Improving Traceability of Requirements Through Qualitative Data Analysis

Andreas Kaufmann, Dirk Riehle

Open Source Research Group, Computer Science Department
Friedrich-Alexander University Erlangen Nürnberg
Martenstr. 3, 91058 Erlangen, Germany
andreas.kaufmann@fau.de, dirk@riehle.org

Abstract: Traceability is an important quality aspect in modern software development. It facilitates the documentation of decisions and helps identifying conflicts regarding the conformity of one artifact to another. We propose a new approach to requirements engineering that utilizes qualitative research methods, which have been well established in the domain of social science. Our approach integrates traceability between the original documentation and the requirements specification and the domain model and glossary and supports adaptability to change.

1 Introduction

In this article we propose an approach to requirements engineering (RE) that utilizes methods that have been established in the field of social sciences. Our approach inherently provides explicit traceability between requirements specifications and the documents from which they were derived as well as between the requirements documentation and domain models. Existing tools for RE support the documentation of traceability through traceability matrices, offering traces back to a specific textual requirement and in the best case they support traceability to an initial artifact, however this documentation is a laborious task that has no additional value for the analysis of the initial artifacts. Most research on traceability for requirements engineering also focuses on "after the fact" requirements tracing [HD06]. By utilizing qualitative research methods to analyze the transcribed interviews we forego the need to extract pre-requirements specification (pre-RS) traces, be it manual or using natural language processing (NLP), after the specification has been finalized. This is achieved by documenting traces as a byproduct of the materials analysis, without it being an additional documentation step, but rather integrated into the method.

While ample research exists on how stakeholders should be approached through workshops, interviews, questionnaires etc. as well as on the way of documenting the elicited requirements [CA07], the actual process of analysis is a rarely documented process. On the other hand within the domain of social science research the analysis of text frequently follows well established methodologies for qualitative data analysis (QDA). In our approach we equate the 'theory building' of social science research with the creation of requirements documents, glossaries, and domain models.

2 Qualitative Data Analysis

Qualitative research seeks to describe, understand and even predict human behavior. It is a particularly well suited method to investigate the reasoning of participants and trying to understand *what*, *why* and *how* they think, act or experience a phenomenon. The main application of this kind of research is typically sociology.

The research question for qualitative research is generally decidedly different from quantitative research. While quantitative studies usually aim to answer very specific questions by means of statistical analysis of large test-data (hypothesis-testing), qualitative research is usually much more open. The goal is to build a theory, that emerges and will be refined during the research process.

In qualitative research, after an area of interest is identified the researcher has to choose a method for gathering data. The most common approach is to use in-depth interviews. After this initial gathering of data, a set of common themes, called the theory, is built through an analysis of the gathered data. The process of concurrent gathering and analysis of data is repeated iteratively until the gathering of new data no longer changes the resulting theory.

A vital step in the analysis process is called coding. A code is a conceptual label, linked to a particular pattern of thought, action or behavior. Strauss claims that "*The excellence of the research rests in large part on the excellence of the coding"* [St87]. A high quality coding should feature codes that do not remain a mere description of the labeled pattern. To achieve this distinction it is necessary to provide context to each code. Within the current state of the art of qualitative research this context is usually only provided informally through natural language text memos.

We would like to extend QDA methods by adding a formal meta model to which the coding should conform, so we can built a formal domain model on top of this data more easily.

In its effort of generating a deep understanding of reality from a textual description theory building bears a striking similarity to requirements engineering, which is why we aim to adopt some of its methods, specifically the gathering and coding text data.

3 Related Work

3.1 Application of QDA Methodologies to Requirements Engineering

The application of qualitative research methods to the elicitation of software requirements has not been investigated much, with a few exceptions that validate our vision and show the feasibility of this task.

Onabajo used a qualitative study to identify key concepts within a meta model for confidentiality requirements [On09]. However the application of qualitative methods are not the central topic in his dissertation and thus only described insufficiently and not developed as a general RE methodology.

Stein et al. used template analysis, a qualitative method similar to grounded theory, as a background reading technique for requirements elicitation [St09]. They found, that using this method gave them sufficient insight into the domain to be modeled, of which they previously had little to no domain knowledge. However they state, that their method of translating the findings into requirements through competence questions was inadequate. Later Zapata et al. also used template analysis for background reading [Za12].

3.2 Domain Modeling within Requirements Engineering

Most recent approaches to generate a formal representation of requirements explore ways of mapping textual requirements to a more formal notation. A common scenario that most of these methods handle unsatisfactorily is the adaptability when changes in the textual requirements occur after the model has been created. Landhäußer et al. addressed this problem with the vision of fully automated round-trip tool support integrating NLP and ontologies [La14]. Previous approaches to translate textual requirements to formal documents by Overmyer et al. [Ov01] and Kroha [Kr00] also recognized the iterative nature of requirements engineering. Another approach to establish traceability between models and text has been proposed by Cerbah & Euzenat [CE01], who used common terminology knowledge to link model elements to texts.

None of these methods however offer traceability beyond the textual requirements, or offer new ways to elicit these. Our proposed approach is incremental by design and provides transparency how a specific requirement is justified through an explicit statement by a stakeholder, an interpretation of such, or a compliance requirement.

Kaiya & Saeki propose a method using domain ontologies for requirements elicitation [KS06]. An advantage of this approach is that formalized domain knowledge represented in an ontology can easily be re-used and refined in later projects within the same domain. Also, ontologies for various domains can be found in libraries.

4 Description of the Approach

Our proposed method consists of two stages, which are detailed in the sections 4.1 and 4.2. Analogous to the original QDA process as a first step, the data collection and analysis through coding will be repeated iteratively until we reach theoretical saturation and the code system is of sufficient quality. Subsequently the code system has to be translated into a domain model, a glossary, and a structured requirements specification.

4.1 Data Collection and Analysis

For data collection we propose conducting interviews with the stakeholders. These interviews have to be transcribed for further analysis. The analysis of these transcriptions through coding can be supported by computer-assisted qualitative data analysis software (CAQDAS). These tools offer a structured way of coding the document, with different views for the documents, the codesystem and memos describing the theory.

To better support the application to requirements engineering we aim to develop our own software which will also support the coding process, but allows the user to develop a more structured code system that conforms to a specific meta model defining the relations between entities and distinguishing between different types of codes.

The coding process itself, similarly to traditional grounded theory approach, is structured into three phases which are repeated iteratively: (1) *open coding*, (2) *axial coding*, (3) *selective coding*.

During the open coding phase concepts that appear relevant within the text are labeled using a CAQDAS package. During open coding there is no structure within the emerging set of codes, the code system. Codes can either be in-vivo, meaning explicitly mentioned or slight abstractions from those.

In the second step, the axial coding a hierarchy is constructed within the code system spanning a wide range of abstraction levels in a tree structure.

During the selective coding phase the emerging theory is then consolidated, the relevance of each code is gauged resulting in core categories emerging, and codes are related to one another.

Although the application of natural language processing (NLP) is not the central argument of this paper, our preliminary results investigating the possible use of NLP to assist the QDA process using a machine learning algorithm warrant further research in this direction, to assist with the problem of large text corpora to be processed during the requirements elicitation process [Ka14]. Using only generic keyword extraction tools the complex task of coding could be automated with precision and recall of around 50%. While these results are far from satisfactory for complete automation a customized algorithm favoring precision may be used for suggesting possibly missed concepts.

4.2 Translation to a Domain Model and Requirements Specification

The translation of the code system to a formal domain model is supported by a more formalized code system that conforms to a specific meta model, compared to traditional QDA. Because of this, some of the relationships have already been modeled in the previous steps.

The formulation of a requirements specification may draw on the domain model and the restriction on the model elements that were coded in the code system. Further, this

process may also utilize the natural language transcriptions that are linked to specific entities to better understand how they are integrated into the system. Also, knowledge that is considered common domain knowledge may not have been touched in the interviews and will have to be identified by the analyst to complete the domain model.

4.3 Impact on Traceability

In Figure 1 we present an overview of the fine grain traceability between natural language requirements specification and domain models which we expect to offer through our approach. To exemplify the traceability we used a section from a requirements document written in natural language describing the user-management component of a software forge, and how roles and access rights are interconnected. This natural language description is then coded and transformed into a UML model. All three artifacts in the different stages of our process are color-coded to illustrate the traceability. The same process may be applied for interview transcripts used for requirements elicitation.

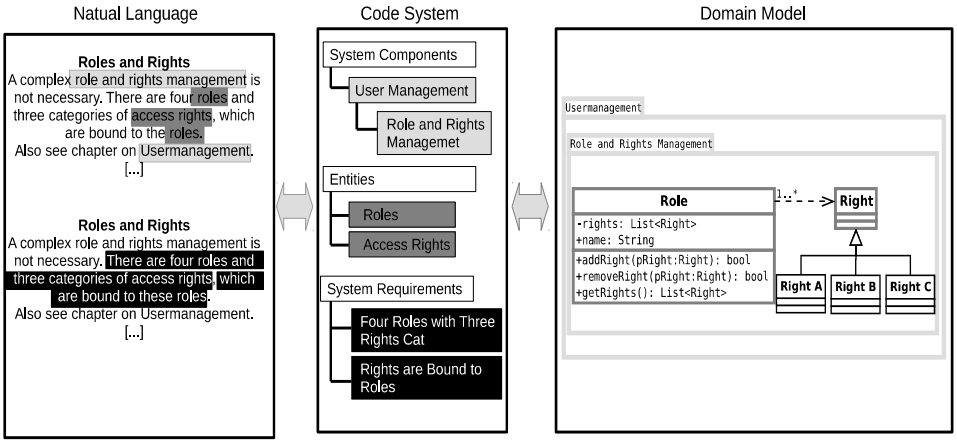

Figure 1: Traceability through the Artifacts

This also illustrates the adaptability to change. If the textual description of how many rights categories have to be implemented changes we can easily identify the corresponding model elements that will have to be adapted to integrate the change. This adaptability works both ways, so if a model is later changed for design or implementation reasons, it is easy to check if the resulting system still conforms to the intention of the stakeholders, and if not, who will be affected by this change.

5 Limitations

Our approach is limited to elicitation methods where the results can be transcribed for further analysis of the textual artifacts. Further, the rigorous application of our methods requires a non-trivial effort, which may not be justifiable for smaller projects.

Another problem is the lack of functionality in tool support for QDA which is currently only geared towards researchers of the social sciences. We therefore aim to develop our own software solution, that supports qualitative research methods and is specifically designed for requirements engineering within the software development process.

In the early stages of our project we focus on quality over speed and efficiency. This aspect will have to be evaluated at a later stage.

6 Conclusion

We proposed a new approach for requirements engineering that is applicable to any software engineering process using natural language documents and transcriptions. We suggest that the process of requirements engineering is similar to the theory building in social science, and want to transfer methods developed for qualitative data analysis to the domain of requirements engineering. Through this, we expect to improve the traceability and adaptability to change in the early stages of software development.

Literaturverzeichnis

[CE01] Cerbah, F.; Euzenat, J.: Traceability between models and texts through terminology. Data & Knowledge Engineering, 38(1), 31-43. 2001.

[CA07] Cheng, B. H.; Atlee, J. M.: Research directions in requirements engineering. In *2007 Future of Software Engineering*. IEEE Computer Society. 2007. pp. 285-303.

[HD06] Hayes, J. H. et al.: Advancing candidate link generation for requirements tracing: The study of methods. *Software Engineering, IEEE Transactions on*, *32*(1). 2006. pp. 4-19.

[Ka14] Kaufmann, A. Using Natural Language Processing to Support Interview Analysis. 2014

[KS06] Kaiya, H.; Saeki, M. Using domain ontology as domain knowledge for requirements elicitation. In Requirements Engineering, 14th IEEE International Conference (pp. 189-198). IEEE. 2006.

[Kr00] Kroha, P.: Preprocessing of requirements specification. In Database and expert systems applications (pp. 675-684). Springer Berlin Heidelberg. 2000.

[La14] Landhäußer, M. Et al.. From requirements to UML models and back: how automatic processing of text can support requirements engineering. Software Quality Journal, 22(1). 2014; pp 121-149.

[On09] Onabajo, A.: Analysis of multilateral software confidentiality requirements (Doctoral dissertation). 2009.

[Ov01] Overmyer, S. et al.: Conceptual modeling through linguistic analysis using LIDA. In Proceedings of the 23rd international conference on Software engineering (pp. 401-410). IEEE Computer Society. 2001.

[St09] Stein, S. Et al. Using Template Analysis as Background Reading Technique for Requirements Elicitation. In Software Engineering. 2009 (pp. 127-138).

[St87] Strauss, A.L. *Qualitative analysis for social scientists*. Cambridge [Cambridgeshire]: Cambridge University Press. 1987. p.27

[Za12] Zapata, J. Et al. (2012, October). An approach for using procedure manuals as a source for Requirements Elicitation. In Informatica (CLEI), 2012 XXXVIII Conferencia Latinoamericana En, 2012 ; pp. 1-8.