSS*, 2016.

[5] D. Berry, R. Gacitua, P. Sawyer, and S. F. Tjong, "The case for dumb requirements engineering tools," in *REFSQ*, 2012.

[6] D. Berry, "Evaluation of tools for hairy requirements and software engineering tasks," in *EmpiRE*, 2017.

[7] J. P. Winkler and A. Vogelsang, "What does my classifier learn? A visual approach to understanding natural language text classifiers," in *22nd International Conference on Natural Language & Information Systems (NLDB)*, 2017.

[8] D. Bacciu, V. Gervasi, and G. Prencipe, "LOL: An investigation into cybernetic humor, or: Can machines laugh?" in *8th International Conference on Fun with Algorithms (FUN)*, 2016.

[1] http://www.stylewriter-usa.com/stylewriter-editing-readability.php