



A validation of QDAcity-RE for domain modeling using qualitative data analysis

Andreas Kaufmann¹ · Julia Krause¹ · Nikolay Harutyunyan¹ · Ann Barcomb^{1,2} · Dirk Riehle¹

Received: 23 September 2020 / Accepted: 29 July 2021
© The Author(s) 2021

Abstract

Using qualitative data analysis (QDA) to perform domain analysis and modeling has shown great promise. Yet, the evaluation of such approaches has been limited to single-case case studies. While these exploratory cases are valuable for an initial assessment, the evaluation of the efficacy of QDA to solve the suggested problems is restricted by the common single-case case study research design. Using our own method, called QDAcity-RE, as the example, we present an in-depth empirical evaluation of employing qualitative data analysis for domain modeling using a controlled experiment design. Our controlled experiment shows that the QDA-based method leads to a deeper and richer set of domain concepts discovered from the data, while also being more time efficient than the control group using a comparable non-QDA-based method with the same level of traceability.

Keywords Domain model · Domain modeling · Qualitative data analysis · Requirements engineering · Controlled experiment

1 Introduction

A domain model helps establish a common understanding of a domain among all stakeholders of a project. The development of such a model helps the analyst gain a deeper understanding of the involved entities and their interactions. It is also helpful for facilitating communication between stakeholders, and it forms a base for building design- and

implementation-oriented models and artifacts. However, the process of eliciting and documenting knowledge from domain experts remains in large parts dependent on the experience of the analyst. Consequently, the consistency and completeness of the domain model and the requirements, which affect implementation [9], can vary.

Traceability can help improve both of these criteria. By linking elements in different artifacts with one another, changes can be consistently applied to multiple artifacts, and through rigorous forward tracing, it can be assured that no part of any artifact is accidentally disregarded.

Requirements traceability in general “refers to the ability to describe and follow the life of a requirement, in both a forwards and backwards direction” [23]. Traces are frequently only documented from the requirements specification (RS) to development artifacts, the so-called post-RS traceability. Because the requirements elicitation process relies heavily on implicit knowledge on the part of the analyst, pre-RS traceability, meaning tracing between requirements and their origin, is a significant problem [23, 47]. Pre-RS traceability supports transparency and thus future decision-making. Therefore, it is highly desirable, but the standard solution employed today—keeping separate documentation which must be continually updated—is burdensome and error-prone. With an integrated method of ensuring pre-RS

✉ Andreas Kaufmann
andreas.kaufmann@fau.de

Julia Krause
julia.krause@fau.de

Nikolay Harutyunyan
nikolay.harutyunyan@fau.de

Ann Barcomb
ann@barcomb.org

Dirk Riehle
dirk@riehle.org

¹ Department of Computer Science, Friedrich-Alexander-University Erlangen Nürnberg, Martensstr. 3, 91058 Erlangen, Germany

² Electrical and Software Engineering Schulich School of Engineering, University of Calgary, 2500 University Drive NW, Calgary, AB T2N 1N4, Canada

traceability, consistent and complete trace documentation is ensured, and the extra step of after-the-fact documentation becomes obsolete. Our method documents traces between the domain model elements and their origin as a by-product of the analysis process. This constitutes a first step toward solving the pre-RS traceability challenge.

We previously published a method called *QDAcity-RE*, based on qualitative data analysis (QDA), borrowing and adapting methods from qualitative research methods to codify the analysis process, guide the analyst, and better document the process of domain analysis [30]. QDA techniques like they are applied in our method, and related work are based on the practice of *coding* data, where the term coding describes the concept of labeling parts of data.

The main benefits of our method are the following:

- Increased completeness and consistency of the resulting artifacts.
- Improvement of the analysis process by definition of a process where previously analysts had to rely on intuition and experience.
- Support pre-RS traceability between a domain model and their source.

We focus on completeness and consistency, since a formally correct specification can be defined as one that satisfies both these criteria [51]. Completeness may be regarded in terms of *external completeness*, and *internal completeness*. *External completeness* requires “that all of the information required for problem definition is found within the specification” [51], whereas *internal completeness* requires the absence of undefined items [51]. *Consistency* requires the absence of conflicts between elements within the artifact (*internal consistency* [4] or *intra-model consistency* [29]) or between different artifacts (*external consistency* [4] or *inter-model consistency* [29]).

From the benefits that were evidenced in our exploratory studies, follow the research questions outlined below, which guided the research design of the work presented in this article.

RQ1 Does the QDAcity-RE method of QDA-based domain modeling improve the consistency and completeness of domain models, compared to another method of domain modeling with the same level of pre-RS traceability?

RQ2 How do the traces between interview data and an intermediate artifact assist the analysis?

Keeping separate documentation of traces consistent with changes in all artifacts is a common problem. We expect that by embedding the process of trace generation within the analysis process, thus making trace generation automatic,

a major problem of pre-RS traceability is solved. Thus, a major hurdle for acceptance of such documentation is lowered.

In this article, we present a controlled experiment to validate the claims of our method. The QDAcity-RE method prescribes practices and processes for stakeholder sampling, data gathering, and data analysis, in addition to producing artifacts and results associated with process documentation, stakeholder materials, unified model, and target artifacts [30]. We evaluate the data analysis phase of the method through examination of the domain model artifacts and analysis of the study participants’ experiences. We compare QDAcity-RE, used in the QDAcity group, against a different method of analysis for domain modeling used by the control.

The method employed by the control group is based on *strategic reading*, a reading method developed in the field of education, which was adapted by Diaz [14] for use in Design Science Research (DSR) and domain modeling. The reason for choosing this method as a control is that it features a similar level of pre-RS traceability between the elements of the domain model and the underlying (source) text on which the analysis is based. The trace links between source material and model are fully tool supported [14], as is the case with QDAcity-RE. Since both methods were new to all participants, we could also evaluate the learning curve and the ease of use for untrained analysts.

The contributions to this article are the experimental evaluation of the QDAcity-RE method [30] and a comparative analysis with a domain analysis process proposed by Diaz et al. [14].

The rest of this article is structured as follows. First, we present related work in Sect. 2. In Sect. 3, we then present the experiment design. The evaluation of the artifacts and the qualitative evaluation of the participant’s experience is described in Sects. 4 and 5, respectively. Finally, the results and the limitations of our work are discussed in Sects. 6 and 7. A conclusion is drawn in Sect. 8.

2 Related work

The QDAcity-RE method uses QDA and principles of qualitative research. These principles have previously been employed in the context of requirements engineering (RE) with a similar motivation to the development of the QDAcity-RE method [6, 7, 38, 50]. Most of this work utilizes QDA in the larger framework of grounded theory (GT) [22]. GT is an approach to theory building that provides a methodological framework next to the analysis practices. Depending on the epistemological stance, different variations exist, whether it is positivistic or constructivistic.

QDA, and in particular GT, has been used to develop models [5–7, 26, 27, 50], requirements [8, 10, 40, 45],

personas [17, 18, 34], and architectures [3, 21]. QDAcity-RE is also focused on producing models. Within the scope of this article, we are specifically evaluating its utility in creating conceptual domain models.

Carvalho et al. compared a descriptive process model created by a researcher experienced with QDA using the GT-based method to a model created by an experienced software engineer [6]. While the experience of the software engineer could not be compensated, some benefits of the systematic way of exploring the data were demonstrated.

Similar findings are described by Pidgeon et al., for applying GT to knowledge elicitation [38]. They add that GT secures the traceability of a derived model back to the original data sources through the documentation of the analysis process via codes and memos. However, they remind the reader that models which are produced still have subjective elements and require validation.

Both Carvalho et al. and Pidgeon et al. criticize the complex and labor intensive analysis process of GT. Their findings can be transferred to the process of domain analysis, which also includes eliciting knowledge from domain experts and analyzing it to derive an abstract model [5]. Our studies concur with previous work concerning the issue of the required effort for performing methods like GT with the purpose of domain modeling [30]. Many of the problems we encountered in this regard, however, can be solved through better tool support tailored toward domain modeling. The tool support given in the context of the study presented in this article is comparably high. Still, problems related to tool support were one of the categories more frequently coded in our analysis of the participant interviews.

The use of GT to model requirements is also investigated in Halaweh's studies [26, 27]. Halaweh states that categories and their relationships derived from Corbin and Strauss' coding paradigm [12] can be compared to classes and their relationships in class diagrams. Thus, the informal model resulting from GT can be translated into a semi-formal model such as a UML class diagram. Theoretical sampling can help identify users for interviewing, and theoretical saturation can be used as an indicator to stop requirements elicitation. Theoretical sampling requires the sampling to be driven iteratively by current gaps in the theory instead of defined a-priori, and theoretical saturation describes the state where additional gathering of data would no longer induce significant changes to the theory. Halaweh argues that by applying GT and thereby letting requirements emerge from the data, requirements are user-driven, supporting user-centered design and satisfying user needs effectively.

Halaweh points out that the analyst needs to apply theoretical sensitivity in order to produce relevant results. Another claim of his studies is that GT is particularly suited to identify non-technical aspects regarding change due to the system's development and implementation, for example

user's resistance to change. This might help initiate proactive measures for implementation and training to overcome organizational problems. Halaweh conducted a case study and analyzed interviews which he used to develop a class diagram. Although he stated that he found equivalent elements between GT and object oriented analysis and design (OOAD), he did not provide guidelines for GT coding and for transferring the informal model to a class diagram. In contrast, we developed a concrete mapping from a code system classified with our code system language (CSL) to UML class diagrams.

While we developed an explicit mapping, and others have also used GT as a means to produce requirements artifacts directly through QDA or GT, Chitchyan employed a strategy for requirements elicitation that uses GT as a precursor to the actual definition of requirements [11]. In this process, GT would serve to define "contexts and preferences which serves as the foundation for the theorised requirements" [11]. From this, understanding of the domain candidates for requirements would then arise. We agree that the in-depth understanding of a domain is the foundation for a good specification. However, we also believe that a visual semi-formal representation of a conceptual model is an adequate representation of one perspective on this knowledge, which is why in our method, it is explicated as a specific view on the code system visualized through a UML class diagram with explicit mapping between this representation and the code system.

Similar to the idea put forth by Hallaweh, the use of GT to elicit social aspects and tacit knowledge has also been suggested by Chitchyan and Bird [10], who evaluated this aspect in a single-case case study. In a similar vein of uncovering tacit knowledge, but more specifically in the area of security requirements, Rashid et al. also investigated a GT approach [40]. Rashid et al. particularly point out benefits to the degree of freedom in the analysis using GT, and its ability to connect knowledge across security incidents in a multi-incident [40]. Rashid et al. state that their GT-based method was created in the pursuit to "[...] find an effective way to deal with the implicitness of domain knowledge [...]". We concur with this sentiment in our method for conceptual domain modeling. In the method presented by Rashid et al., as with most proposed approaches based on GT, relationships between concepts are documented during memo writing. We believe, that using semi-formal notation as we did with QDAcity-RE would be beneficial in these types of approaches as well. A model of security requirements with focus on the socio-technical aspects of a system was also developed by Fléchaïs using GT [19].

The social and political factors that may influence stakeholder's requirement was also addressed by Thew and Sutcliffe with their own method [44]. The method proposed by Thew and Sutcliffe is more structured than any GT-based

approach or our own method. They also recommend transcription and annotation of interviews during the elicitation process. However, this recommendation is limited to novice analysts, with experts foregoing this step. Further, the required annotation is more focused and structured by a fixed taxonomy on stakeholder's values, motivations and emotions [44].

Würfel, Lutz and Diehl propose a holistic approach for data elicitation, data analysis and the determination of requirements, similarly to the approach that is proposed by us [50]. They define two process phases: In the first phase, GT practices are employed for data elicitation and analysis. The second phase aims at transferring domain descriptions into use cases.

Hughes and Wood-Harper express the need for addressing the organizational context during requirements determination. They demonstrate the use of GT to develop an abstract account of the organization with two case studies [28]. They adapt GT by using pre-defined categories to address time constraints. The requirements determined in the case studies cover mostly organizational aspects. Examples of such aspects are high-level goals, constraints and aspects of change. However, the studies did not show how to extract specific requirements on a lower abstraction level or structural elements of an organization. The publications on these studies also do not describe the data analysis process in their case studies in detail.

Chakraborty and Dehlinger explain how the coding procedure of GT can be applied to determine enterprise system requirements and to derive UML diagrams, thus bridging the gap between qualitative data and final system descriptions [7]. They demonstrate their approach by deriving a UML class diagram from a textual high-level description of a university support system. However, the developed diagram is not consistent. Features and information about the implementation are represented as classes and the relationships between classes are not specified. An important adaptation in their procedure was the addition of conjectural categories to their model, which were not derived from the data but based on the experience of the analysts. They discovered that, apart from the advantage of traceability, the iterative process of GT allowed the analyst to discover and close information gaps earlier in the process.

Based on their previous work [7], Chakraborty et al. proposed a procedure called *Grounded and Linguistic-Based Requirements Analysis* for eliciting non-functional requirements (NFR) [8]. They argue that the application of GT-based practices in the analysis process improves the requirements specification by facilitating the sense making of multiple viewpoints into a cohesive description. However, Chakraborty et al. also point out that the differences between RE and theory development make adaptations to GT necessary. Also, because system analysts are not familiar with GT,

Chakraborty et al. propose to support the analyst in developing theoretical sensitivity and identifying the important concepts by giving him or her guidance about the theoretical principles to apply.

Chakraborty et al. used predefined categories of NFR. These categories were related using Mylopoulos, Chung, and Nixon's NFR framework [35]. Thomas, Bandara, Price, and Nuseibeh also use an analytical framework, including predefined thematic codes and extraction rules, to use QDA for the determination of privacy requirements for mobile applications [45]. They state that QDA improves requirements elicitation by accounting for contextual factors and securing traceability. While the use of an established taxonomy for NFRs is common in RE, Sharma and Biswas criticize the lack of such a categorization for functional requirements. They used a GT-based approach to develop a classification system based on the coding on RS documents [41].

One adaptation of GT that many of the presented articles propose is the use of pre-defined categories. This alleviates the high amount of effort required for a systematic analysis using GT. Traditional GT would not allow for such a-priori constructs or would at least defer their use to the end of a study to make sure that the theory development is not biased by preconceived notions of the researcher. In our research, we found that besides the obvious impact on resources, the usefulness of pre-defined categories is highly dependent not only on the domain but even more so on the desired artifact the analyst wishes to create at the conclusion of the analysis. For instance when the derivation of natural language requirements from the data were desired, this proved immensely helpful, while for the conceptual model, we found it more helpful to start without preconceptions. In the study presented here, all categories emerged from the participant's analysis, although the dimensions to explore were given by means of the code system language and the coding paradigm. The latter turned out to be a main point of confusion for some of the participants.

A contribution that distinguishes the research presented in this article from related work is a direct comparison of two different methods in the methodological setting of a controlled experiment. Most approaches presented in this section were either the presentation of an idea showcased with an example or studied in the setting of a single-case case study.

3 Experiment design

To validate our claim that using the QDAcity-RE method improves the consistency, completeness, and traceability of domain models, and to comparatively evaluate its utility for

the analyst to gain deep insight into a domain, we conducted a controlled experiment.

To identify potential limitations of our experiment design and apply corrective measures where possible, we performed multiple peer debriefing sessions [24, 43] throughout the planning and execution of the study. In separate instances, we debriefed colleagues on the fit of the overall experiment design for the research question, the sampling strategy, the design of the interview outline, the coding of the interview transcripts, and the strategies for investigator triangulation.

We divided our population into two groups, using a between-subject design. One group was tasked with analyzing the data using the analysis approach of our method, hereinafter referred to as QDAcity group. The other group was tasked to perform an analysis of the same data using a method utilizing *strategic reading* [14, 15], hereinafter referred to as control group. The participant sampling is described in more detail in Sect. 3.3.

Both groups were given the same task of creating a conceptual domain model in UML on the basis of six already transcribed semi-structured interviews with domain experts. The domain that was modeled within this study was human resource (HR) development. We describe the target domain briefly in Sect. 3.1.

Since the population of our experiment groups was small (nine participants), we cannot make a strong claim of generalization. Instead, in addition to our measurement instruments for the evaluation of the quality of the created artifacts, we also gathered qualitative insights from the study participants through interviews.

Our study consequently considers the following variables:

Independent variable:

- Analysis method (Sect. 3.2)

Dependent variables:

- Set of domain concepts and relationships
- The difference in time the analysis took

Controlled variables:

- Data to be analyzed
- Demographic characteristics of the participants
 - Age
 - Gender
 - Education
 - Prior experience with QDA and domain modeling

Within this context, completeness was operationalized in terms of information retrieval metrics with regard to a model of the domain previously validated through domain

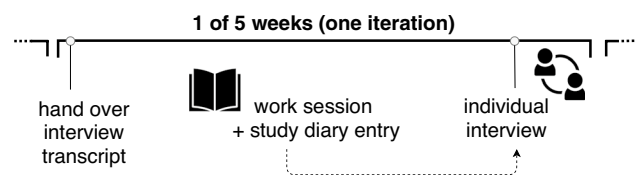


Fig. 1 Overview schedule of each iteration

experts. Intra-model consistency was analyzed with regard to conflicts within the intermediate artifacts, as well as the final model and strategies our participants used to identify and resolve conflicts.

Traceability was ensured in both groups through the tool support, which linked each model element in the intermediate artifact to quotes in the source data, and entities in the intermediate artifact could be mapped to elements in the final model.

The experiment was conducted over a five-week period. The participants were not given all interview transcripts at once. Instead we divided the data up in chunks and each week handed them one or two additional interview transcripts. This way, even though the data were not collected by participants themselves, the natural temporal progression of collecting more data as the project progresses could be maintained.

The order in which the interview data were handed out to participants was the same in which the data were originally gathered.

All participants were asked to keep a study diary, where they documented their experience after each work session, which they scheduled at their own discretion. The study diary approach is described in Sect. 5.1.1.

The timeline of one week of the study is outlined in Fig. 1.

In each week, we had an individual interview with each of the participants to assess their progress. During some of the interviews, a second researcher was present, ensuring *investigator triangulation* [25]. These interviews had the purpose of measuring a self-assessment of competency with regards to method and domain, as well as a documentation of the expended effort. Further, the interview was conducted as a semi-structured conversation on the participants' success or problems they were experiencing during the latest iteration of applying the respective method. The interview guideline was adapted each week to reflect both expected changes over time, like enquiring about saturation toward the end of the study, and also to include questions that were brought up by the entries in the study diary.

The interviews and the artifacts we analyzed are described in more detail in Sect. 3.4.

Table 1 Method comparison

	QDAcity-RE [30]	Strategic reading [14]
Methodological origin	Qualitative research	Education
Application Domain	RE, Domain modeling	Design science research, Domain modeling
Motivation	Guide domain experts creating domain models	Create purposeful artifacts for DSR and analyzing research papers
Process	Iterative process based on three coding steps	Execution of seven reading strategies
Resulting artifacts	Annotated text, conceptual model as UML	Annotated text, root-cause analysis (RCA) as mind map
Completeness	Theoretical sampling, coding paradigm, saturation	Root-cause analysis (RCA)
Consistency	Trace links, constant comparison	Trace links
Traceability	Between annotations (called: codings) in text and elements of UML by codes	Between annotations in text and elements of mind map by annotations
Tool support	QDAcity	Mendeley as reading platform, MindMeister as modelling tool, DScaffolding a Google Chrome plugin to connect Mendeley and MindMeister

3.1 Data on target domain

The data we used for the evaluation of the analysis phase of the method are the same as in our exploratory study on HR development [30]. The conceptual domain model created in our exploratory study had already been evaluated by experts to be an appropriate model of the domain. Therefore, we used this model as a base line of comparison for both groups.

The interview data given to the participants to be analyzed were the transcriptions of six semi-structured interviews with domain experts. The data were collected in the context of structuring a domain for requirements elicitation with the purpose of replacing an existing system for HR management in a medium-sized company in the financial services industry. The data collection was performed by us in parallel to the standard elicitation process the company pursued which wanted to develop the software. The interviews ranged in length from five to fifteen pages each.

3.2 Comparison of analysis methods

In order to evaluate the QDAcity-RE method, we searched for a method with similar characteristics for comparison. The method described by Diaz [14], based on strategic reading, features a similar level of pre-RS traceability between the elements of the domain model and the underlying (source) text on which the analysis is based [14]. Furthermore, both methods, QDAcity-RE and strategic reading are tool supported, which enforces trace links between model elements and text quotes. Table 1 presents a comparison of both methods.

The resulting artifacts of both methods differ from each other, while the QDAcity-RE method creates a UML model, with strategic reading a mind map is the result of the analysis. Therefore, an additional step is necessary to arrive at a model in the same notation. The strategic reading method was extended to further translate the resulting mind map

into a UML class diagram. The mind map thus served as an intermediate artifact similar to the code system in the QDAcity group.

3.2.1 QDAcity-RE

The QDAcity-RE method [30] is based on QDA to support the creation of complete, consistent, and traceable domain models. The method process is outlined in figure 2.

At the core of our method is an iterative process. It comprises data gathering from stakeholders, an analysis that guides further data gathering, and a stopping criterion, called *saturation*. *Saturation* is used to decide when to conclude the iterative cycle of data gathering and analysis and to consider the current state of analysis sufficiently accurate and complete.

During the analysis process, analysts annotate the input material such as interviews and workshop transcripts, with *codes* which are structured hierarchically in a *code system*. The process is called *coding* and is structured in three phases: *open*, *axial* and *selective coding* [30]. At the end of each iteration, the code system is refined through constant comparison to ensure consistency. Constant comparison requires the analyst to search for evidence and contradictions in previously analyzed data when new concepts come up, as well as search for corroborating and conflicting instances of previously found concepts in new data. This forces a frequent move between micro- and macro-perspectives and fosters a consistent perspective covering all gathered data.

The analysis, as well as the sampling strategy, is underpinned by a *coding paradigm*. The coding paradigm defines five dimensions: actions/strategies, consequences, causal conditions, contextual conditions, and structural conditions. The paradigm is meant to ensure all relevant dimensions of the problem domain are captured sufficiently. The identification of information gaps leads to the adaptation of

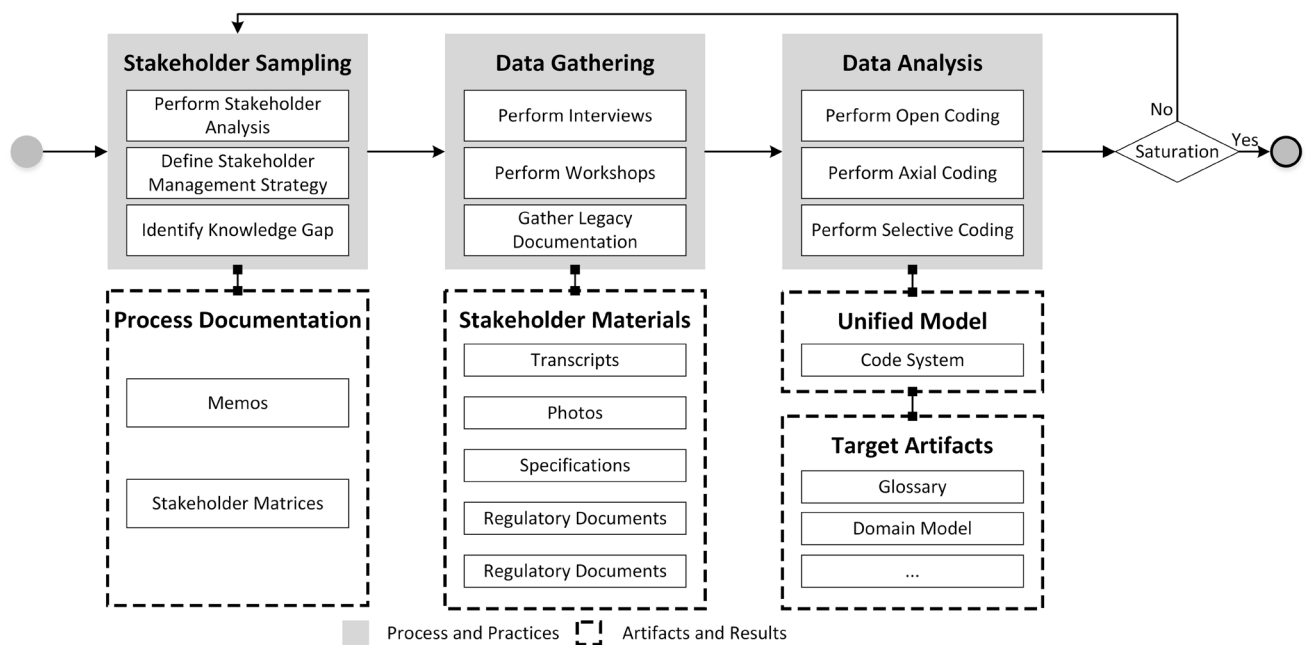


Fig. 2 The QDAcity-RE process for structural domain modeling [30]

the sampling strategy to fill these information gaps. This is called *theoretical sampling*.

The QDAcity-RE method is tool supported by a web-based platform for QDA called *QDAcity*¹. Throughout the rest of this article, *QDAcity-RE* always refers to our method, and *QDAcity* always refers to the tool.

QDAcity is built as an integrated analysis and modeling tool. The cloud-based application allows for all basic functions for coding text, i.e., creating and managing a code system and text documents, and assigning codes to text segments. Codes can be defined through code book entries with a short definition and instructions on when to use and when not to use, and participants were encouraged to use this feature. Codes can further be dragged and dropped into an editor for UML class diagrams. This allows for modeling relationships between these concepts and gives a different visual representation to the code system. All changes made to either the class or the code are reflected in the other. At any point, also during work in the UML editor, a list of coded text segments of a selected code, or class can be displayed without switching back to the text documents. Further, a table can be generated showing most frequently co-located codes with a percentage of the text area of overlap. This helps finding potentially missed relationships between concepts.

We previously evaluated our method using four exploratory project studies with industry partners. These projects

also guided the development of the QDAcity-RE method [30]. In these exploratory cases, as well as in other published research, QDA-based domain modeling has shown great promise in improving the final artifact's quality while at the same time improving the documentation of the process.

3.2.2 Strategic reading

Strategic reading, similar to QDAcity-RE, also is an attempt to codify the step from unstructured source material to conceptual domain models and establishing traceability as a result of using the method.

Strategic reading is a term coined in the educational domain [33] and has been suggested for design science research and domain modeling [14, 15]. At its core, strategic reading requires the reader to actively employ all of the following seven reading strategies:

- *Activating*: Recalling relevant knowledge from long-term memory.
- *Inferring*: Connecting implicit and explicit meaning of the text to one's own experiences and knowledge.
- *Monitoring-Clarifying*: Actively checking regularly if the text is fully comprehended.
- *Questioning*: Self-questioning and finding the answers to those questions in the text.
- *Searching-Selecting*: Identifying samples in the text that answer these questions, that solve problems, and that define important terms.

¹ <https://qdacity.com>.

- *Summarizing*: Writing a summary of the text in one’s own words.
- *Visualizing-Organizing*: Creating a “mental image”, some graphical representation of the text, to elicit meaning from the text.

The goal of the method described by Diaz is to make annotations in the literature traceable to *root-cause analysis* (RCA) issues that triggered the literature review in the first place. And, the other way around, to have related annotations readily accessible while performing the RCA. The main artifact of the RCA in this context is a mind map, rather than a UML diagram. The original motivation was to guide researchers reading and analyzing research papers, but has been extended to general domain modeling. Therefore, the control method not only shares a common purpose with our method but also shares its genesis in the context of research methods.

This analysis method used by the control group is also tool-supported. It uses *Mendeley*² as a reading platform and *MindMeister*³ as a modeling tool for mind maps. Both tools are connected using the Google Chrome plugin called *DScaffolding*.⁴

Through this linkage, sections of text in Mendeley will be automatically copied as child nodes to one of seven nodes in the MindMeister mind map which are marked as current reading purposes. Each of the up to seven nodes maps to one of the seven colors available in Mendeley for color-coded annotations.

The interaction between the RCA platform and the reading tool is facilitated by the “Purpose Pipe”, which channels RCA issues to the reading platform, and the “Annotation Pipe” which channels literature quotes to the RCA platform [14]. The two pipes are depicted in Fig. 3. The selection of nodes as a reading purpose happens in Mindmeister and is depicted in the figure with a thumbtack attached to the node. Each reading purpose is then assigned a color in Mendeley.

However, the philosophy of the tool support for strategic reading is entirely different in the sense that it purposefully only connected existing tools whereas the tool-support for the QDAcity-RE integrated both the annotation phase of the process and the modeling phase of the process. DScaffolding is constrained by attributes of the connected tools that were not originally designed for this kind of inter-tool compatibility. The most significant example of these limitations is that only eight colors are available for highlighting in Mendeley. Since one of the colors is reserved, this leads to only

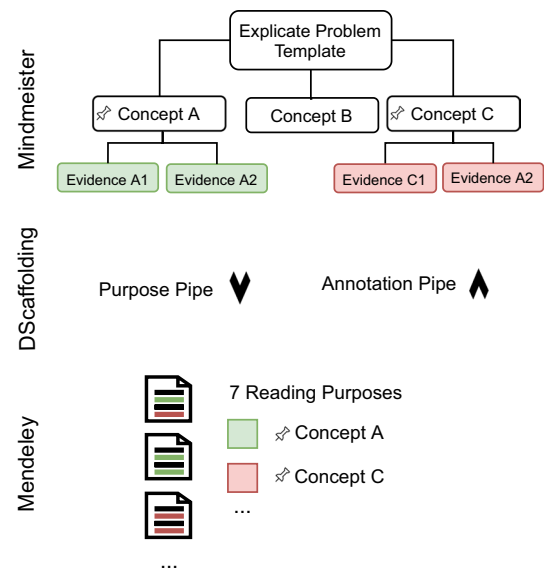


Fig. 3 Purpose pipe and annotation pipe

seven reading purposes which can be defined at any point in time. While reading purposes can be changed throughout the process, the seven colors remain the same. This leads to the seven colors being re-used for different concepts in the mind map.

3.3 Participant sampling

The participants of our experiments were ten students in the Master’s degree program of computer science or international information systems at the Friedrich Alexander University Erlangen-Nürnberg during the summer semester of 2019. We advertised the experiment in search for recruits within our courses and through facebook groups for students of the respective degree program at our university. As incentive for participation with an estimated effort of 50 hours in total, we offered 500EUR monetary compensation.

Each participant filled out an entry-survey covering the following demographic aspects:

- Age
- Gender
- Degree program currently enrolled in
- Prior academic degrees
- Work experience in the software industry in years
- Experience with QDA and domain modeling on a six-point scale
- Whether they have taken the course on conceptual modeling or UML at our university or equivalent courses at another university

² <https://www.mendeley.com>.

³ <https://www.mindmeister.com>.

⁴ <https://chrome.google.com/webstore/detail/dscaffolding/hkgmnnjalpmapogadekngkbgbdjlnne>.

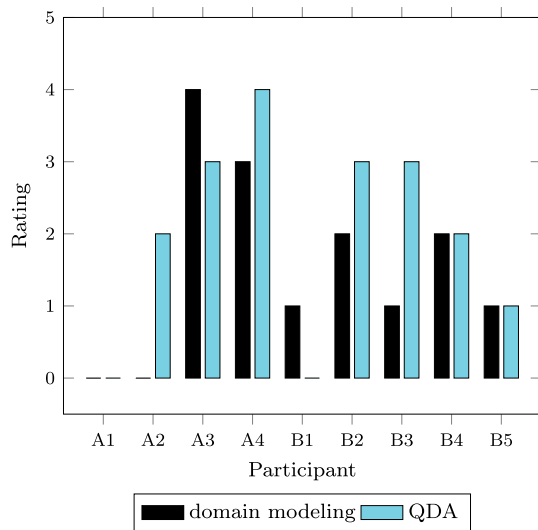


Fig. 4 Self-assessed competency of participants

Of the 19 candidates who finished the entry-survey, we selected ten that allowed for an equal distribution of the survey dimensions into two groups. Of the ten, nine participants remained in the study through the end.

We aimed at building two comparable groups in terms of the QDA and domain modeling competences of the study participants. To evaluate these competences, we asked the participants to self-assess their domain modeling experience and their QDA experience. The responses are shown in Fig. 4. The rating was performed on a scale from 0 points defined as *None* to 5 points defined as *Expert*. The values in between should be interpreted as a discrete linear scale and were not labeled. Participants whose ID starts with *A* were in the control group and participants whose ID starts with *B* were in the QDAcity group.

The average for domain modeling is 1.55, while the average for QDA is significantly higher at 2. We assume the overall assessment of QDA competence being assessed significantly higher than for domain modeling is due to the fact that six of the participants had previously taken our course on research methods, where part of the exercise includes the qualitative analysis of expert interviews. However, none of the participants had been involved in a research project outside of the teaching setting at university. None of the participants considered themselves an expert in any of the two categories, which was expected.

The participants with no or little experience with domain modeling allowed us to examine and compare the learning curve for inexperienced analysts of both methods. Since domain modeling relies heavily on the analysts' experience, a method that provides more guidance for novices is especially beneficial.

Table 2 Demographic averages by group

Characteristic	QDAcity	Control
Sample size	5	4
Age	29.6	28
Gender	40% m 60% f	75% m 25% f
Work experience (years)	0.3	0.125
Competence in modeling	1.8	2.25
Competence in QDA	1.4	1.75

Based on the answers in the initial survey, we divided the participants into two groups, with a sample size of five each. However, one participant in the control group withdrew his participation in the third week of the study leaving only nine valid results of final artifacts to consider.

The demographics of the average participant in each group is presented in Table 2. We tried to maximize equal distribution of these characteristics in both groups at the start of the experiment to avoid selection bias. Hence, the assignment of participants to a group was not random, but after all participants were assigned to one group, the treatment was assigned randomly to avoid experimenter bias during the selection of groups.

Table 2 reflects only the participants who finished the study. After one participant dropped out mid-study, re-shuffling of the groups was no longer a possibility, which is why the groups are not as similar as they were at the beginning of the experiment. In both dimensions for self-assessed competency, the average slightly favors the control group.

In the beginning of the study, we held kickoff meetings with both groups separately, where each was given a 90-minute introduction into the method, and how to set up the tooling. They were also provided with the respective paper describing the method and instructed to read it before starting the analysis. The introduction was interactive, and participants were encouraged to ask questions and demand clarifications about anything that might be unclear about the method application after reading the paper and at any time during the experiment.

3.4 Evaluation methods

We analyzed the results of the experiment in two dimensions: One dimension is (a) the quality of the resulting artifacts, the domain models, and how well they compared to a baseline model. The baseline model had previously been evaluated by domain experts as being a good representation of the domain [30]. The other dimension is (b) how the participants experienced the analysis process through qualitative analysis of a study diary and semi-structured interviews.

How these two aspects are separately evaluated is described in the following Sect. 4 for (a), and Sect. 5 for (b).

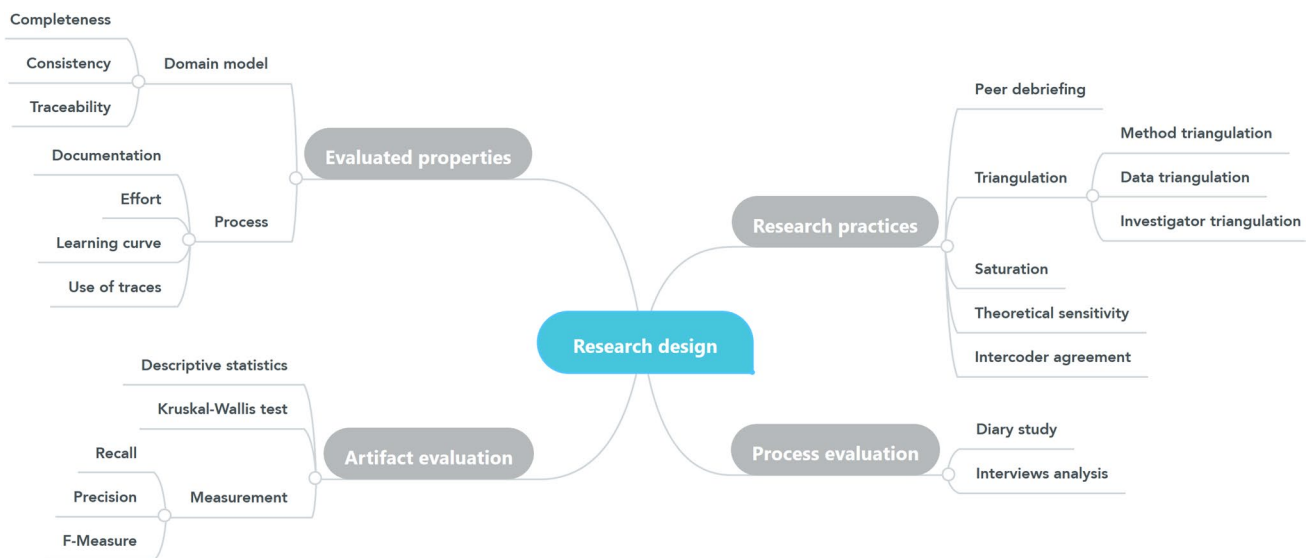


Fig. 5 Evaluation overview

As subject of the evaluation, three different artifacts were available to us:

- The resulting domain model artifacts (9 final artifacts and snapshots for every week)
- The 98 study diary entries from each work session
- The 40 semi-structured interviews

Our study resulted in a number of domain models produced by experiment participants and other artifacts such as descriptive statistics and QDA process measurements such as recall, precision, and f-measure. We assessed the resulting domain models using the evaluation criteria of completeness, consistency, and traceability. This enabled the comparison between the different methods followed by the two participant groups of the study.

Beyond the artifacts, we also evaluated the analysis and domain modeling process each participant followed. For this, we used interview analysis and diary study methods, where each participant would document their progress and challenges throughout the experiment.

Finally, we also focused on the internal validity of our study. We employed research methods such as peer debriefing, intercoder agreement, and triangulation of data, methods, and investigators. These practices ensured the rigor of our research method and the resulting findings.

For the summary of the above-mentioned evaluation strategy components, see Fig. 5.

4 Artifact evaluation

4.1 Evaluation method

Lindland et al. [32] established a widely adopted quality framework for conceptual models. They suggest an evaluation using three dimensions: syntactic, semantic, and pragmatic quality.

Syntactic quality is how well the model conforms to the language; semantic quality is how well the model conforms to the domain; and pragmatic quality is how well the model conforms to its audience interpretation.

In the following list, we present how these dimensions are tied to model properties we test for evaluating the artifact quality.

- (a) Syntactic quality
 - Formal syntax (UML)
- (b) Semantic quality
 - Consistency
 - Completeness
- (c) Pragmatic quality
 - Structuredness
 - Comprehensible complexity

Syntactic correctness was evaluated by counting syntax errors regarding the UML specification. We expected that syntactic correctness would be ensured through the use of tool support in both groups, still an independent evaluation ensured that any discrepancies were documented.

To assess whether our objectives concerning the completeness and consistency of the resulting domain model have been met, we compared the results of both the experimental and the control group with the expert solution.

We compared both groups against the expert model to discover whether any of the groups covered a larger set of significant domain concepts and relationships by measuring recall. *Recall* is defined as the ratio of true positives to selected elements.

To determine whether any one group had a significant larger amount of potentially superfluous concepts, we evaluated precision. *Precision* is defined as the ratio of true positives to relevant elements.

We also present the *f-measure*, which is the harmonic mean of both *recall* and *precision*.

Besides comparing the mean values of both groups and their standard deviation, we determined whether there was stochastic superiority in one of the groups. We compared the results of both groups using a Kruskal–Wallis one-way analysis of variance (ANOVA) on ranks [31], which for two samples is similar to a Wilcoxon test [49] in that it compares the ranks of values for each variable rather than the actual values.

The Kruskal–Wallis test [31] allows for a comparison of two or more groups. In this experiment, we compared just two groups, while adding another group of comparison constitutes potential future work extending on the experiment presented here. The reason for choosing a nonparametric test was the small population of our sample. Normal distribution of all relevant features of both our groups cannot be assumed. Using the Kruskal–Wallis H-test, we could test whether the probability distributions in all groups were significantly different or not.

A key use-case for conceptual modeling in requirements engineering is improving the communication with stakeholders who may not be trained in object oriented modeling. The lower complexity of the model is, therefore, beneficial to removing barriers to understand the model and use it as a shared basis of communication. We therefore evaluated the complexity of the generated model and our model solution and thus evaluated its pragmatic quality.

4.2 Results

We compared the resulting artifacts of each participant with the expert solution and counted the concepts that had an equivalence in the expert solution, as well as those that did not. With this comparison, we created equivalence classes

Table 3 Artifact comparison with expert solution—Control Group

Participant	Recall (%)	Precision (%)	F-measure (%)
Average	29.688	57.981	38.160
A1	41.667	58.824	48.780
A2	39.583	59.375	47.500
A3	16.667	47.059	24.615
A4	20.833	66.667	31.746

Table 4 Artifact comparison with expert solution—QDAcity Group

Participant	Recall (%)	Precision (%)	F-measure (%)
Average	48.958	55.258	49.474
B1	66.667	47.059	55.172
B2	43.750	32.308	37.168
B3	43.750	58.333	50.000
B4	41.667	83.333	55.556
B5	56.250	52.941	54.545

of concept names which we consider synonyms for the same underlying constructs. For instance “Training”, “Training Course”, “Training Measure” and “Personnel Development Measure” were considered synonyms, and if either of these concepts existed, it scored a true positive. Another example would be “Onboarding” and “New Employee Integration”. However, if the concepts were too abstract, they were not considered equivalent. For instance, a concept labeled “Documentation” was considered too broad to be considered equivalent to any of the more specific concepts “Target Agreement” or “Performance Target Agreement”.

The results of this comparison are shown in Table 3 for the control group and in Table 4 for the QDAcity group.

Comparing the two groups, it is apparent that, on average, the QDAcity group scored significantly higher in recall, while the control group scored marginally higher in precision. This suggests that participants using QDAcity-RE produced more detailed models overall. This is also reflected in the total number of concepts present in the models. The models in the control group contained, on average, only 24.5 concepts, while for the QDAcity group, that number is almost double, with 48.8 on average.

While the number of concepts itself is not an indicator of quality, significantly reduced recall to a comprehensive model is indicative of that something might have been missed. On closer inspection of the results, there is no single area that was missed consistently across participants, but the reduced recall was indeed due to important subsets of the domain missing. While some models missed a detailed modeling of actors (i.e. “Supervisor”, “speciality department”, “finance department”, “work council”, etc.), others were missing methods of personal development

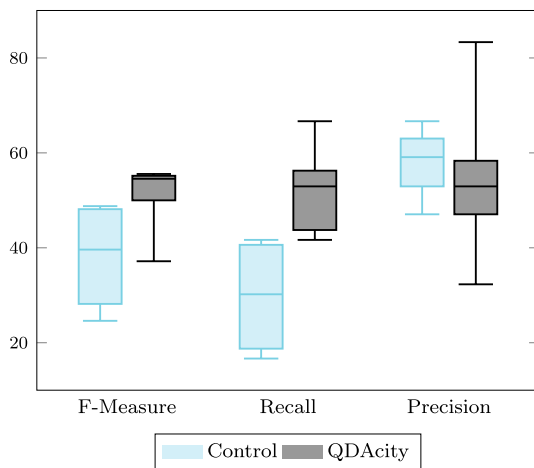


Fig. 6 Boxplot of recall, precision and f-measure by group

(i.e., “Mentoring”, “Coaching”, “External Training”, “Virtual Training”, etc.), and yet others were missing explicit modeling of competencies (i.e., “Method Competencies”, “Social Competencies”, “Professional Skill”, “Competency Profile”, etc.).

Some concepts frequently appeared in more detail across both groups or covered an area that was frequently mentioned in the interviews, which we did not include in our domain model for the purpose of avoiding idiosyncrasies of single companies. As an example of this, participant artifacts often included a concept for the tools used in the companies, like SAP, PeopleSoft, or Microsoft Excel.

The large deviation within the precision measurement in the QDAcity group, as shown in Fig. 6, is an artifact of different modeling styles more than of understanding of the domain. We come to this assumption because often, two participants had similar concepts in the code system, but some participants decided to model almost all codes as their own concepts, while others were more selective.

When comparing the data from the entry survey with the results in Tables 3 and 4, it becomes apparent, that there was no correlation of the self-assessment of competency to the scoring with the final artifact. The best performer in the control group claimed no experience in either domain modeling or QDA. The pre-existing experience of an individual participant was less relevant for

our analysis, since these effects seem to be offsets to the baseline of each participant, not affecting trends up- or downward.

4.2.1 Statistical evaluation

Figure 6 shows a boxplot of the precision, recall and f-measure [48] by group, and the descriptive statistics are also presented in tabular form in Table 5, where the column for f1 stands for f-measure (also called F1-Score).

The null and research hypotheses of our Kruskal–Wallis test are as follows:

H₀: The probabilities distributions in both groups are the same.

H₁: The probabilities distributions in both groups are not the same.

The null hypothesis can be rejected if the H-value of the test is larger than a critical value for a given significance level α .

The hypotheses have to be evaluated for each of the measured variables.

The H-value is computed as follows:

$$H = \frac{12}{n(n+1)} \sum_i \frac{R_i^2}{n_i} - 3(n+1) \quad (1)$$

with R_i being the sum of all ranks within group i , n_i is the number of valid results in group i , and n being the total number of samples in all groups. The aggregation of the rankings of each sample split across the two groups, as well as their mean, is shown in Table 6.

The test was run for each measured attribute, recall and precision, separately, so we might see results where for one of them the null hypothesis holds and for the other one it does not.

Table 5 Descriptive statistics

Measure	QDAcity			Control		
	Rec.	Prec.	f1	Rec.	Prec.	f1
Min.	41.7	32.3	37.2	16.7	47.1	24.6
Max.	66.7	83.3	55.6	41.7	66.7	48.8
Mean	50.4	54.8	50.5	29.7	58.0	38.1
Std.Dev	10.8	18.7	7.8	12.8	8.1	11.9

Table 6 Ranking for H-Test

Measure	Sum of Ranks		Mean Rank	
	QDAcity	Control	QDAcity	Control
Recall	34.5	10.5	6.9	2.625
Precision	21.5	23.5	4.3	5.875
F-Measure	33	12	6.6	3

Table 7 Kruskal–Wallis H-Test results

Measurement	H-value	<i>p</i> value
Recall	5.507	0.0189
Precision	0.741	0.3893
F-Measure	3.84	0.05

With only two groups of samples, the degrees of freedom in our test is one.

The computed H-values and *p* values are presented in Table 7.

The χ^2 approximation for the critical value of H for a significance level of 0.05 and one degree of freedom is 3.84, which is smaller than the H-value for recall and equal to the H-value for the F-Measure in Table 7.

With an α of 0.05, the null hypothesis H_0 can be rejected in favor of our research hypothesis H_1 for the variables of *F-Measure* and *Recall*. Hence, we conclude that it is stochastically better to pick a random participant using the QDAcity-RE method, than picking one using the control method. However, for the variable *Precision*, the test could not conclude a statistically significant difference on the ranks of the participants' results.

This conforms with our observation that participants of the QDAcity group did produce models with about double the number of concepts, overall being more detailed. It is obvious that the more detailed two models become, the more room for difference in variation on these details arises. A portion of the false positive concepts modeled could have been avoided due to a better understanding of conceptual modeling by the participants.

5 Process evaluation

5.1 Evaluation method

To gain insight into the application of the method from a process perspective, we analyzed the diary study and followed up in semi-structured interviews. The diary study helped us to gather data on the following properties of method application: efficiency of the method, the learning

curve, and perception of how accurate the model represents all data as a subjective proxy for completeness. The interviews allowed us to follow up on any encountered problems and further allowed us to inquire about expected pain-points such as the context-switch between reading and modeling, and the employed strategies for ensuring consistency. We also used the interviews to analyze the utilization of the provided traceability, and the use cases when they were considered helpful.

While the interview outline changed over time, the elements of the diary remained constant throughout the study. Information given in the diary guided the line of questions in the interviews.

5.1.1 Diary study

Diary studies have been used in computer science and information systems research to study a variety of phenomena such as the contribution to particular demographics to open source [13], or the activities of novice professional software developers [2].

Participant samples in diary studies are frequently below ten participants. This allows for an in depth analysis and opens up the possibility to triangulate data from the study diary with other forms of data, like observation [2], or interviews [1], adding method- and data-triangulation [25].

The study diary was conducted using an online form containing 11 questions, in which participants were asked to answer after each work session. The form was composed of four sections:

1. Base data on the session including time spent, tasks performed, and goal of the session
2. Reflection on results including changes to the artifacts, novel insights, and a rating on the confidence in the results
3. Reflection on process including problems and challenges encountered, a rating of the success of the session and an explanation for the chosen rating
4. Additional comments

The ratings within the diary study were all on a seven-point Likert scale. Participants had to create a diary entry after every work session, leading to 98 entries throughout our experiment.

The diary study allowed us to track the same variable over time and better understand the learning curve of each method, giving our experiment a repeated-measure-design.

The results of the analysis of the study diary are presented in Sect. 5.2.1.

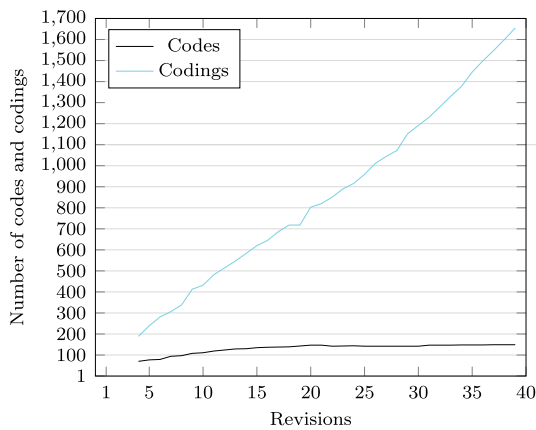


Fig. 7 Number of codes and codings

5.1.2 Interviews

We performed 38 in-person and two online interviews. The total of 40 individual interviews were conducted weekly. The interviews took between 30 and 90 minutes and the recordings were subsequently transcribed and coded using QDAcity.

To ensure that, the emerging codes were adequate for the analyzed data, and to add investigator triangulation [25] as a means of quality assurance, we performed three iterations of inter-rater reliability, involving three additional coders. The first two rounds were conducted after the first ten interviews had been coded by the first author. Through this process, definitions of many codes could be refined, the structure could be improved, and a few missing codes were added.

Due to the complexity of the code system, each inter-coder focused only on a subset of codes. While the third author coded all codes under the category *process*→*analysis process*, the second and fourth author coded all codes under the categories *process*→*challenges* and *process*→*strategies*. During the first round of inter-rater reliability, five documents were considered, two from the QDAcity group, and three from the control group.

Inter-raters documented ten codes they considered missing, 15 suggestions for renaming and re-definition of codes, and two suggestions for re-categorizing, as well as three general comments that impacted a common understanding of the code system in a shared document. After the first round concluded, each of the points were jointly discussed and resolved to represent a common interpretation of the data.

A second round of interrater-reliability was performed in immediate succession also on the data of the first two weeks of the study. Here, three new codes were suggested and one general comment was made. A third round was added after three quarters of the interviews had been coded. In this last round, four new codes were suggested, four

candidates for re-defining codes, as well as one suggestion for re-categorizing.

A significant portion of the suggested new codes could be resolved through a discussion, since there existed a similar code in a different category that was not coded by the suggesting intercoder. As a result of the discussions, a total of four codes were added, eight codes were renamed, 21 codes were redefined, and five were relocated.

The total number of codes and coding over time is shown in Fig. 7. While the number of codings in the project climbs linearly, the number of codes converges at around revision 20.

The results of the analysis of the interview data are presented in Sect. 5.2.2.

5.1.3 Saturation

We configured QDAcity to calculate saturation throughout the qualitative analysis of the interview data. We measured saturation after each additional coded interview based on changes of the number of different types of user interactions with the data.

We operationalized saturation of our analysis based on suggestions of Francis et al. [20]. After an initial sample size of ten, we consider saturation reached when within the last three iterations, no significant changes occurred to our theory, manifested in changes to the code system. For this, we used logs of changes to our code system over time, which were categorized in additions of new codes, deletion of existing codes and relocation of codes, meaning changes to the hierarchy. With this information, we calculated the percentage of changes within the last three iterations compared to all changes. The saturation data presented here were documented automatically by QDAcity, which we used for our analysis.

The results of the saturation measurement are presented in Sect. 5.2.3.

5.2 Process evaluation results

While the comparison of the artifacts gave us some insight into the output of the method, we also were interested in the experience of the participants when applying the method, and what challenges they faced. A focus with both the diary study and the interviews were also on investigating changes over time. Significant changes in the answers given in the diary were picked up in the interviews and induced changes to the interview guideline, while the questions in the diary remained constant.

We distinguish quotes presented in this section only by group, not by participant. Quotes marked with an [A] were made by a participant of the control group, while those

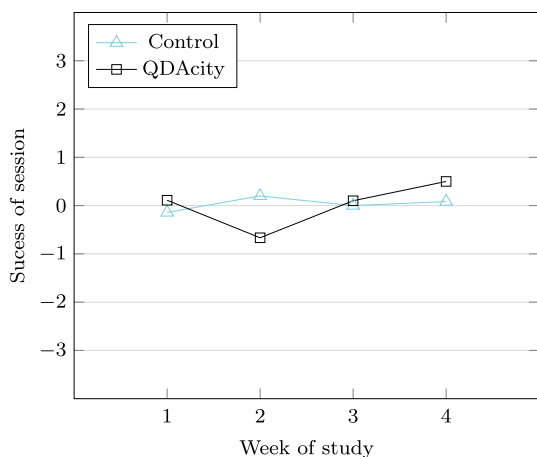


Fig. 8 Self-assessment of session success

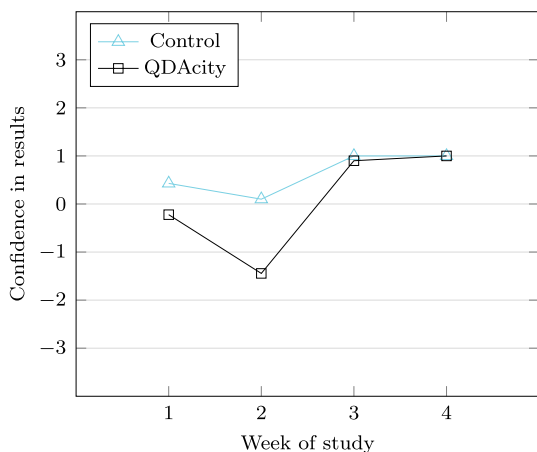


Fig. 9 Self-assessment in confidence in results

marked with a [B] were made by a participant of the QDAcity group.

We present the main results of the study diary in Sect. 5.2.1 and the results from the interview analysis in Sect. 5.2.2.

5.2.1 Study diary

Our participants made 98 diary entries in total.

The evaluation of the participants' self-assessment of the success of the work session is presented in Fig. 8, and the self-assessment of the confidence in the resulting artifacts is shown in Fig. 9.

The rating of the success of the session remained in a short range around the neutral mark for both groups. However, there are small differences in week two and four, where the qualitative data we gathered offer some explanation. In week two, the average rating in the control group went up, while that in the QDAcity group went down. Our

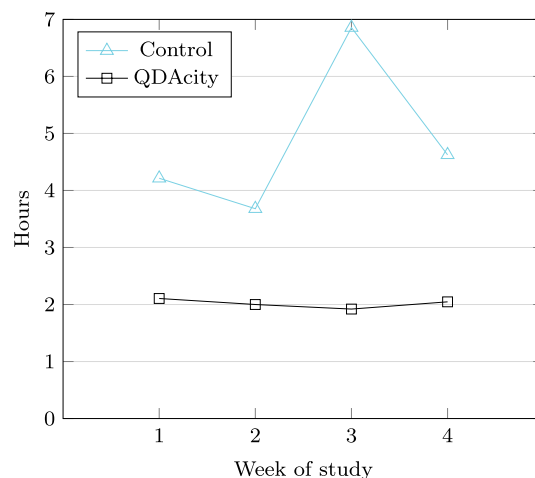


Fig. 10 Average length of work session

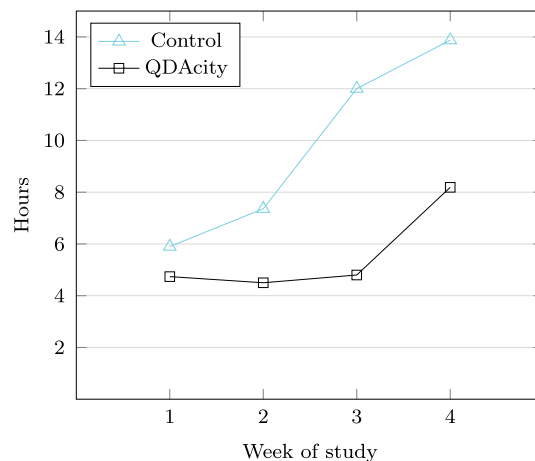


Fig. 11 Total time spent by participants

explanation for this observation is that while the participants in the control group started to get familiar with categories in the provided template, participants in the QDAcity group struggled creating a structure in their code system, because it was more reliant on them finding good categories of their own. On top of being unsure how to categorize, the coding paradigm which does provide some structure, was mostly perceived as an additional burden or confusing at this stage in the study.

It became apparent in the first round of interviews that some participants were not aware that they should apply it, or how to apply it. Then, in the diary entries from the second week, quotes like the following were a common theme:

A problem during this session was that I was sometimes unsure to which category regarding the coding paradigm the codes belong to. [B]

Two other categories of problems more frequently mentioned in week two were inexperience with UML and tooling-related issues. Both of which were reported by both groups. Other method-specific reasons given from the QDAcity group were challenges with the code granularity and the creation of good categories, and especially that the latter could consume “endless hours”.

The situation then turned around in week four, as participants became more acquainted with the method over time. Our interpretation of this observation is a confirmation of our expectation that our method requires a significant effort to be learned, which seems to pay out over a longer period of time. The fact that this could be observed, and was confirmed by participants within the interviews, within the relatively short time of the experiment is encouraging. The disadvantage in the second week was, hence, turned into a small advantage in the fourth week.

The evaluation of the diary showed significant differences in the average length of the session as shown in Fig. 10. While participants in the QDAcity group spent around two hours per session consistently throughout the study, participants of the control group spent, on average, more than twice as long in each session. The total time expended for the study was also significantly higher for the control group, as shown in Fig. 11.

Both groups also steadily increased their total hours spent on the analysis per week, with the gap between the two groups widening.

The overall increase in effort within both groups was expected, since the consolidation of all evidence in a single intermediate artifact, be it the code system or the mind map, requires additional effort. In the QDAcity-RE method, this is codified as the *constant comparison* paradigm, but the control group method enforced the consistency between all interviews just the same. In fact, in our analysis, the code *process*→*challenges*→*consistency between interviews* was applied frequently and equally across both groups, 33 times in the QDAcity group, 32 times in the control group.

What we expected, but did not observe, was a spike in time expended in week one due to familiarization with the novel methods and tools. The study diaries do show, that the purpose of many of the initial work sessions had a defined goal of setting everything up, understanding the tools and understanding the notation and the templates.

While the gap between the two groups regarding the total effort expended per participant was fairly small in the beginning with a 1.2 hour difference, this gap in favor of the QDAcity group widened throughout the study up to an astonishing 5.7 hour difference for the average participant. The hypothesis generated by this observation is that this difference may, in part, be caused by the fact that the frequency at which switching between the different tools is required increases exponentially with number of nodes in the mind

map. This problem may be exacerbated by the limitation of a maximum of seven colors being assigned as reading purpose at any given time. Hence, with a growing number of nodes as potential reading purpose, the changing and re-using of colors becomes ever more frequent.

Switching between the tools is an overhead not required in the QDAcity group. Furthermore, through the reuse of colors, information that may be visible on first glance in QDAcity may only be revealed by documenting the name of the assigned node in the comment within Mendeley, rather than just relying on the color.

Analyzing the noted challenges over time, it was also apparent and unsurprising that the problem of restructuring the code system and creating subcategories weighed increasingly heavy on participants.

In the final week, the challenges related mostly to finding good names for the already existing code system that unify the information with distinct but related vocabulary in all interviews.

5.2.2 Participant interviews

Participants frequently mentioned possibilities for extension of the QDAcity-RE method to include other artifacts such as process models and use case diagrams.

While the process of deriving models from the annotations was generally perceived as useful, most of the issues our participants encountered were less method related, but more focused on tooling limitations.

One participant of the control group summarized the experience as follows:

[I] think the big constraint in the whole experiment, if I was to summarize it, was not the model [mind map template] but definitely the tools that I used. [A]

This was reflected in our coding of the interviews, with *tooling problems* being one of the most frequently coded codes in the category *process*→*challenges* across both groups. There emerged total of ten subcategories with six being specific to the control group, one specific to the QDAcity group and three independent of the groups, such as *tool restricting method application* and *accidental changes*. The code *tool restricting method application* was coded 21 times in nine separate interviews, and while it was not a method specific code, it only occurred exclusively in interviews with all participants of the control group.

One related reason why the QDAcity group seems to be less affected by the increased effort of keeping consistency between all interviews might be the more tight integration between model and text though a single tool. This allowed for more frequent switching back and forth between assigning codes and modeling concepts and their relationships.

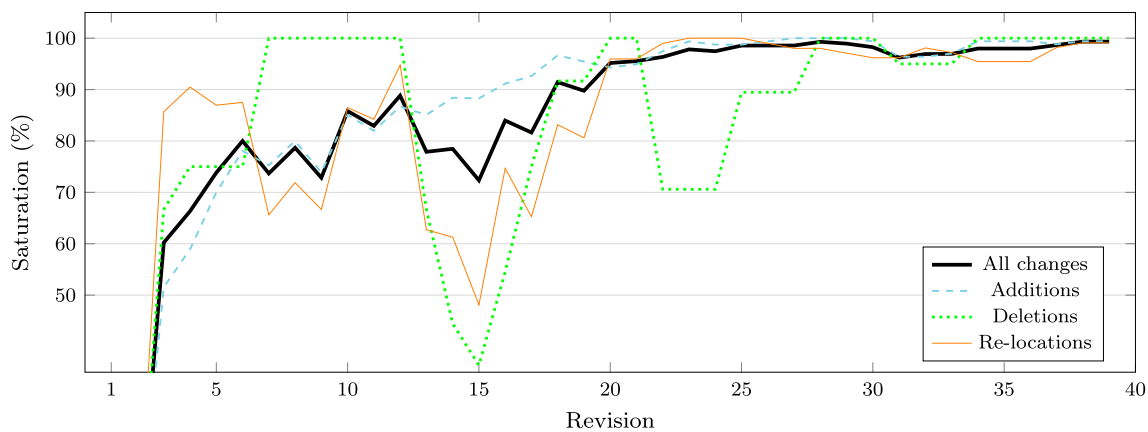


Fig. 12 Saturation over time

In fact, one participant from the QDAcity group told us the following in their last interview:

The process is very, very fluent, how I decided to assign the codes and so on. [...] It was quite easy for me this time, and I realized also that it did not take so much time compared to the beginning. [B]

They then went on to describe that their focus shifted to better definitions of codes, and general clean-up work before the conclusion of the study, which in the end did consume more time, explaining the spike in time spent for week four in the QDAcity group (Fig. 11).

Comparing the context switch required between reading and modeling, there was a significant difference in perception between the two groups. While the switching back and forth between model and text was mostly described as easy by the QDAcity group, several participants of the control group told us they employed workarounds for minimizing the number of times a switch between the different tools was required. One code that was more frequently applied in interviews with the control group than in interviews was *annotating broad concepts first*. This code had a 49.28% overlap in coded text with the code *context switch: reading/modeling*. The close relationship of these two codes suggests that the practice of focusing on broader concepts was employed as a means to ease the burden of the context switch particularly in the control group. This also may be one factor contributing to significantly lower recall value of the control group due to the less detailed models.

The second context switch, when going from the intermediate artifact (code system or mind map) to the UML model, was consistently postponed to the end of each work session in the control group. We could observe that while in the beginning, participants in the QDAcity group also postponed the UML modeling to the end, after getting more acquainted with the tool, there was a shift toward more

intertwined reading, coding and UML modeling. One participant of the QDAcity group told us the following already after the second week:

In contrast to the first interview, I already also did that UML part directly from the beginning on. Not after my full coding process, instead in parallel to my coding process that was definitely easier for me in order to fulfill a good UML model [B]

The interviews provided interesting insight on how the provided traceability was utilized. Both methods forced the participants to document traces from all entities in the model to evidence in the original data in the form of text segments. All participants made frequent use of these, and described them as helpful in particular to ensure, that among the evidence for each element there were no unresolved conflicts. Participants also used the overview of textual sources when modeling relationships and to refresh their understanding of a concept, rather than relying purely on an often very abstract definition.

Some participants of the QDAcity group also described, that while they were in the UML modeler, they were occasionally inspired to go back to the text data to look for evidence of concepts and relationships that became apparent to be missing more easily in the graphical view of the model. The ease of going back and forth was described as helpful in these instances. Participants also frequently checked a list of coded text segments available from inside the UML editor.

Since in the control group, the UML editor was a non-connected third party application in addition to MindMeister and Mendeley, this information was not integrated into the modeler. Yet still, participants described they found it useful to check up on a corresponding mind map node to a concept in question and look up the list of associated text segments there.

Despite the confidence rating on the resulting artifacts in the study diary only moved up slightly during the study, the qualitative feedback in the interviews suggests a significant change in attitude toward the final artifact. The following quote is representative of many participants' feelings in the first week:

I made a UML for interview one, which was, I was not happy with it because, at interview one I was kind of like, "I have no idea what's happening", and stuff. [A]

This completely changed toward the end of the study, leading to quotes like the following, indicating that participants of both groups started to feel more confident that their models are a good and stable representation of the domain:

For the sixth interview, there's just small modification, just add one more thing, one more class in the UML diagram, so I think if there's more interviews, I assume that there not be any large modification, just some minor change or in-depth new knowledge. [A]

5.2.3 Saturation

The saturation based on all changes, as well as the three categories of changes, are presented in Fig. 12. After ten iterations, we have reached a saturation of 72.85%, while in the final iteration, the value reached 99.33% with just one new code and one change to the hierarchy in the last three iterations.

The most significant restructuring happened with the analysis of the interviews from the second week of the experiment. At this time, participants could reflect on what they have learned from the first week and things that they changed to keep their models consistent with the information gathered from more than one expert.

6 Discussion

Strategic reading and QDAcity-RE both draw on the idea that besides creating a documentation artifact, the process of annotation also improves the comprehension of the analyzed text [36, 37, 39]. The integration of documentation as a side product of the analysis process worked in both cases.

Comparing the two methods, however, we found strong evidence supporting an answer to **RQ1**, that QDAcity-RE indeed creates more complete and consistent models of the domain, as compared to the method based on strategic reading, as measured through information retrieval metrics based on a baseline model.

Addressing **RQ2** about traceability, we found that traces helped both groups when modeling relationships between concepts. After finishing an initial model based on the

short-term memory of analyzing a particular interview, participants double-checked if any relationships to codes that may not have been present in the last interview were missed. Some participants reported that they used the traces instead of their own definitions they were required to document with each concept. While we believe this practice may have worked well in our small example, to the point where it could substitute writing definitions, it would not in a project where each element is linked to many more instances. Participants in the QDAcity group tended to use the traces more frequently because the evidence for each concept could be displayed within the UML editor.

One aspect that, according to our findings, needs some re-evaluation is the aspect of the coding paradigm as one of the elements that were deemed to guide the analysis, and the creation of new codes. Although this was sometimes the case, the evidence for this aspect being perceived as confusing is significantly larger, both in the study diary and in the interviews.

Similar complaints were also sometimes mentioned with regard to the explicate problem template in the mind map. However, participants still consistently adhered to it, whereas in QDAcity, the coding paradigm was frequently brushed aside to deal with at another point in time. Whether this means that the use of the coding paradigm should be more enforced or adapted to be more low-key is unclear. We did gather sporadic evidence that participants identified gaps in the code system when actively thinking about which dimensions are already covered and which are not.

We focused our research presented in this paper on providing a clear picture of one important fragment of our method, the analysis phase using interviews. We acknowledge that this is just one of many aspects. Gaining high rigor of the experimental methodology, abstractions from reality were deemed a worthy trade-off. However, it has to be acknowledged that the data sampling and data gathering play a huge role, which were not examined in this study. The appropriateness for other types of material also remained out of scope, however, we have previously used our method in more realistic settings with a larger variety of input material [30].

7 Limitations

Within this study, we only evaluated the analysis phase of QDAcity-RE. Iterative data gathering and analysis is an important cornerstone of QDAcity-RE and should be further evaluated in detail. We attribute the successful application of QDAcity-RE in our exploratory studies [30] in part to the iterative execution of all of these phases. Our solution of only focusing on the analysis phase in this experiment makes the results more easily comparable, because they were all

based on the same data, disregarding the ability of each participant to conduct interviews. Thus, we believe it led to more reliable results. We included an iterative aspect into our test by not giving participants access to all source data right away, they were not able to let their sampling strategy be guided by theoretical sampling.

To avoid carryover effects, we chose a between-subject design. This raises a concern about the homogeneity of the sample to make the groups comparable. However, we believe a within-subject design would have had stronger negative impact on any results than any inevitable difference in the characteristics of two groups when careful consideration of obvious confounding factors was exercised. Even though our population was small, we believe they are homogeneous enough to produce valid results.

The statistical generalizability of our study to a larger population is limited by our small sample size of $n=9$. We made use of the limited sample size, which was due to constraints out of our control, by triangulating with, and focusing on, the qualitative data gathered through the interviews and the study diary. We, therefore, gathered an in-depth insight into the experience rather than just relying on a statistical evaluation of the resulting artifacts. We present both types of evaluation, which in many instances complement and support each other.

Further, the theoretical generalizability of our results to situations outside of our test-environment is impacted by the following considerations.

- Students' lack of experience in domain modeling calls the transferability of the results to professional requirements engineers in question.
- The scope of the project with data sources from four stakeholders is very small compared to most real-life projects. And the types of data were restricted to interview transcriptions.

The lack of experience was also a positive in that it allowed us to better analyze the learning curve for novice analysts. The psychological effects of participants' self-assessment varying independently of modeling success can be partly attributed to lack of experience and was mitigated by having a particular focus on changes over time and aggregate values across each group.

Using students in software engineering experiments, even though it is a commonly done, has been criticized for making the transferability of the results to professionals inappropriate, and thus lowering the external validity of the research [16, 42]. However, as Tichy points out, student experiments should be considered a prerequisite to performing experiments with professionals. They can also be indicative of a trend, regardless of the magnitude of the observed differences which is most likely to differ with professionals [46].

As pointed out by one of the participants, a considerable limitation is also related to the tool support for both methods. They are similar in that they are tool-supported, but both solutions had tooling issues. While the individual components in the control group with Mendeley and MindMeister mostly worked reliably since they are commercial products, they were not designed for the specific use case, and the Chrome plug-in connecting the two had its own issues. QDAcity forgoes some of the problems that the re-purposing of existing tools brings with it, but in itself is not yet a commercially viable product, and participants sometimes struggled with an unpolished user experience and some bugs.

A natural limitation of any experiment of this type is the artificial nature of the setup, which was further restricted to include only the analysis phase of the method, and excluded data gathering. We see the need for further validation including the full process of data gathering and analysis in a case study context. Besides being able to cover the complete process, the more natural setting would yield higher external validity than our experiment in an abstracted environment. In a case study setup, however, a detailed comparison with a different method across multiple independent analysts using the same data would not have been possible. Our experiment was able to provide this comparison.

8 Conclusion

We present an experimental evaluation of our novel method for conceptual domain modeling, QDAcity-RE, comparing it to a control group using a method for conceptual modeling with a similar level of traceability.

Our results show that both groups experienced value in being able to link quotes in expert interviews with model elements either through a code system or a mind map. Both groups also shared the experience of the detailed analysis and documentation requirements requiring significant effort. However, despite this feeling being prevalent in both groups, the quantitative evaluation shows a significant difference in the time expended for the analysis. The average participant in the QDAcity group was almost twice as fast than in the control group, needing only 56.79% of the time the average participant in the control group needed.

We explain part of this discrepancy through the tighter integration of reading and modeling in one tool. Both groups described significant tooling problems, however, the combination of different tools in the control group was particularly mentioned as burdensome and participants were developing strategies for switching less between the texts and the models.

Despite the significantly lower effort, the average recall and f-measure were significantly higher in the QDAcity

group. Precision was, on average, slightly lower. The difference for recall and f-measure was evaluated as being statistically significant with an α of 0.05. The difference for precision was evaluated as not statistically significant.

The models created with the QDAcity tool using the QDAcity-RE method were generally more detailed, manifesting in a larger number of concepts, and leading to the increased recall measurement. Some of the detail in the number of concepts, however, was arguably due to inexperience with conceptual modeling with UML, because even minor attributes of concepts were sometimes modeled as their own concept, for example “Location” or “Time”.

The number of nine participants in our experiment makes generalization to a larger population difficult. But we gathered in-depth data, on the experience of the participants which revealed some significantly differing themes in the 40 semi-structured interviews we analyzed, as well as in the study diary the participants kept. Also, the characteristics of the models in the QDAcity group being more detailed throughout the experiment population were significant.

Even though the learning curve of getting acquainted with and applying the method was frequently mentioned throughout the study, the changes in sentiment over time even within a short time span of four weeks were encouraging. Still, we conclude a confirmation of our previously held assumption that both our method and the control group method are more suited to projects that are longer running or require a lot of collaboration in which a more detailed and traceable documentation is particularly helpful.

Acknowledgements We would like to thank our colleague Dr. Maximilian Capraro for guidance during the execution of the experiments.

Funding Open Access funding enabled and organized by Projekt DEAL. This project was funded by DFG Grant RI 2147/7-1.

Availability of data and material The full interview transcripts will not be public to protect the anonymity of our participants.

Code availability Code of all tools used in this study are proprietary.

Declarations

Ethics approval Not applicable.

Consent to participate All participants signed a consent form prior to their participation. We’re happy to provide a copy of the form without the names of any of our participants upon request.

Consent for publication The consent form signed by all participants explicitly stated publication of aggregated statistics and quotes from the interviews, which would be anonymized by a participant ID.

Conflict of interest The authors declare that they have no conflicts of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Adler A, Gujar A, Harrison BL, O’hara K, Sellen A (1998) A diary study of work-related reading: design implications for digital reading devices, pp 241–248
2. Begel A, Simon B (2008) Novice software developers, all over again. In: Proceedings of the fourth international workshop on computing education research, pp 3–14
3. Berry DM, Godfrey MW, Holt R, Kapsner CJ, Ramos I (2013) Requirements specifications and recovered architectures as grounded theories
4. Boehm BW (1984) Verifying and validating software requirements and design specifications. *IEEE Softw* 1(1):75
5. Byrd TA, Cossick KL, Zmud RW (1992) A synthesis of research on requirements analysis and knowledge acquisition techniques. *MIS Q* 1992:117–138
6. Carvalho L, Scott L, Jeffery R (2005) An exploratory study into the use of qualitative research methods in descriptive process modelling. *Inf Softw Technol* 47(2):113–127
7. Chakraborty S, Dehlinger J (2009) Applying the grounded theory method to derive enterprise system requirements. In: 10th ACIS international conference on Software engineering, artificial intelligences, networking, and parallel/distributed computing, SNP’09, 2009. IEEE Computer Society, pp 333–338
8. Chakraborty S, Rosenkranz C, Dehlinger J (2015) Getting to the shalls: facilitating sensemaking in requirements engineering. *ACM Trans Manag Inf Syst* 5(3):14
9. Chen F, Power N, Collins J, Ishikawa F (2019) Contemporary requirements challenges and issues: an empirical study in 11 organizations. In: Proceedings of the 34th ACM/SIGAPP symposium on applied computing, pp 1592–1599
10. Chitchyan R, Bird C (2020) Theory as a source of software requirements. In: 28th IEEE International Requirements Engineering Conference. IEEE Computer Society
11. Chitchyan R, Shah P, Bird C (2019) Eliciting requirements for demand response service design to households: a pilot study. In: Proceedings of the evaluation and assessment on software engineering, pp 299–304
12. Corbin J, Strauss A (2014) Basics of qualitative research: techniques and procedures for developing grounded theory. Sage, Thousand Oaks
13. Davidson JL, Mannan UA, Naik R, Dua I, Jensen C (2014) Older adults and free/open source software: a diary study of first-time contributors. In: Proceedings of the international symposium on open collaboration, pp 1–10
14. Díaz O (2017) Strategic reading & conceptual modeling. In: Conceptual modeling perspectives. Springer, pp 41–53
15. Díaz O, Contell JP, Venable JR (2017) Strategic reading in design science: let root-cause analysis guide your readings. In: Maedche

- A, vom Brocke J, Hevner A (eds) *Designing the digital transformation*. Springer, Cham, pp 231–246
16. Easterbrook S, Singer J, Storey MA, Damian D (2008) Selecting empirical methods for software engineering research. In: *Guide to advanced empirical software engineering*. Springer, pp 285–311
 17. Faily S, Flechais I (2011) Persona cases: a technique for grounding personas. In: *Proceedings of the SIGCHI conference on human factors in computing systems*, pp 2267–2270
 18. Faily S, Iacob C, Ali R, Ki-Aries D (2020) Identifying implicit vulnerabilities through personas as goal models. *arXiv preprint arXiv:2008.04773*
 19. Fléchais I (2005) *Designing secure and usable systems*. PhD thesis. University College London
 20. Francis JJ, Johnston M, Robertson C, Glidewell L, Entwistle V, Eccles MP, Grimshaw JM (2010) What is an adequate sample size? Operationalising data saturation for theory-based interview studies. *Psychol Health* 25(10):1229–1245
 21. Gidey HK, Marmsoler D, Eckhardt J (2017) Grounded architectures: using grounded theory for the design of software architectures. In: *2017 IEEE international conference on software architecture workshops (ICSAW)*. IEEE, pp 141–148
 22. Glaser BG, Strauss AL (2009) *The discovery of grounded theory: strategies for qualitative research*. Transaction Publishers, Piscataway
 23. Gotel O, Finkelstein A (1994) An analysis of the requirements traceability problem. In: *International conference on requirements engineering*. IEEE Computer Society, pp 94–101
 24. Guba EG (1981) Criteria for assessing the trustworthiness of naturalistic inquiries. *Ectj* 29(2):75
 25. Guion LA, Diehl D, McDonald D (2002) *Triangulation: establishing the validity of qualitative studies*. Tech. rep., University of Florida Cooperative Extension Service, Institute of Food and Agricultural Sciences, EDIS
 26. Halaweh M (2012) Application of grounded theory method in information systems research: methodological and practical issues. *Rev Bus Inf Syst* 16(1):27
 27. Halaweh M (2012) Using grounded theory as a method for system requirements analysis. *J Inf Syst Technol Manag* 9(1):23–38
 28. Hughes J, Wood-Harper T (1999) Systems development as a research act. *J Inf Technol* 14(1):83–94
 29. Huzar Z, Kuzniarz L, Reggio G, Sourrouille JL (2004) Consistency problems in UML-based software development. In: *International conference on the unified modeling language*. Springer, pp 1–12
 30. Kaufmann A, Riehle D (2019) The qdacity-re method for structural domain modeling using qualitative data analysis. *Requir Eng* 24(1):85–102. <https://doi.org/10.1007/s00766-017-0284-8>
 31. Kruskal WH, Wallis WA (1952) Use of ranks in one-criterion variance analysis. *J Am Stat Assoc* 47(260):583–621
 32. Lindland OI, Sindre G, Solvberg A (1994) Understanding quality in conceptual modeling. *IEEE Softw* 11(2):42–49
 33. McEwan EK (2004) *Seven strategies of highly effective readers: using cognitive research to boost K-8 achievement*. Corwin Press, Thousand Oaks
 34. M'manga A, Faily S, McAlaney J, Kadobayashi Y, Miyamoto D (2018) Qualitative adaptation: informing design for risk-based decision making
 35. Mylopoulos J, Chung L, Nixon B (1992) Representing and using nonfunctional requirements: a process-oriented approach. *IEEE Trans Softw Eng* 18(6):483–497
 36. Nor NFM, Azman H, Hamat A (2013) Investigating students' use of online annotation tool in an online reading environment. *3L: Language, Linguistics, Literature* 19(3)
 37. O'hara K, Sellen A, (1997) A comparison of reading paper and on-line documents. *CHI* 97:335–342
 38. Pidgeon NF, Turner BA, Blockley DI (1991) The use of grounded theory for conceptual analysis in knowledge elicitation. *Int J Man-Mach Stud* 35(2):151–173
 39. Porter-O'Donnell C (2004) Beyond the yellow highlighter: teaching annotation skills to improve reading comprehension. *Engl J* 82–89
 40. Rashid A, Naqvi SAA, Ramdhany R, Edwards M, Chitchyan R, Babar MA (2016) Discovering “unknown known” security requirements. In: *Proceedings of the 38th international conference on software engineering*, pp 866–876
 41. Sharma R, Biswas KK (2015) Functional requirements categorization grounded theory approach. In: *2015 international conference on evaluation of novel approaches to software engineering (ENASE)*. IEEE Computer Society, pp 301–307
 42. Siegmund J, Siegmund N, Apel S (2015) Views on internal and external validity in empirical software engineering. In: *Proceedings of the 37th international conference on software engineering-volume 1*. IEEE Computer Society, pp 9–19
 43. Spall S (1998) Peer debriefing in qualitative research: emerging operational models. *Qual Inqu* 4(2):280–292. <https://doi.org/10.1177/107780049800400208>
 44. Thew S, Sutcliffe A (2018) Value-based requirements engineering: method and experience. *Requir Eng* 23(4):443–464
 45. Thomas K, Bandara AK, Price BA, Nuseibeh B (2014) Distilling privacy requirements for mobile applications. In: *Proceedings of the 36th international conference on software engineering*. ACM, pp 871–882
 46. Tichy WF (2000) Hints for reviewing empirical work in software engineering. *Empir Softw Eng* 5(4):309–312
 47. Torkar R, Gorschek T, Feldt R, Svahnberg M, Uzair Akbar R, Kamran K (2012) Requirements traceability a systematic review and industry case study. *Int J Softw Eng Knowl Eng* 22(3):385–433
 48. Van Rijsbergen CJ (1979) *Information retrieval*, 2nd edn. Butterworth-Heinemann, Newton
 49. Wilcoxon F (1945) Individual comparisons by ranking methods. *Biometrics Bull* 1(6):80–83
 50. Würfel D, Lutz R, Diehl S (2016) Grounded requirements engineering: an approach to use case driven requirements engineering. *J Syst Softw* 117:645–657
 51. Zowghi D, Gervasi V (2003) On the interplay between consistency, completeness, and correctness in requirements evolution. *Inf Softw Technol* 45(14):993–1009