Domain Modeling Using Qualitative Data Analysis

Domänenmodellierung Mittels Qualitativer Datenanalyse

Der Technischen Fakultät
der Friedrich-Alexander-University
Zur
Erlangung des Doktorgrades
Doktor-Ingenieur (Dr.Ing.)

Vorgelegt von
Andreas Kaufmann
Aus Neuss
Abstract

The creation of domain models from qualitative input relies heavily on experience. An uncodified ad-hoc modeling process is still common and leads to poor documentation of the requirements analysis.

In this thesis, we present a novel method for domain analysis based on qualitative data analysis (QDA). The method helps identifying inconsistencies, ensures a high degree of completeness, and inherently provides traceability from analysis results back to stakeholder input.

In our approach, the QDAcity-RE method, the research process of theory building facilitates domain analysis within the requirements elicitation phase of a software development project.

We show how an iterative process of concurrent data collection and analysis can be applied to requirements engineering (RE), including open, axial, and selective coding of qualitative data.

The traceability of domain model elements back to original statements by stakeholders, generated by our method, does not have to be created and maintained separately after the fact. The traces are documented in an analysis artifact called the code system, which evolves iteratively with the analysis process. The code system can act as a universal model from which a variety of artifacts can be derived, describing both behavioral and structural aspects. This thesis focuses on the creation of conceptual domain models using QDAcity-RE, but peripherally also demonstrates this capability through the generation of behavioral models and a software requirements specification.

We applied and evaluated our method for domain modeling in four exploratory projects in the domains of medical imaging diagnostics, railway systems, HR development, and qualitative research.

We show that by applying QDA to domain analysis, structural elements and relationships needed to derive a UML class diagram can be extracted from a code system based on interviews with domain experts. Constant comparison and theoretical sampling assist in integrating differing domain descriptions into an abstract model.

While the analysis process still requires interpretations and modeling decisions, our method provides more guidance than existing domain analysis approaches and a thorough documentation of these decisions. In addition, codes and memos ensure traceability between the original data and the derived model and assist in connecting several RE artifacts, ensuring a high degree of inter-model consistency.

We validated our claim that QDAcity-RE helps an analyst gain a deeper understanding of a problem domain through a controlled experiment.

In dieser Dissertation präsentieren wir eine neuartige Methode zur Domänenmodellierung basierend auf qualitativer Datenanalyse (QDA). Die Methode unterstützt die Identifikation von Inkonsistenzen, sichert einen hohen Grad an Vollständigkeit und erzeugt die Rückverfolgbarkeit (traceability) von Ergebnissen der Analyse zu Aussagen von Stakeholdern.

In unserem Ansatz, der QDAcity-RE Methode, ermöglichen Prozesse der theoriebildenden Forschung die Domänenanalyse in der Phase der Anforderungserhebung zur Softwareentwicklung.

Wir zeigen wie ein iterativer Prozess aus gleichzeitiger Datensammlung und Analyse im Requirements Engineering (RE) angewendet werden kann, welcher die Schritte des offenen, axialen, und selektivem Codieren beinhaltet.


Wir wendeten unsere Methode zur Domänenmodellierung in vier explorativen Studien an und evaluierten sie fortlaufend. Die Domänen dieser Studien waren medizinische Bildgebung, Eisenbahnsysteme, Personalentwicklung und qualitative Forschung.

Wir zeigen, dass durch die Anwendung von QDA zur Domänenmodellierung strukturelle Eigenschaften und deren Beziehungen aus einer Domäne, welche für die Erstellung eines UML Klassendiagramms benötigt werden, aus einem Codesystem abgeleitet werden können, welches auf Experteninterviews beruht.

Die Methode des ständigen Vergleichs (constant comparison method) und theoretisches Sampling helfen verschiedene Beschreibungen der Domäne in einem abstrakten Modell zu vereinigen.

Obwohl die Analyse nach wie vor Interpretationen und Modellierungsentscheidungen erfordert, so ermöglicht unsere Methode ein strukturierteres Vorgehen als existierende Ansätze.
zur Domänenmodellierung sowie eine detaillierte Dokumentation der Modellierungsentsehei-
dungen. Zusätzlich ermöglichen Codes und Memos eine Rückverfolgbarkeit zwischen Stake-
holder Aussagen und den Artefakten der Analyse. Diese Verbindungen (Traces) zwischen
den Artefakten sichern ein hohes Maß an Konsistenz zwischen den Modellen.

Durch ein kontrolliertes Experiment validierten wir unsere Annahme, dass QDAcity-RE
einem Analysten dabei hilft ein tiefes Verständnis für eine Domäne zu gewinnen.
Contents

1 INTRODUCTION ........................................... i
  1.1 Motivation and Original Idea .......................... 8
    1.1.1 Main Concepts ................................... 8
    1.1.2 Expected Benefits ................................ 10
  1.2 Structure of the Thesis ................................ 12

2 RELATED WORK ............. 13
  2.1 QDA in Requirements Engineering ..................... 13
  2.2 Quality Criteria of Domain Models ...................... 20
    2.2.1 Consistency ................................... 20
    2.2.2 Completeness ................................... 22
  2.3 Traceability ........................................ 23
    2.3.1 Pre-RS Traceability ............................... 25

3 EXPLORATORY CASES .......... 29
  3.1 Domain Modeling for Medical Imaging Diagnostics .......... 33
  3.2 Domain Modeling for the openETCS Toolchain .............. 35
    3.2.1 Data Sources ................................... 35
    3.2.2 Data Analysis ................................... 36
    3.2.3 Evaluation of openETCS Study ...................... 37
  3.3 Domain Modeling for Human Resource Development .......... 37
    3.3.1 Project Pilot: Industry Study ...................... 38
    3.3.2 Data Sources ................................... 40
    3.3.3 Data Analysis ................................... 42
    3.3.4 Evaluation of HR Study ............................ 43
  3.4 Domain Modeling for Qualitative Research ................. 46
    3.4.1 Data Sources ................................... 47
    3.4.2 Data Analysis ................................... 47
    3.4.3 Evaluation of Qualitative Research Study .......... 48
  3.5 Limitations of Exploratory Cases ......................... 51
    3.5.1 Application Domain ................................. 51
    3.5.2 Sampling and Saturation .......................... 52
    3.5.3 Lack of Tool Support .............................. 52
  3.6 Evaluation of Exploratory Studies ......................... 52
  3.7 Conclusion .......................................... 55
4 The QDAcity-RE Method
4.1 Method Overview ........................................ 57
4.2 Method Context ........................................ 58
4.3 The Code System ....................................... 59
4.4 The QDAcity-RE Process ............................... 61
  4.4.1 Stakeholder Sampling ............................ 61
  4.4.2 Data Gathering .................................. 62
  4.4.3 Data Analysis .................................. 63
  4.4.4 Iterative Refinement ............................ 66
4.5 Codesystem Language .................................. 68
  4.5.1 Mapping to Conceptual Model ................... 73

5 QDAcity: A QDA Tool Supporting QDAcity-RE ............ 75
5.1 Motivation .............................................. 75
5.2 Architecture and Design ............................... 76
  5.2.1 GAE Backend Service ........................... 77
  5.2.2 Real Time Collaboration Service ................. 78
  5.2.3 Auth Proxy Service ................................ 80
  5.2.4 Frontend Client .................................. 82
5.3 Offline Experience ..................................... 85
  5.3.1 Conflict Resolution ............................... 88
5.4 Agreement and Saturation Metrics ....................... 93
  5.4.1 Inter-Rater Agreement ............................ 93
  5.4.2 Saturation ....................................... 100
5.5 CD Pipeline ............................................ 104

6 Validation .................................................. 107
6.1 Experiment Design .................................... 108
  6.1.1 Data on Target Domain ......................... 110
  6.1.2 Participant Sampling .............................. 110
  6.1.3 Control Group .................................... 113
  6.1.4 QDAcity Group .................................... 116
  6.1.5 Evaluation Methods ............................... 117
6.2 Artifact Evaluation ..................................... 119
  6.2.1 Evaluation Method ............................... 119
  6.2.2 Artifact Evaluation Results ..................... 121
6.3 Process Evaluation ..................................... 125
  6.3.1 Evaluation Method ............................... 125
  6.3.2 Process Evaluation Results ...................... 129
6.4 Discussion ............................................. 137
6.5 Limitations ............................................. 138
7 Conclusion

7.1 Demonstration of Research Rigor

7.1.1 Naturalistic Paradigm

7.1.2 Rationalistic Paradigm

7.1.3 Analysis of Rigor in Exploratory Cases

7.1.4 Analysis of Rigor in Experiments (Rationalistic)

7.1.5 Analysis of Rigor in Experiments (Naturalistic)

7.1.6 Conclusion

7.2 Future Research

7.2.1 Extension to Other Types of Artifacts

7.2.2 Validate Inter-Model Consistency

7.2.3 Impact on Change Management

7.2.4 Supporting QDA Through NLP

7.2.5 Investigation of Alternatives to the Coding Paradigm

7.2.6 Quality Metrics

7.2.7 Project Specific CSL

References

Appendix A Experiment Artifacts

A.1 Informed Consent

A.2 Study Diary

Appendix B Conceptual Models

Appendix C Codebook of Validation Experiment
### Listing of figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>The QDAcity-RE Process for Structural Domain Modeling</td>
<td>59</td>
</tr>
<tr>
<td>4.2</td>
<td>Codesystem Example</td>
<td>60</td>
</tr>
<tr>
<td>4.3</td>
<td>Code System Language (CSL)</td>
<td>70</td>
</tr>
<tr>
<td>5.1</td>
<td>QDAcity deployment diagram</td>
<td>77</td>
</tr>
<tr>
<td>5.2</td>
<td>Successful Authentication in QDAcity With Federated Identity Provider</td>
<td>81</td>
</tr>
<tr>
<td>5.3</td>
<td>UML Editor Components</td>
<td>83</td>
</tr>
<tr>
<td>5.4</td>
<td>Communication for Adding a Code as New Class</td>
<td>85</td>
</tr>
<tr>
<td>5.5</td>
<td>Components Involved in Conflict Resolution</td>
<td>89</td>
</tr>
<tr>
<td>5.6</td>
<td>Three-Way Merge</td>
<td>90</td>
</tr>
<tr>
<td>5.7</td>
<td>Selected and Relevant Items</td>
<td>98</td>
</tr>
<tr>
<td>5.8</td>
<td>The QDAcity.com CD Pipeline</td>
<td>104</td>
</tr>
<tr>
<td>6.1</td>
<td>Overview Schedule of Each Iteration</td>
<td>109</td>
</tr>
<tr>
<td>6.2</td>
<td>Self-Assessed Competency of Participants</td>
<td>111</td>
</tr>
<tr>
<td>6.3</td>
<td>Purpose Pipe and Annotation Pipe</td>
<td>115</td>
</tr>
<tr>
<td>6.4</td>
<td>Evaluation overview</td>
<td>118</td>
</tr>
<tr>
<td>6.5</td>
<td>Boxplot of Recall, Precision and F-Measure by Group</td>
<td>123</td>
</tr>
<tr>
<td>6.6</td>
<td>Number of Codes and Codings</td>
<td>128</td>
</tr>
<tr>
<td>6.7</td>
<td>Self Assessment of Session Success</td>
<td>130</td>
</tr>
<tr>
<td>6.8</td>
<td>Self Assessment of Confidence in Results</td>
<td>130</td>
</tr>
<tr>
<td>6.9</td>
<td>Average Length of Work Session</td>
<td>132</td>
</tr>
<tr>
<td>6.10</td>
<td>Total Time Spent by Participants</td>
<td>132</td>
</tr>
<tr>
<td>6.11</td>
<td>Saturation Over Time</td>
<td>136</td>
</tr>
<tr>
<td>7.1</td>
<td>Evaluation of Research Rigor - Overview</td>
<td>146</td>
</tr>
<tr>
<td>7.2</td>
<td>Measures Ensuring Rigor in Theory Building (naturalistic)</td>
<td>152</td>
</tr>
<tr>
<td>7.3</td>
<td>Measures Ensuring Rigor in Theory Validation (rationalistic)</td>
<td>157</td>
</tr>
<tr>
<td>7.4</td>
<td>Measures Ensuring Rigor in Theory Validation (naturalistic)</td>
<td>171</td>
</tr>
<tr>
<td>B.1</td>
<td>Domain Model for HR-Development Exploratory Study (Expert Solution)</td>
<td>210</td>
</tr>
<tr>
<td>B.2</td>
<td>Conceptual Model of Participant A1</td>
<td>211</td>
</tr>
<tr>
<td>B.3</td>
<td>Conceptual Model of Participant A2</td>
<td>212</td>
</tr>
<tr>
<td>B.4</td>
<td>Conceptual Model of Participant A3</td>
<td>213</td>
</tr>
<tr>
<td>B.5</td>
<td>Conceptual Model of Participant A4</td>
<td>214</td>
</tr>
</tbody>
</table>
Listing of tables

3.2 Overview of Exploratory Studies. ........................................ 30
3.1 Coded Data. ............................................................... 30
3.3 Evaluation Model. .......................................................... 32
3.4 Evaluation of Domain Model by Domain Experts. ....................... 44
3.5 Evaluation of Domain Model by Domain Experts. ....................... 50
3.6 Study Evaluation. .......................................................... 53

4.1 Application Rules for CSL ............................................... 72
4.2 Mapping from CSL to UML Class Diagram. ............................. 74

5.1 Levenshtein Matrix Example (Initialized) .......................... 92
5.2 Levenshtein Matrix Example (Fully Populated) ...................... 92
5.3 Reliability Matrix ....................................................... 97
5.4 Matrix of Observed Coincidences .................................... 97

6.1 Demographic Averages by Group .................................... 112
6.2 Artifact Comparison with Expert Solution - Control Group ........ 121
6.3 Artifact Comparison with Expert Solution - QDAcity Group ........ 121
6.4 Descriptive Statistics .................................................. 122
6.5 Ranking for H-Test ...................................................... 124
6.6 Kruskal-Wallis H-Test results ....................................... 125

7.1 Evaluation Criteria by Research Paradigm, Adapted From [79] .... 145
7.2 Threats to Statistical Conclusion Validity ........................... 158
7.3 Threats to Internal Validity ............................................. 161
7.4 Threats to Construct Validity ......................................... 164
7.5 Threats to External Validity .......................................... 167

C.1 Color Coding of Code System Hierarchy .......................... 221
C.2 Codebook ................................................................. 222
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANOVA</td>
<td>analysis of variance</td>
</tr>
<tr>
<td>API</td>
<td>application programming interface</td>
</tr>
<tr>
<td>BPMN</td>
<td>business process model and notation</td>
</tr>
<tr>
<td>CAQDAS</td>
<td>computer aided qualitative data analysis software</td>
</tr>
<tr>
<td>CLI</td>
<td>command line interface</td>
</tr>
<tr>
<td>CT</td>
<td>computer tomography</td>
</tr>
<tr>
<td>CD</td>
<td>continuous deployment</td>
</tr>
<tr>
<td>CRUD</td>
<td>create read update delete</td>
</tr>
<tr>
<td>CSL</td>
<td>code system language</td>
</tr>
<tr>
<td>CSS</td>
<td>cascading style sheets</td>
</tr>
<tr>
<td>DB</td>
<td>database</td>
</tr>
<tr>
<td>DSL</td>
<td>domain specific language</td>
</tr>
<tr>
<td>ES</td>
<td>ECMAScript</td>
</tr>
<tr>
<td>GT</td>
<td>grounded theory</td>
</tr>
<tr>
<td>GAE</td>
<td>google app engine</td>
</tr>
<tr>
<td>GCP</td>
<td>google cloud platform</td>
</tr>
<tr>
<td>HR</td>
<td>human resources</td>
</tr>
<tr>
<td>MRI</td>
<td>magnetic resonance imaging</td>
</tr>
<tr>
<td>HTML</td>
<td>hypertext markup language</td>
</tr>
<tr>
<td>HTTPS</td>
<td>hyper text transfer protocol secure</td>
</tr>
<tr>
<td>JS</td>
<td>javascript</td>
</tr>
<tr>
<td>JSON</td>
<td>javascript object notation</td>
</tr>
<tr>
<td>JSX</td>
<td>javascript XML</td>
</tr>
<tr>
<td>JWT</td>
<td>JSON web token</td>
</tr>
<tr>
<td>NFR</td>
<td>non-functional requirements</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>NL</td>
<td>natural language</td>
</tr>
<tr>
<td>NLP</td>
<td>natural language processing</td>
</tr>
<tr>
<td>NPM</td>
<td>node package manager</td>
</tr>
<tr>
<td>NYT</td>
<td>nailing your thesis</td>
</tr>
<tr>
<td>OCL</td>
<td>Object Constraint Language</td>
</tr>
<tr>
<td>OOAD</td>
<td>object-oriented analysis and design</td>
</tr>
<tr>
<td>post-RS</td>
<td>post-requirements-specification</td>
</tr>
<tr>
<td>pre-RS</td>
<td>pre-requirements-specification</td>
</tr>
<tr>
<td>PWA</td>
<td>progressive web app</td>
</tr>
<tr>
<td>RE</td>
<td>requirements engineering</td>
</tr>
<tr>
<td>RCA</td>
<td>root cause analysis</td>
</tr>
<tr>
<td>RS</td>
<td>requirements specification</td>
</tr>
<tr>
<td>RT</td>
<td>requirements traceability</td>
</tr>
<tr>
<td>RTCS</td>
<td>real-time collaboration service</td>
</tr>
<tr>
<td>SW</td>
<td>service worker</td>
</tr>
<tr>
<td>UI</td>
<td>user interface</td>
</tr>
<tr>
<td>UML</td>
<td>unified modeling language</td>
</tr>
<tr>
<td>QDA</td>
<td>qualitative data analysis</td>
</tr>
<tr>
<td>XML</td>
<td>extensible markup language</td>
</tr>
<tr>
<td>YAML</td>
<td>YAML ain’t markup language</td>
</tr>
</tbody>
</table>
Acknowledgments

During my journey to complete this work I was supported by many people to whom I want to express my deepest thanks.

My supervisor, Dirk Riehle, introduced me to this exciting topic and provided valuable feedback and advice throughout my research. I especially enjoyed our many discussions on research methods and epistemology. His precise arguments often inspired new thinking, and an engaged back and forth towards a deeper understanding.

I first encountered Ann Barcomb as a supervisor while writing my Master’s thesis, and the work we did together inspired me to pursue further research leading to this thesis. Co-authoring papers with her is a pleasure, and her guidance helped me to significantly improve my research design for the validation of this thesis.

My colleagues at the research group provided me with endless emotional support and constructive discussions. Thank you, Andi, Elcin, Fariba, Georg, Hannes, Julia, Max, Michi, Nick, and Sebastian. Special thanks to Max and Nick. Our weekly dissertation workshop, and your drive to finish, were excellent motivation.

Thanks again to Ann, Dirk, Elcin, Fiona, Julia, Max, and Nick for providing feedback on parts of this thesis, to help improve its presentation.

Finally, I want to thank my parents, Brunhilde and Bernd, for supporting my education and fostering independent, critical, and inquisitive thinking.
Related Publications

Within this thesis, except for the declarations in the preamble, the third person plural is used to acknowledge the support I received in performing the research leading to this thesis.

I co-authored the publications on the following page with my colleagues, in which we published parts of the research results that are part of this thesis.

For three of the listed articles, I was the leading author and main contributor. The research design, in all cases, was supported by feedback from my doctoral supervisor, and co-author, Dirk Riehle, who also contributed to improving the presentation of all articles through written feedback.

The research design for the yet unpublished paper was further improved by Julia Krause and Nikolay Harutyunyan through participation in peer debriefing sessions, and by all co-authors through suggestions within an informal collegial exchange. The execution of the research was supported through investigator triangulation performed by Julia Krause, Nikolay Harutyunyan, and Ann Barcomb. All co-authors also contributed with feedback to improve the presentation of the paper.
Kaufmann and Riehle [106] This article describes the initial idea, which was developed in collaboration with my doctoral supervisor, Dirk Riehle. There is conceptual overlap with this thesis regarding the motivation and related work, but none of the text was reused in this thesis.

Kaufmann and Riehle [107] This article describes the method developed as the main result of the research work leading to this thesis. Consequently, the method description in chapter 4 and the description of the exploratory studies in which it was iteratively developed and evaluated in chapter 3 partially rely on the same textual description of the research.

Krause et al. [111] This technical report, led by my colleague, Julia Krause, presents a systematic literature review on the topic of pre-RS traceability. Some of the learnings of this technical report, where I am second author, helped improve the related work chapter 2. None of the writing of this technical report was used in this thesis.

Kaufmann et al. [submitted] This article, titled “A validation of QDA-based domain modeling using QDAcity-RE”, describes the theory validation research performed within the context of this thesis. It was co-authored with my colleagues Julia Krause, Nikolay Harutyunyan, Ann Barcomb, and my doctoral supervisor Dirk Riehle. The description of the research presented in chapter 6 was, in parts, reused from this publication, which, at the time of this thesis’ submission, is under peer-review.

Krause et al. [submitted] This article, submitted for publication, titled “A Systematic Literature Review of Pre-Requirements Specification Traceability”, describes the systematic literature on pre-RS traceability, led by my colleague Julia Krause. I am second author and was involved in the research execution through peer-debriefing and improved the presentation. None of the writing of this paper was used in this thesis.
The success of a software development project is highly dependent on the quality of its requirements engineering (RE) process \cite{207, 53, 204}. Poor understanding of requirements negatively impacts quality, lead time, and cost of the software product \cite{100}. The later within the development process problems with the quality of a requirements specification (RS) are identified, the more expensive it is to rectify the issues \cite{62}.

The quality of a requirements specification mainly depends on the experience of the analyst and the understanding of the context in which the desired system is supposed to operate (the problem domain). To establish a good understanding of the problem domain, the analyst may create a domain model as part of the analysis \cite{153, 97, 117}. A domain model is an abstract representation of the relevant entities within an area of interest, their attributes, relationships, and behavior.

The quality of a requirements specification may be regarded in two dimensions. The analyst has to make sure that (1) they elicit the right requirements (satisfying a stakeholder’s
need), and that (2) that the requirements have to be documented the right way (unambiguous, traceable, etc.).

Methods and metrics for measuring artifact quality can help ensure that the documentation is of high quality, addressing the second goal. Both dimensions can also be improved through training and experience. However, to determine the requirements which satisfy the stakeholders’ needs, the analyst has to gain a deep understanding of the problem domain [46, 90, 163]. The analyst has to understand the stakeholders’ goals, assumptions, opinions, and desires [21, 55].

Therefore, data elicited from different stakeholders needs to be conceptualized to derive an abstract description of the world in which an envisioned system will operate [145].

A common way of creating and documenting this abstract description is through conceptual domain modeling [153, 97]. Besides gaining a deeper understanding of the problem domain during its creation, one pivotal purpose of conceptual models is the facilitation of a shared understanding between all stakeholders. For this purpose, the domain model can be regarded as a visual dictionary [117], which is a more elaborate extension of a glossary. The modeling of domain knowledge, both structural and behavioral, is also sometimes referred to as information modeling [184, 183, 84].

A domain model may also form the foundation 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 latent experience of the analyst. Consequently, the consistency and completeness of the requirements, which affect implementation [31], may be impacted.

Domain models must correctly represent the reality of the domain and be easy to understand from the stakeholder’s perspective [18, 45]. The degree to which this goal can be achieved depends heavily on the modeling experience of the analyst [155, 94].

In scientific research, similar challenges to those encountered for domain analysis arise. These similarities may be categorized as follows:
Dealing with a large amount of unstructured qualitative data as input

Triangulating results using multiple types of input data (interviews, documents, observation notes, etc.)

Uncovering both explicit and tacit knowledge from participants

Discovering and resolving inconsistencies in the data

These challenges are addressed by using codified methods for qualitative data analysis (QDA). QDA methods focus on extracting the relevant information from qualitative data, interpreting the data, and abstracting from it. QDA is employed in theory building research to study a wide range of phenomena through the gathering and interpretation of qualitative data. The process leading to the resulting theory ensures thorough documentation of the analysis process. The documentation is facilitated through memos written by the researcher. The adherence to one of the established methods and their documentation through an audit trail [164] is one of the main pillars of lending trustworthiness [79, 120] to the resulting artifact or theory.

We equate the process of theory building to the domain analysis process. Nuseibeh and Easterbrook state that “we can compare the problem of validating requirements with the problem of validating scientific knowledge” [145]. We take this comparison one step further by proposing that not only the validation of knowledge but also its elicitation can follow similar procedures.

The inspiration drawn from the similarities between qualitative theory building research and domain modeling led us to our overarching research question:

*How can research practices for qualitative data analysis be effectively applied to conceptual domain modeling?*

The domains under study both in theory building and domain modeling are highly complex and the gathered data might therefore be unstructured, inconsistent, and incomplete
Therefore, both theory building research and domain modeling require a systematic method for extracting relevant information from qualitative data. This method should guide the analyst in structuring the data and resolving inconsistencies to arrive at a coherent theory or domain model, respectively. The context in which requirements engineering takes place is usually a socio-technical work system, in which humans interact with one another and with systems. Understanding how these social units interact and deal with a phenomenon can be viewed as a theory building task.

Since, unlike in qualitative research, the extraction of domain knowledge from domain experts in RE frequently follows an unstructured process, this leads to the results lacking empirical evidence.

By drawing on theory building in qualitative research as a form of knowledge generation and applying these concepts to domain analysis we expect to improve the following aspects of RE.

• Improved documentation of the analysis process.

• Stakeholders frequently have different goals, leading to conflicting requirements. Managing these conflicts is a challenge and only one solution can be documented in the final artifacts. Within a new intermediate artifact, however, all resolved and unresolved conflicts can be documented.

• While methods and notations for a multitude of models have matured, maintaining inter-model consistency between these models is still a challenge. Our method interconnects several final model types through a new intermediate artifact.

• Organizational politics and other social aspects that are uncovered during the analysis can be documented in the same artifact as the analysis result. In fact, the methods we adapt from the context of qualitative research have been used to study social constructs for decades. The appropriateness of these methods for investigating social aspects has been well established.
Further challenges such as reusability of requirements artifacts and management of traces are directly impacted by improved documentation as an integral part of the analysis method.

Traces are frequently only documented from the requirements specification to development artifacts, resulting in 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 [78, 200]. If assumptions made during the creation of the RS are not (or no longer) documented, this knowledge may leave the project with the responsible analyst. Furthermore, pre-RS traceability supports change management [99]. If at a later stage in the project lifecycle, requirements are subject to change, the original rationale for the changed requirements is well documented through the traces. Stakeholders impacted by the change can also be identified, as well as possible alternatives that might have been previously dismissed.

Pre-RS traceability is therefore highly desirable, but the standard solution employed today – keeping separate documentation that must be continually updated – is burdensome and error-prone.

With an integrated method of ensuring pre-RS 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.

Within the scope of this thesis, we have developed a method for domain modeling, called QDAcity-RE [107].

The QDAcity-RE method was inspired by QDA methods that have been a mainstay of social science research for more than 50 years [205]. They are also becoming increasingly common in other fields of research, such as information systems research [126]. In this thesis, we describe the QDAcity-RE method and evaluate it using four exploratory studies. Through these studies, we provide an evaluation of its utility and the qualities of its output. We further
validated parts of the method using a controlled experiment.

In the QDAcity-RE method, requirements engineers sample stakeholders, interview them, correlate other materials, and perform QDA of the materials to derive a so-called *code system*. The code system is then extended to derive the relevant requirements engineering results. Traces from elements of the resulting artifacts to the source material are facilitated through the code system. These traces are therefore automatically generated through the analysis method, making a documentation after the fact unnecessary.

Keeping separate documentation of traces consistent with changes in all artifacts is a common problem. We expect, that embedding the process of trace generation within the analysis process solves a major problem of pre-RS traceability. Thus, a major hurdle for acceptance of such documentation is lowered.

This thesis focuses specifically on the derivation of conceptual domain models using the QDAcity-RE method. Peripherally we also examined the code system as a unified model that connects multiple artifacts of the RE process, such as behavioral models, domain specific languages (DSL), and requirements specifications written in natural language (NL). Here, the code system can facilitate inter-model consistency.

We developed our initial idea through four exploratory studies in four different domains. With each application, we refined our method.

The exploratory studies were all performed with traditional computer aided qualitative data analysis software (CAQDAS). Additional information required for documentation of traces and application of a meta model was documented either in the code memos or in supplemental spreadsheets. This manual documentation allowed us to learn valuable insights into the application of the method, however, it made the application impractically laborious and comparisons to other tool-supported methods would be of little value. We, therefore, developed our own cloud-based software for QDA, called QDAcity¹. QDAcity supports the creation of a UML class diagram based on a code system. The UML diagram presents a

¹www.qdacity.com
special view on a subset of the code system. Changes to either the code system itself or the UML model representation are modifications on the same data model, ensuring consistency between the two views.

Throughout the rest of this thesis, QDAcity-RE always refers to our method, and QDAcity always refers to the tool we developed to support the method.

The four exploratory projects also guided the development of the method. 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, in particular adding traces from a natural language RS to individual stakeholder statements. However, while these studies are good indicators of whether to pursue further study of these methods, their context is significantly different from each other and specific to the given application, making comparisons difficult. Using multiple methods in the same case is often not possible due to resource constraints.

With full tool support for the method and after several iterations of revising the method, we therefore conducted a controlled experiment to validate our hypothesis on improvements we expect from following the analysis phase of our method.

In the experiment we evaluate our method, used in the QDAcity group, against a different method [50, 51] for domain modeling used by the control group. The method employed by the control group is based on strategic reading and 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 [50]. Since both methods were new to all participants, we could also evaluate the learning curve and the ease of use for untrained analysts.

Our experiment showed that while the control group spent significantly more time analyzing the data, with the difference increasing over time, the QDAcity group scored significantly higher in the F-Measure when compared to an expert solution.

The contributions of this thesis are threefold:

1. A novel method for domain modeling, which improves over the state of the art by
exhibiting the following qualities:

(a) Inherent pre-requirements-specification (pre-RS) traceability between domain model elements and stakeholder statements.

(b) Efficient identification of inconsistencies in stakeholder statements.

(c) A high degree of completeness of the resulting domain model.

2. An evaluation of this method using four exploratory studies.

3. A validation of the analysis phase of the method in a controlled experiment with student participants.

1.1 Motivation and Original Idea

While our method is described in detail in chapter 4, the description of our exploratory studies in chapter 3 describes our process of arriving at this method. To better understand and motivate the exploratory studies, and this thesis as a whole, we highlight the main concepts of our method in the following subsection 1.1.1 and outline the expected benefits of applying these concepts to conceptual domain modeling in section 1.1.2.

1.1.1 Main Concepts

Iterative Process

A cornerstone of the analysis process is its iterative nature. It allows for early and frequent corrections. Each iteration consists of the sampling of stakeholders, the gathering of data, and the analysis process.

Data gathering should ideally be driven by theoretical sampling. *Theoretical sampling* requires the analyst after each iteration to determine gaps within the knowledge or contradictions, and then sample stakeholders that may help to fill these gaps specifically.
The iterative process concludes when saturation is reached. Saturation is a measure of how stable the analysis results are. It is reached when further data gathering does not lead to significant novel insight. Within QDAcity this measure is user-defined based on the number of added codes, deleted codes, changed code definitions, etc., over time. For example, saturation may be defined as no new codes emerging and less than five percent of definitions having changed in the last three iterations.

Coding

The data analysis consists of three distinct successive steps called open coding, axial coding, and selective coding.

Open Coding. In the open coding phase of the analysis, new concepts emerge from the data in the form of new codes. These codes are linked to the relevant parts of the data through a coding. Open coding leads to a list of unstructured codes. It is common that in the beginning this list is long, and may contain candidates that are later discarded.

Axial Coding. During axial coding, relationships between the codes are defined. This leads to a hierarchical structuring of codes into categories. Besides the hierarchy, other types of relationships may also be documented with each code. Further, during axial coding, the code system should be classified according to any underlying model, such as the coding paradigm, or the code system language (CSL).

Selective Coding. During the last of the three coding steps, all concepts need to be related to the single core category, which best addresses the purpose of the analysis. Everything that does not have a strong enough relationship to the core category should be discarded. This helps to focus on what is relevant.

Recommended Practices

The following practices support the analysis, reduce bias, and help ensure a valid and repeatable result.
Triangulation. Triangulation is a practice that requires the analysis results to be corroborated through different perspectives of multiple analysts, multiple types of data and different methods of data gathering and analysis.

Intercoder Agreement. One way of triangulating the results is to have multiple coders code the same data. The similarity of the two results is measured as intercoder agreement. A high score of intercoder agreement is evidence of low bias of the analyst.

Attention to Negative Cases. During the analysis, special attention should be given to identifying evidence that contradicts the current state of the analysis. These arising contradictions should then feed into theoretical sampling.

Member Checking. The results of the analysis should be reflected to the stakeholders. It is beneficial to not only perform member checks as an evaluation criterion at the end but also with preliminary results, in order to allow for necessary corrections to truthfully represent the expert’s perspective in the analysis result.

Peer Debriefing. During the analysis, the results as well as the processes leading to the results should be scrutinized and questioned by peers. In a peer review session, the analyst explains the process documentation and answers any questions their peers may pose to the process.

1.1.2 Expected Benefits

When applied to the creation of domain models during requirements engineering, the expected benefits of our approach are the following:

1. It closes the gap between the informal stakeholder material and formal domain models by adding pre-Requirements-Specification (pre-RS) traceability.

   This traceability is embedded in the RE process and documented in a new unified model, the code system. The inherent traceability eliminates the need to create and maintain traces after the fact.
2. It improves the process for deriving domain models from stakeholder materials by
   (a) providing a defined process, where previously analysts mostly had to rely on intuition and experience.
   (b) allowing the definition of a measurable stopping criterion to determine when the requirements elicitation process exhausted the relevant cases.

3. It improves domain model quality by
   (a) ensuring completeness of domain models, where previously key input might have been missed.
   (b) ensuring consistency by following principles of the constant comparison method.

Current tools support the documentation of traceability manually, linking requirements back to specific artifacts. However, creating and maintaining these matrices is a laborious documentation task that does not provide additional benefits for the actual analysis of the source documents [90, 17, 35]. If the documentation, however, is created as part of the text analysis, it can serve both purposes: to better understand the target domain and to create better documentation of the analysis process. The improved documentation makes each element traceable.

We use qualitative research methods to solve part of the “grand challenge of traceability” [77], ubiquitous traceability, in a pre-RS context.

The fine grain traceability provided by using QDA methods further improves the ability to perform change impact analysis. If any passage within the source documents changes, the corresponding parts of the model can be identified and adapted.

Through the traces, decisions made during the analysis process become explicit. For instance, when resolving conflicting descriptions, alternative interpretations or conflicting viewpoints are documented beyond what is visible in the final model.
Our approach also empowers less experienced analysts by offering a codified method, that the analyst may follow to achieve higher quality models of the domain.

Further, the in-depth analysis through coding, especially when a high degree of in-vivo codes (code using exact terminology from the data) are used, contributes to a better understanding and definition of terminology that is close to the language use of the stakeholders.

These benefits can, however, not be achieved at zero cost. Our experience, which is in line with previous related studies, is that using QDA methods for domain analysis increases the effort required for the analysis significantly. The required effort is directly dependent on the tool support.

1.2 Structure of the Thesis

The remainder of this thesis is structured as follows: First, we relate our work to the existing body of research in chapter 2.

Four exploratory studies that were used for the iterative development and evaluation of our method in four different domains are presented in chapter 3. The result of these four iterations of refinement is the method description presented in chapter 4.

The engineering work necessary for tool support for the QDAcity-RE method is presented in chapter 5.

The controlled experiment we conducted is then presented in chapter 6, before closing with a conclusion in chapter 7.
Related Work

In this chapter, we first lay out related work on approaches for utilizing QDA and techniques of qualitative research within the context of requirements engineering, and contrast these with our own research in section 2.1. We then present an overview of research on the quality aspects of domain models, in terms of which we evaluate our own method in section 2.2. Subsequently, we summarize research on traceability, and in particular pre-RS traceability, which we consider a key benefit of our method in section 2.3.

2.1 QDA in Requirements Engineering

We are not the first to use QDA in requirements engineering. Related fields like knowledge engineering and process modeling have also employed QDA [27, 28, 214, 151, 187]. Most of this work does not just use QDA, rather most related work utilizes QDA in the larger framework of Grounded Theory (GT) [75, 74, 73, 30]. GT is an approach to theory building using qualitative data that provides a methodological framework in addition to the supporting anal-
ysis practices. Depending on the epistemological stance, different variations exist, whether it is positivistic or constructivistic.

QDA, and in particular GT, have been used to develop models \cite{27, 23, 214, 28, 82, 83}, requirements \cite{29, 198, 33, 157}, personas \cite{60, 61, 133}, and architectures \cite{15, 72}. Our approach differs from the mentioned work in that we define and evaluate our own method, QDAcity-RE. QDAcity-RE was inspired by GT but drops many of its epistemological assumptions and preconditions. We do not believe that GT itself can be applied to RE, but rather that we need to define and validate our method independently of qualitative research. An important key tenet of GT is the requirement of a purely inductive approach to knowledge generation. By developing an independent method, we take a more pragmatic stance on epistemology and the requirement to discard any preconceived notions about the domain. This sentiment is also apparent in much of the published research utilizing the terminology of GT in RE.

Carvalho et al. tested the application of GT to descriptive process modeling by producing two process models: one by an experienced software engineer and one by an experienced qualitative data analyst (researcher) without software engineering experience \cite{27}. They found that using the GT method cannot compensate for the software engineer’s expertise and experience. However, its application can improve the modeling process, because it forces the analyst to explore the complexity of the data and to systematically abstract from it. Similar findings are described by Pidgeon et al., who applied GT to knowledge elicitation \cite{151}.

Pidgeon et al. add, that GT secures the traceability of a derived model back to original data sources through the documentation of the analysis process in codes and memos. However, they point out that the produced model is still an interpretation that needs to be validated.

Both authors 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 \cite{23}. Our studies concur with \cite{27, 151} concerning the effort required for using methods like GT with the
purpose of domain modeling. Many of the problems we encountered in this regard, however, can be solved through better tool support tailored toward domain modeling.

Würfel et al. propose a holistic approach for data elicitation, data analysis and the determination of use cases, similar to the approach that is proposed by us [214]. They define two process phases: In the first phase, GT practices are employed for data elicitation and analysis. In the second phase, domain descriptions are turned 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 [95]. They adapt GT by using predefined 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. These studies do, however, neither show how to extract specific requirements on a lower abstraction level, nor how to extract structural elements for a conceptual domain model. In addition, they do not describe the data analysis process of their case studies in any 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 [28]. They demonstrate their approach by deriving a UML class diagram from a textual high-level description of a university support system. However, the diagram they developed 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 is the addition of conjectural categories to their model, which were not derived from the data, but were based on the experience of the analysts. They discovered that, apart from the advantage of traceability, the iterative process of GT allows the analyst to discover and close information gaps earlier in the process.

Chakraborty, Rosenkranz, and Dehlinger propose a procedure called Grounded and Linguistic-
Based Requirements Analysis for eliciting non-functional requirements (NFR) [29]. 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 of GT necessary. Also, since 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 providing them guidance about the theoretical principles to apply.

Chakraborty et al. used predefined categories of NFR [29]. These categories were related using Mylopoulos, Chung, and Nixon’s NFR framework [138]. Thomas et al. also use an analytical framework, including predefined thematic codes and extraction rules, to use QDA for the determination of privacy requirements for mobile applications [198]. Thomas et al. 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 [181]. They used a GT-based approach to develop a classification system based on the coding of RS documents.

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 was desired, pre-defined categories proved immensely helpful. An example of such categories were the dimensions
for software quality according to ISO/IEC 25010. However, for the conceptual model, we found it more helpful to start without preconceptions. All of this, however, is highly dependent on the domain.

The use of GT to model requirements is also investigated in Halaweh’s studies [82, 83]. Halaweh states that categories and their relationships derived from Corbin and Strauss’ coding paradigm [39] 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 to identify users to interview and theoretical saturation can be used as an indicator to stop requirements elicitation. Halaweh argues that by applying GT and thereby letting requirements emerge from the data, requirements become 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 implement pro-active measures and improve training measures to overcome organizational problems, such as resistance to change. In a case study Hallaweh conducted and analyzed interviews from which he then retrieved a class diagram [83]. However, although he states equivalence of GT and object oriented analysis and design (OOAD) elements, he does not explain why and how these can be transferred, and does not present guidelines for coding and transferring the informal model to the class diagram. In contrast, we developed a concrete mapping from a code system which is classified with our code system language (CSL) to UML class diagrams.

Using the CSL, we use our method to produce requirements artifacts directly through QDA, as others have also proposed using GT. Chitchyan et al., by contrast, employed a strategy for requirements elicitation that uses GT as a precursor to the actual definition of requirements [34]. In this process, GT would serve to define “contexts and preferences which serves as the foundation for the theorized requirements” [34]. From this understanding of
the domain, candidates for requirements would then arise. We agree that an 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. That is why in our method a conceptual model is explicated as a specific view on the code system visualized through a UML class diagram with an explicit mapping between this representation and the code system.

The visualization of the code system in this specific view is only possible since we underpin the code system with the CSL. The need for more precise expression of the information encoded within a code system has also been discussed in qualitative methods research. The introduction of the coding paradigm is one of the main divergences between the variants of GT proposed by the two founders of the method, Glaser and Strauss. The coding paradigm serves to classify codes belonging to distinct dimensions of the investigation and helps the researcher as guidance on how to further develop the code system.

Glaser proposes a more flexible method for relating different concepts of a theory by using theoretical codes, which he divides into coding families [73]. Charmaz criticizes that Glaser does not provide a comprehensive model and that some of the theoretical codes overlap and seem random [30]. Their use is therefore difficult for a requirements engineer who is a novice at GT [28, 30].

Knowledge relevant for the domain analysis is typically in part implicit. Stakeholders rely on a common understanding of things that are not necessarily explicated in any way. The knowledge might be perceived as obvious or trivial by experts, but not clear to the analyst or developers. Uncovering such tacit knowledge is a widely acknowledged problem in requirements elicitation [69]. Similar to the idea put forth by Hallaweh [82, 83], the use of GT to elicit social aspects and tacit knowledge has also been suggested by Chitchyan and Bird [33], 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 [157]. 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 analysis [157]. 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. The visualization of such a notation allows for a much more efficient and precise overview compared to an analysis of written memos. A model of security requirements with a focus on the socio-technical aspects of a system was also developed by Fléchais using GT [63].

The social and political factors, that may influence stakeholder’s requirements was also addressed by Thew and Sutcliffe with their own method [196]. 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 stakeholders’ values, motivations and emotions [196].

The only other existing approach bridging the gap between reading and conceptualizing that is currently supported through tools is based on strategic reading and root cause analysis (RCA). The method and tool support was proposed by Diaz [50]. One primary application of the method was illustrated in the research context, with the purpose of analyzing research papers on related work with a clear research question in mind. The method further claims to facilitate conceptual modeling. The method is specifically geared towards design science research, which has the purpose of creating a new, innovative, and purposeful artifact [203] which “extends the boundaries of human problem solving and organizational capabilities by providing intellectual as well as computational tools” [88].
Diaz argues that bridging the gap would be achieved more effectively not by creating entirely new tools, but rather through connecting existing tools. By contrast, our approach embraces an integrated tool for reading, annotating, and modeling. Diaz defined two types of tools needed for conceptual modeling in design science research, which are reading tools such as Mendeley\(^1\) and Hypothes.is\(^2\) and RCA tools such as Mindmeister\(^3\). As an artifact of his research, Diaz designed an extension for the chrome browser, called D\_Scaffolding\(^4\), which facilitates the connection between the reading tools and the RCA platform. Between the two types of tools, Diaz describes two types of pipes: the “annotation pipe”, and the “purpose pipe”. The annotation pipe allows for evidence uncovered from the reading tool, which supports or refutes ideas, to be considered during the RCA. The purpose pipe enables guidance of activities in the reading tool through reading purposes set within the RCA tool. In the concrete case of Mendeley and Mindmeister, the chrome plugin then “pipes” the information between the two tools.

Because of the comparable degree of traceability from reading material to the created conceptual model we used this method as a control to compare it with the QDAcity-RE method within the validation of this thesis. We present more details about this method within the description of our experiment design in section 6.1.3.

2.2 Quality Criteria of Domain Models

2.2.1 Consistency

Consistency is a crucial quality criterion for evaluating the semantic quality of a conceptual model. Consistency can be distinguished as either intra-model consistency which deals with semantic inconsistencies within one model, or inter-model consistency which describes the level of conformance of two separate models towards one another.

\(^1\)https://www.mendeley.com
\(^2\)https://hypothes.is
\(^3\)https://www.mindmeister.com
Our proposed method addresses both. The analysis process ensures an internally consistent model in the form of a code system. From the code system, multiple types of domain models can be derived. The derived models feature inter-model consistency between each other because they were derived from the same model. Inter-model consistency was also used to evaluate experimentally created models against a model solution. Whenever the code system changes, all derived models automatically evolve reflecting those changes.

Artale, Calvanese & Ibanez-Garcia propose to measure intra-model consistency in terms of full satisfiability, meaning that for a model to be consistent, there has to be at least one possible instantiation of it [7]. Blauboer, Sikkel & Aydin proposed an algorithm for determining satisfiability [17].

Other methods of consistency checking propose the definition of a set of rules defined in Object Constraint Language (OCL) which restricts the expressiveness of UML to a subset that ensures consistency [76].

For inter-model consistency Sabatzadeh et al. have proposed a method based on merging different models into one and then checking the resulting model for well-formedness regarding UML and any predefined constraints [171]. The merged model may for instance have cyclic inheritance relationships resulting from the inconsistent placement of classes in hierarchies in separate models. Such cycles can then be identified. Unifying multiple views into a single model to check consistency is a common approach, however, the inconsistencies discovered will be in terms of the semantic framework of the unified model. The resulting model often can only partially be translated back to the languages of the original models [49].

In our approach, the use of QDA helps the business analyst create one central model, the code system. The code system is a unifying model from which all other specific RE artifacts can be derived. We, therefore, do not need to merge other models to derive a central unified model for checking consistency, because it already is the foremost output of our method.

Our approach establishes the code system as a unified model from which we can create different individual models, so the merging step can be eliminated by checking the consistency
of the code system instead. To this end, we defined the CSL to precisely describe the code system and enable the creation of conceptual models based on it.

2.2.2 Completeness

Completeness means that the model contains all the statements about the domain that are correct and relevant. Although this is a desirable attribute, in practice full completeness can usually not be reached unless the domain is very small. Therefore, the analysis process has to stop at a point where further modeling is less beneficial than applying the model in its current state. Lindland, Sindre & Solvberg call this state feasible completeness [122].

For evaluating the completeness of Entity Relationship models, Moody proposes to count the instances where no traces can be created from model to user requirements and vice versa [134]. This, however, implies a perfect specification of user requirements. As an additional metric, Moody proposes to evaluate inconsistencies between process models and the data model as a means for determining the completeness of the data model.

The metrics for measuring the completeness of conceptual models are usually impractical in a real project. Completeness of a domain model is defined as the proportion of concepts that are known versus concepts that are in the model [114]. Other definitions of completeness rely on comparing functional dependencies between the concepts rather than the concepts themselves [96], but the fundamental problem with these metrics remains. In most projects, an actual measurement of completeness of the model is intractable due to the complexity of the domain and the diffuseness of the knowledge representation of the known concepts. For the natural language specification, the known concepts that would constitute completeness can be concepts in an ontology or a conceptual model [103]. We, however, want to gauge the completeness of the conceptual model by measuring saturation.
2.3 Traceability

Requirements traceability (RT), done right, is an important property of RE artifacts. It helps to ensure quality aspects such as adequacy and transparency, and facilitates better maintainability due to more consistent and complete requirements [213, 188].

Traceability is a powerful tool to foster consistency between different artifacts. Shukla et al. describe it as particularly helpful for decision making in multi-view modeling [182].

However, Winkler and Pilgrim state on the limitations of RT that “traces cannot be identified completely using automatic processing without human intervention”[213]. This exemplifies the widespread notion, even among researchers, that traceability is something that is produced on top of the normal process, and the intervention required for its creation is regarded as a burden.

In contrast, we suggest that traceability can be produced naturally, as a side product of the analysis method. In this thesis, we apply and validate qualitative research methods as a solution for solving part of the “the grand challenge of traceability”, as proclaimed by Gotel et al. [77]. This grand challenge is traceability, that “is always there, without ever having to think about getting it there, as it is built into the engineering process; traceability has effectively disappeared without a trace”[77]. We focus on the pre-requirements specification (pre-RS) traceability, which links requirements with their source. Related work on pre-RS traceability is laid out in more detail in the subsection 2.3.1

As Hofmann and Lehner [90] point out in a survey of 15 software projects within the telecommunications and banking industries, stakeholders identify the lack of traceability as hurtful to their projects. The more successful projects in this study incorporated traceability matrices, yet only five out of 15 projects actively tracked their requirements throughout the development life-cycle. Similar findings were presented by Blaauboer, Sikkel, and Aydin [17], who interviewed eight project managers of IT projects in the banking industry of which only three had a clear concept of what traceability is and how to implement it. In
some cases, traceability techniques were used, but the project manager was unfamiliar with the term “traceability”. A survey by Huang et al. [35] suggests that practitioners in safety critical domains have a comprehensive understanding of how traceability should be established, however, they do not implement traceability more willingly or effectively.

The current disconnect between traceability research and its practice is significant. We first demonstrated the feasibility of our method through the exploratory application of our method in multiple industry projects and we evaluated the acceptability for practitioners.

Within an ongoing exchange with our industry partners, we have observed that the potential benefits of high quality traceability are being understood by management. Although the impact is described as hard to measure, it is considered to be tangible and substantial. Developers tasked with the documentation of such traceability, however, frequently have a much more aloof perspective on the effectiveness of such methods. This indicates that there is a benefit to be realized, but current implementations leave much room for improvement.

One reason of poor adoption of traceability practices in industry lies in the often times disparate groups of stakeholders being burdened by the creation of trace links (the providers [78, 87]) and stakeholders making use of them (the end-users [78, 149]).

Besides for the creation of traces, significant effort must be expended for their maintenance. The ephemerality of traces, or traceability decay is considered a significant problem [160, 159, 201].

To address concerns about the effort and cost required for implementing traceability practices, Espinoza et al. recommend, that within a project there should exist “guidelines to maintain the number of artifacts to trace as low as possible” [58]. To ease the burden of manual trace generation for the analyst, natural language processing (NLP) can be employed which can recommend candidates for new traces [86].

Inadequate tool support is also a recognized problem, contributing to poor adoption of traceability practices [190, 213, 139].

A strong driver for adopting strategies for traceability is regulatory compliance with norms
and certifications \[160, 139, 52].

2.3.1 Pre-RS Traceability

The method described in this thesis has a clear focus on impacting pre-RS traceability. Pre-RS traceability means establishing and maintaining traces between requirements and their origins. The origin of a requirement can be documentation from interviews with stakeholders, workshop notes, requirements of legacy systems or systems archaeology, etc.

Traceability from the RS to artifacts which are created later in the development life-cycle, such as design models, and source code, is called post-RS traceability.

Pre-RS traceability is less prominently researched than post-RS traceability \[140\]. Krause et al. performed a systematic literature review on pre-RS traceability, finding only 67 articles relevant to the topic specifically within the years 1992 and 2020 \[111\].

Pre-RS traceability allows for the identification of the rationale behind a requirement, and the decision making that led to the requirement becomes transparent \[4, 19\].

Reasons for poor adoption of pre-RS traceability are similar to those for RT in general. The trade-off between effort and benefits is often considered unfavorably \[78, 210, 200\]. In some cases, this assessment is rooted in the fact that the costs and the benefits of implementing pre-RS traceability are not realized by the same person.

Issues regarding tool support for full RT are especially pronounced for pre-RS traceability. Often, only parts of the development cycle are being supported by tools, or they lack certain capabilities for pre-RS traceability. Our work, by contrast, does provide tool support for pre-RS traces. However, our solution is not a general tool for trace management, hence misses ways of handling the later stages in the life cycle of a specification.

Established pre-RS traceability practices typically exist due to regulatory requirements on the traceability, and extra funding needs to be allocated towards managing the traces, independent of where potential cost savings might be the benefit later in the development cycle \[156\].
Liang et al. suggest the need for pre-RS traceability especially for larger, more complex projects \cite{119}. They propose to use a “collaborative tagging method to tag the requirement statements by stakeholders themselves, and then these collaborative tags (collective knowledge) is transformed into requirement ontologies with formal semantics and richer expressivity”\cite{119}. Liang et al. suggest to apply this method during an early step of the requirements elicitation, which they coined the pre-requirements analysis. The idea is similar to what we are proposing. However, by contrast, we do not use ontologies, rather the equivalent artifact with formal semantics and richer expressivity would be the code system which is the foundation for deriving various other models and artifacts. Further, our method is underpinned by practices adopted from qualitative research where procedures for such tagging have been established that ensure rigor in the analysis.

In contrast to Liang et al. \cite{119}, Ahmad and Ghazali concluded that pre-RS traceability is the first type of traceability that yields benefits to the project in the context of their multiple-case case study, specifically for smaller projects \cite{3}. Ahmad and Ghazali stated four reasons why pre-RS traceability was regarded as more favorably compared to post-RS traceability in their case companies. The reasons were listed as follows \cite{3}:

1. There were no prerequisites to documenting traces.
2. A large group of stakeholders profited.
3. Pre-RS traceability was easier to gather and less expensive to document.
4. Pre-RS traces were the first and most frequently documented traces.

Pre-RS traceability also helps with evolving requirements and change management \cite{158, 4, 3, 213}. Traceability in general is recognized as helpful in particular for change impact analysis \cite{99, 11, 182}. This is most prevalent within complex systems. However, a case study by Panis suggests this effect to be a less significant reason for documentation of pre-RS traceability \cite{149}. 

26
When pre-RS traceability is consistently documented, not just the traces themselves but also any potential lack of traces for certain requirements, yields valuable insight. Either the requirements where traces are missing are obsolete [210], or they are evidence of undocumented tacit knowledge either from the analyst or the domain expert [69, 192, 193].
We developed our method iteratively while applying it in different contexts. We conducted four exploratory studies, in each testing our working model of the process, documenting strengths and weaknesses, and subsequently further refining the method. The four studies are detailed in their respective chapters 3.1, 3.2, 3.3 and 3.4.

The first two projects in which we developed our initial version of the method were performed in the domains of medical imaging diagnostics and railway systems. Both required the creation of a conceptual domain model which we developed using QDA techniques. Following these two projects, we applied our learnings in two additional studies within the domains of human resource (HR) development and qualitative research methods.

All of these projects were interwoven with the creation of the method and were used to explore how the transfer from research method to domain analysis can be implemented.

Table 3.1 provides an overview of the qualitative data coded in the scope of the four exploratory studies.
Table 3.2: Overview of Exploratory Studies.

<table>
<thead>
<tr>
<th></th>
<th>Medical Image Diagnostics</th>
<th>Railway Systems</th>
<th>Human Resource Development</th>
<th>Qualitative Research</th>
</tr>
</thead>
<tbody>
<tr>
<td>Member Checks</td>
<td>Informal</td>
<td>Follow-up</td>
<td>Follow-up</td>
<td>Informal</td>
</tr>
<tr>
<td>Data Triangulation</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Outline Conversation</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Fully Traceable</td>
<td>-</td>
<td>-</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Table 3.1: Coded Data.

<table>
<thead>
<tr>
<th>Study</th>
<th>Data</th>
<th>Coded Segments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medical Image Diagnostics</td>
<td>8 in-depth interviews</td>
<td>1563</td>
</tr>
<tr>
<td>Railway Systems</td>
<td>4 in-depth interviews</td>
<td>754</td>
</tr>
<tr>
<td></td>
<td>project documentation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>norms and standards</td>
<td></td>
</tr>
<tr>
<td>Human Resource Development</td>
<td>6 in-depth interviews</td>
<td>1237</td>
</tr>
<tr>
<td></td>
<td>6 workshop transcripts</td>
<td></td>
</tr>
<tr>
<td>Qualitative Research</td>
<td>6 in-depth interviews</td>
<td>778</td>
</tr>
</tbody>
</table>

Table 3.2 further details some of the key differences among the four cases.

With each of the four cases, which were executed in the order presented here, we aimed to refine our method and thus focused on a specific aspect. The focus of each study is as follows:
1. Medical Image Diagnostics
   - Establish feasibility for semi-formal modeling
   - Create a DSL using QDA

2. Railway Systems
   - Improve conceptual modeling using QDA

3. HR Development
   - Compare workshops and interviews as input
   - Document full traceability

4. Qualitative Research
   - Derive a conceptual model, a behavioral model and a natural language specification from the same code system

The first two studies were conducted in collaboration with an industry partner. The advantage of this was, that it was an excellent way to gauge if and how our approach makes sense in real-life scenarios. On the other hand, this was a trade-off between relevance and rigor, since we lacked full control over the process. This became apparent in our third study, which started out as an industry project, but we then ended up having no control over the sampling strategy. We, therefore, conducted this, and the final study without an industry partner, to more accurately test our method as we envisioned it.

The evaluation model we used to determine the success of the exploratory studies is presented in table 3.3. The criteria of this model are examined for each case in sub-chapter 3.6.
<table>
<thead>
<tr>
<th>Quality / Goal</th>
<th>Evaluation Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Leads to better documentation</td>
<td>ー Model elements can be traced to stakeholder statements</td>
</tr>
<tr>
<td></td>
<td>ー All model elements are grounded in stakeholder statements</td>
</tr>
<tr>
<td>2. Improves completeness</td>
<td>ー Domain expert does not consider important elements missing</td>
</tr>
<tr>
<td></td>
<td>ー Model covers all aspects in existing knowledge representations</td>
</tr>
<tr>
<td></td>
<td>ー The analysis process reached saturation</td>
</tr>
<tr>
<td>3. Improves consistency</td>
<td>ー Domain expert can not identify any inconsistencies</td>
</tr>
<tr>
<td></td>
<td>ー Inconsistencies were uncovered during coding</td>
</tr>
<tr>
<td>4. Can handle different types of input materials</td>
<td>ー Different input materials used (Interviews, Workshops, etc.)</td>
</tr>
<tr>
<td>5. Supports the creation of different target artifacts</td>
<td>ー Different output artifacts created with inter-model traceability</td>
</tr>
</tbody>
</table>
We used MaxQDA\(^1\) as the analysis tool for our exploratory studies. Additional metadata and documentation that our method produced was either documented in memos within MaxQDA, or in separate spreadsheets.

3.1 Domain Modeling for Medical Imaging Diagnostics

We first applied our method to the design of a domain specific language (DSL) for medical imaging diagnostics. This study was conducted in collaboration with Siemens Healthcare and Mempel [131].

To create a DSL, the analyst not only requires technical know how but needs to have a deep understanding of the domain. A deep understanding is required so the DSL will be accessible intuitively for the domain experts, who represent the target audience of most DSLs. By grounding all concepts in the original data provided by the domain experts, especially when using mostly in-vivo codes, it is ensured, that the language used in the DSL stays close to the terminology used by the experts.

We, therefore, conducted a domain analysis using our QDA based method. As part of this domain analysis, a feature model and a conceptual domain model were created.

Through our analysis, we identified typical workflows for medical imaging diagnostics using computer tomography (CT) and magnetic resonance imaging (MRI). These workflows guide the diagnostician through the diagnosis processes. The goal of defining these workflows using a DSL was to achieve standardized medical findings.

To elicit the processes in which users typically engage to diagnose an image, eight interviews were conducted. These were subsequently transcribed and analyzed using an early outline of our qualitative analysis method. The interviewed personnel included the two project managers, two test engineers, a software architect, and three product coaches who were working with all stakeholders to define the product requirements.

In addition to these interviews, we utilized background material on typical oncological

\(^1\)http://www.maxqda.com
diagnostics provided by the Siemens AG. We triangulated the results using multiple data sources. This increases trustworthiness [81].

After the qualitative analysis concluded, and a stable code system supported by over 1500 coding instances within the texts had emerged from the data the code system was then transformed into a formal feature model.

Even without tailoring the coding process toward feature models around 80% of the codes that emerged from the data could directly be translated into a node of a feature model.

While the translation into a feature model could be performed with only minor modifications, the modeling of the domain’s structure through a UML class diagram required more implicit knowledge of the domain. This knowledge was not explicitly represented in the code system. We attribute this to the fact that within this study the coding was performed very similarly to how it is performed in theory building research. This meant a high degree of freedom regarding the structure and semantic of the code system. Also, methods like GT which we based our approach on tend to be process-centric by nature.

A classification of codes as classes, attributes, or relationships had to be added ad-hoc during the manual transformation process. Some of this information could be extracted from code memos, but most of it was created through an additional interpretative step.

In a final step, the feature model was then translated into a DSL, ensuring that the terminology used in the DSL is the same that was used by the interviewed participants of the study.

From this first exploration of our idea we concluded, that a partial automation of transforming a code system into a domain model warrants further investigation. We also concluded, that such a transformation would have to be supported by a more formal code definition which is supported by a meta model.

A more structured approach would help this transformation. However, if the coding becomes more restrictive it also limits the possibility of unexpected results emerging from the analysis. Further research into finding the appropriate balance in this respect was needed.
3.2 Domain Modeling for the openETCS Toolchain

The second exploratory study was performed in the domain of railway systems in collaboration with members of the openETCS project. Our partner for this second study was Deutsche Bahn AG. The purpose of this project was to understand the needs within the openETCS project toward their tool chain. In essence, we performed requirements elicitation and analysis for the tool chain required for the software development process within the openETCS project.

3.2.1 Data Sources

Two major data sources were used for this project: We conducted four interviews with three openETCS stakeholders and we processed the relevant official documents from the openETCS repository on GitHub and involved them in our coding process. By drawing on the vast and diverse existing documentation as well as our own interviews, we achieved data triangulation [81].

The individuals we interviewed were directly involved in the openETCS project as team members of the development team. All interviews were executed in a semi-structured way.

The initial interview took place in September 2013. Our interview partner was the project leader of the openETCS initiative. At this point, we had received an informal introduction to the project through a telephone conference with two of the project leaders. We had also received access to the documents repository.

In preparation for the first interview, we analyzed the currently existing requirements document and carved out inconsistencies, imprecise wordings, and potential mistakes. The discussion of these aspects served as an introduction to the different topics, but the interview became very open and often one aspect brought up the next one. This resulted in a long and detailed conversation that covered the whole project.
3.2.2 Data Analysis

The interview guideline was incrementally adapted after each round of interviews to address issues that were revealed through the constant comparison method. We focused on the following items. This study was conducted in collaboration with Schmitt [176].

- Gaps within the current code system, where a lack of deeper information was evident.
- Discrepancies within the current code system which originated in contradictions in the statements of different interviewees.
- Aspects of the target domain that appeared during the analysis, but were not evident until the most recent iteration.

We used open questions to create a relaxed atmosphere and encouraged the interviewees to talk about what came to their minds. They were also free to change the topic if they wanted to. The prepared questions served only as an outline of the conversation. The intention was to let the interviewee speak freely, which is also transferred from the traditional GT techniques. This shall ease the discovery of topics that the analyst might not yet be aware of.

However, when statements came up that we did not understand or included unclear details, or when the current interviewee contradicted statements from earlier interviews, we specifically deepened the discussion on these issues and asked for more details.

After running through the coding process, we then revised and checked the new version of our code system for quality. Hence, the code system is smoothed and corrected after each iteration. In addition, the memos containing the code definitions were updated after each iteration. Inconsistencies were documented, as were poorly understood concepts, which served as the basis for discussion within the next interview.
3.2.3 Evaluation of openETCS Study

Within the openETCS study, we investigated the abstraction levels which naturally occur in the code system through the coding process with regard to a mapping of codes to domain model elements.

We found that our method is capable of processing pieces of information on all levels of abstraction since it facilitates their hierarchical and logical ordering. An abstract concept will be found on a high level within the code system and its details will be subsumed in the subordinate levels.

In general, the codes that were mapped into concepts of the domain model could typically be found on the middle levels. The highest levels within the code system provided an abstract perspective split and therefore a structural order, i.e. “tools” vs. “artifacts”. The low-level codes on the other hand mostly represented details of a concept such as the concept’s behavior or particular attributes.

With this being the first within our four studies that documented full traceability between the domain model and original stakeholder material by means of the code system, we were frequently challenged by lacking tool support for the coding process. We chose to document the additional meta information necessary for creating a conceptual model from the data within the code memo using an XML structure and maintaining these manually. On the side of the domain model, the links were represented by the code ID, which was also maintained manually. Changes made in one of the artifacts had to be consistently copied into the other one. We later addressed this issue with our own tooling solution.

3.3 Domain Modeling for Human Resource Development

In our third study, we employed our method for domain modeling in the domain of human resource (HR) development. This study was conducted in collaboration with Kunz [116].

The main functions of HR development are (1) training and development, (2) organiza-
tion development, and (3) career development of employees within an organization [209]. The goal of this project was to evaluate the use of QDAcity-RE for the creation of a conceptual domain model, using in-depth interviews with experts working in the field of HR development as a data source.

3.3.1 Project Pilot: Industry Study

We initially started our project in the domain of HR development as an industry project, in collaboration with an industry partner that wanted to replace their existing tooling for HR development.

The goal of this project was the elicitation of requirements for the new tool. Before we joined the project, a system archaeology on the existing system had already been conducted, and attempts at formalizing the requirements had been undertaken which were not satisfactory. The so-called gap-lists of items that were considered missing in previous attempts to formalize the requirements were then used as a basis of discussion in a workshop setting to formalize the target domain.

Data Sources

The data used in this study was gathered from workshop transcripts. We were not allowed to create audio-recordings of the workshop for verbatim transcriptions but were allowed to transcribe the workshop while sitting in. In each workshop, there were always two of us present, and we transcribed each participant’s statements as exact as possible, including non-verbal communication such as laughter.

After the workshop, we then matched the two transcription versions to identify inconsistencies. While the transcriptions varied in their degree of detail in some instances, there were no instances where significantly different meaning could be attributed to any of the contributions to the workshop.

The workshop participants consisted of
• the product manager
• the technical lead for the project
• two developers
• two internal business analysts
• a consultant
• the moderator

Data Analysis

We analyzed the transcripts iteratively after each interview and applied constant comparison to match new evidence with previous findings.

Because the workshops were already highly structured, the coding of the data for the most part followed the structure of the workshop, which in turn followed the structure of the previously generated gap-lists.

Evaluation

Even though this study was not completed, we did learn some very useful insights into the application of our method.

Even though our method is independent of the type of data, and also works with workshop transcripts, it became apparent during this study, that for the type of top-down workshops done during this study, our method yielded little benefit in the form of unexpected insight. Still, the required effort for transcribing and coding of the data was significant.

We generalize from our experience that, the more structure already exists in the data, the less utility there is to our structured analysis approach.

We conclude, that QDAcity-RE is better suited for a bottom-up, or middle-out, approach than for a top-down analysis starting with significantly structured analysis artifacts. We have
used structured data as supplemental material successfully. Either for triangulation of a discovered truth or for specifically pre-structured parts of the code system.

A further problem we encountered, which contributed to the failure of this project, was the lack of support from employees within the cooperating organization. While we did have the full support of management, the participants in the workshop expressed concerns about the documentation of not just the results of the workshop, but the whole process, and the flow of conversation.

In particular, the documentation of tacit communication raised concerns, with regards to the fear that particular sensitive issues might be perceived as not taken seriously when laughter was a documented reaction to the topic being brought up.

Therefore, when implementing our method special attention has to be given to the governance of data privacy and security. Further study into the aspects of organizational change is needed.

3.3.2 Data Sources

All domain experts who participated in this study had high level management positions in HR and experience in HR development.

Their employing companies varied in size from a local company with 50 employees to an international corporation with over 100,000 employees worldwide and operated in the sectors IT and market research.

The first interview was guided by 12 open questions, which aimed at gaining an overview of the domain. For the following interviews, analysis results determined the interview questions according to the principle of theoretical sampling. We conducted semi-structured interviews, so the prepared questions were used as a guideline and we adjusted to participant’s answers [39, 137]. This was important because we wanted to capture the knowledge of the domain experts and not force preconceptions on the data [39, 143].

To clarify inconsistencies, close information gaps, and extract more detailed information,
we conducted follow-up interviews with two of the domain experts.

As a secondary data source used for data triangulation, literature on HR development [170, 1, 13, 197] was used to clarify the definitions of terms. Although, within a research context, literature research at the beginning of the research project is avoided in GT, Corbin and Strauss believe that literature may be used to support the analysis as soon as the main categories of the theory have emerged [71]. For the purpose of domain modeling in a requirements engineering context, we believe that using all possible data sources from the beginning can help mitigate the main drawback of our method, its high initial investment of resources, without significantly distorting the structural aspects of the domain the analyst is trying to capture through a model of the domain.

The interviews were audio recorded, anonymized, and transcribed. It is important to note, that the decision on what and how to transcribe may significantly affect the end result [130], so outsourcing to an external transcription service should only be considered after careful and explicit deliberation of what parts of speech to transcribe into text form. In this study, we decided to do the transcription manually ourselves.

Corbin and Strauss advise transcribing interviews fully at the beginning of the research project and in later stages only to transcribe those parts of an interview which are important for the theory [39].

To limit the risk of missing useful information, we transcribed the whole content, but left out introductory and closing conversations and defined a simplified transcription system [109]. The speech parts of interviewees were transcribed word for word, including laughter. However, we did not include details such as accentuation or the lengths of breaks, because they are not relevant for the purpose of our research [12]. For the speech parts of the interviewer, we left out parts that did not include any information such as expressions of comprehension, because this would interrupt the information given by interviewees unnecessarily.
3.3.3 Data Analysis

When we applied our QDA based method to our example, concepts emerged from the data during open coding. The coding process started after the first interview had been conducted and transcribed. In order to represent the domain terminology, primarily in–vivo codes were used [12, 117]. Units of coding varied in size from one phrase to a whole paragraph. Coding a whole paragraph was sometimes necessary to preserve information about the relationships between concepts. The units of coding belonging to one concept were compared to investigate their differences and similarities and to guide the questions for the following interviews.

Usually, actors are not coded explicitly in GT research projects, because they are intertwined with other concepts. For example, a study investigating how patients deal with pain includes concepts such as “experiencing pain” or “pain”, but no concept “patient” [30, 39]. However, actors, including external systems and organizational units, need to be represented in a conceptual domain model [117, 165, 208]. For the domain of HR development, for example, “employee” is a central concept. The same is the case for objects and places, which are normally not investigated explicitly during GT research. Therefore, actors, places, and objects, which include tangible and intangible objects and the concept type “idea” of GT, need to be coded as well.

Because conceptual domain models represent the entities of a domain, these are the phenomena we want to study and were therefore developed into categories. Concepts which seemed to belong to the same aspect were grouped into categories. For example “giving feedback”, “feedback survey”, “360-degree feedback” and “evaluating feedback” were grouped under “feedback”.

We also coded background information, such as the position of the interviewee in the organizational structure and the current systems in use, as well as information about the purpose of HR development. Although these codes should be clearly distinguished, such information should be captured and kept in mind during the analysis, as it might be the reason for differences between incidents and contain important information for later design decisions.
3.3.4 Evaluation of HR Study

The domain model created through our method was evaluated with regard to the following quality aspects proposed by Bolloju and Leung [18]:

- Syntactic quality: The domain model adheres to the modeling language.
- Semantic quality: The domain model represents reality correctly and completely.
- Pragmatic quality: The domain model is easy to understand from the stakeholders’ perspective.

We used basic notation elements of UML class diagrams in accordance with the UML. Adherence to the syntax was ensured by using tool support for domain modeling.

To assess the perceived semantic and pragmatic quality, we conducted a qualitative survey of the participating domain experts.

The evaluation of semantic quality was completed by comparing our domain model with an existing ontology of the domain to assess the congruence of identified concepts with established research.

For our written survey (adapted from [152]), we received answers from three of the four participating domain experts as shown in table 3.4, where the numbers indicate the number of responses in the respective category.
<table>
<thead>
<tr>
<th>Question</th>
<th>Rather</th>
<th>Rather</th>
</tr>
</thead>
<tbody>
<tr>
<td>It was easy for me to understand what the model was trying to model.</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>The model represents the domain correctly.</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>The model is a realistic representation of the domain.</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>All the elements in the model are relevant for the representation of</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>the domain.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The model gives a complete representation of the domain.</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>The model contains contradicting elements.</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
The domain model was evaluated to give a rather complete, realistic, and correct representation of the domain. The only concept which was identified as missing was “criteria of potential”. Within the interviews we conducted, the topic of potential was only mentioned once as being currently in discussion for implementation, thus did not show to be significantly relevant according to the data. However, as saturation could not be reached, this concept might appear during further analysis.

The only inconsistency which was reported was that performance assessment did not necessarily evaluate target agreements. Domain experts’ descriptions of the relationship between competency, performance, employee assessment, and target agreements were inconsistent and imprecise. Their statements were therefore compared and further investigated in interviews, which resulted in the distinction between competency and performance assessment and a defined relationship between performance evaluation and target agreements. However, the inconsistencies and imprecisions in the data were not completely resolved because saturation could not be reached and would need to be investigated further with additional interviews. The received feedback suggests that regular validation of analysis results should be part of the domain analysis process to improve the quality of the domain model.

The domain experts were undecided if all elements in the domain model were relevant for the representation of the domain. This was to be expected as the evaluation of relevance depends on the purpose of the domain model and the desired level of abstraction. These concerns also led experts to be undecided whether the domain is represented correctly.

Answers regarding the perceived pragmatic quality varied. The domain model was perceived as confusing by some of the domain experts. This might be attributed to the challenge of identifying the optimal abstraction levels within the code system to be matched to the conceptual model and presents an opportunity for improvement in regard to the design of the domain model and our analysis method. Specifically, it should be investigated if clearly defined abstraction levels in the code system can help to improve the clarity of the domain model.
To further assess the congruence of identified concepts, we compared our domain model with Schmidt and Kunzmann’s competency-based ontology of HR development \[175\]. While the ontology only covers HR development with regard to competency management, all participating domain experts stated that performance management was also a part of HR development and our analysis showed a close interrelationship between these two sub-domains. Thus, our domain model provides a more holistic representation of the domain. In comparison, our domain model covers 70% of the concepts from the ontology, while 50% of the competency-related classes (excluding sub-classes) from our domain model are represented in the ontology.

However, the identification of equivalent concepts was based on our interpretation of the ontology concepts by their name, because Schmidt and Kunzmann do not provide definitions of their concepts. This highlights the value of creating a glossary to provide a thorough understanding of the identified concepts. Using our method, concepts and their definitions are developed simultaneously and directly linked, which ensures consistency between the domain model and the glossary.

### 3.4 Domain Modeling for Qualitative Research

The goal of our fourth study was the application of our method for the domain analysis and requirements elicitation for tool support for qualitative analysis methods such as employed in our domain modeling method. The study was conducted in collaboration with Schmitt \[177\].

This study had a focus on the integration of multiple views in code systems (conceptual view, process view, natural language requirements specification, etc.). The specific research question of how various views or model types can be integrated into our unified model is beyond the scope of this thesis. We therefore describe this study as another application of our method with a conceptual domain model as a resulting artifact which was evaluated through expert feedback.
3.4.1 Data Sources

The data source for this project was a series of expert interviews. In total, five interviews with five stakeholders were carried out. Four of them were professional researchers from social sciences. The goal of these interviews, besides the creation of a conceptual domain model, was to generate a theory on how social science researchers perform theory building, with a focus on QDA and specifically on how they perform the task of coding.

Three of the four researchers we talked to perform social science research and use QDA methods. All of them are experienced researchers and employed QDA methods in multiple research projects. One of them holds a doctor’s degree, the other two are Ph.D. students and hold master’s degrees or equivalent titles. The fifth interviewee was a software engineer working on QDA tooling.

3.4.2 Data Analysis

As with the other exploratory studies, we let the code system structure emerge out of the data. Hence, we initially kept very close to the GT guidelines.

All interviews were performed in a semi-structured way. The intention was to let the interviewees speak freely. This eases the discovery of topics that the analyst might not yet be aware of. However, when statements came up we did not understand or included unclear details, or when the current interviewee contradicted statements from earlier interviews, we asked for more details about these topics.

During the first iterations’ axial and selective coding steps, the code system developed into a direction that might be the expected outcome of a conventional analysis as well: Since the purpose of this project was the creation of a domain model and a requirements specification through QDAcity-RE, the structure split up into separate parts for the social science domain analysis and for the requirements analysis. Issues like the project steps for the former and development constraints for the latter emerged out of the data as core concepts. These concepts naturally became superordinate codes in the code system.
This may partly result from the interview outlines of the initial interviews, since they addressed questions like “What project phases are there in a research project when you apply QDA methods?”.

The speech of both interviewer and interviewee were transcribed word for word by ourselves. We excluded accentuation of statements and non-verbal communication. Furthermore, we left out speech parts that only stated an expression of comprehension, which would interrupt the flow of information and make the text more difficult to read. In some cases, we decided to directly skip short passages in statements where the speaker misspoke and corrected himself. Some expressions of colloquial speech or dialect were transformed into equivalent standard language expressions.

In order to assess the current state of the code system with regard to its completeness at any given point in time, we assigned a traffic light color to each code within the memo, specifying how well the concept is described in the code system:

- Red: huge gaps of information, lots of data missing
- Yellow: contradictions within related codes, information gaps, open tasks for this chapter
- Green: chapter is complete, no questions, requirements are written down, tasks are done

We denoted the specifics of the gaps, problems, contradictions, thoughts or questions within the code memos. Hence, they provide both a quick overview of the status and a flexible way to denote issues.

3.4.3 Evaluation of Qualitative Research Study

We asked the domain experts to give feedback on our results by participating in a written survey.
Two of the experts provided feedback, which was positive. One of them pointed to a missing detail. Beyond, they agreed to our results (see table 3.5).
<table>
<thead>
<tr>
<th>Question</th>
<th>Rather</th>
<th>Rather</th>
<th>Can’t be</th>
</tr>
</thead>
<tbody>
<tr>
<td>It was easy for me to understand what the model was trying to model.</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>The model represents the domain correctly.</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>The model is a realistic representation of the domain.</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>The model gives a complete representation of the domain.</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The model contains contradicting elements.</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
During the evaluation of the previous study, we assumed that some of the “undecided” answers were the result of a lacking context with regard to the purpose of the model. We, therefore, added the category “can’t be assessed” to allow for this distinction to be made explicit.

3.5 Limitations of Exploratory Cases

3.5.1 Application Domain

During our exploratory studies, some important differences between traditional GT and its application to domain analysis became apparent. The most significant difference is the focus on either behavioral or structural aspects.

GT focuses on interaction systems and therefore mainly uses concepts to describe dynamic aspects of a research area, i.e. actions which indicate a phenomenon \([40, 135]\). In their seminal work “The discovery of grounded theory”, Glaser and Strauss write that “Although this method can also be used to generate static theories, it especially facilitates the generation of theories of process, sequence, and change pertaining to organizations, positions, and social interaction” \([74]\). The core categories of the analysis are also sometimes called Basic Social Process or Basic Social Psychological Process \([73]\).

A conceptual domain model, however, describes the important entities of a problem domain and their structural relationships \([22]\).

Although the data sources used for domain analysis, such as domain expert knowledge, contain mostly dynamic descriptions of the domain, an analysis method must provide a way to extract structural entities \([169, 208]\). These structural aspects need to be described with their attributes and related to each other, the same way phenomena under study in GT are represented with categories and properties.
3.5.2 Sampling and Saturation

In all four cases, theoretical sampling could not be fully applied due to limited access to interview partners. Especially when we interviewed domain experts from different companies, we had to start each interview with basic questions to understand the specific context within the company. Thus, the interviews provided rather high level information.

To address a lack of detail in certain cases we conducted follow-up interviews to retrieve more detailed information. However, theoretical saturation could not be reached due to availability constraints of domain experts, and time constraints for each of the projects.

3.5.3 Lack of Tool Support

Due to a lack of dedicated tool support for parts of the method, assumptions we have on the effectiveness of the method regarding the effort needed for its execution can not be validated.

While the activity of coding the data is supported by existing tooling which we employed in our cases, these tools do not adequately support the documentation of machine readable meta information required for our method. Thus many of the process steps were executed using a pen and paper method of tracking all links between the documents. This is an error prone process and also distracts from the actual task. We later addressed these concerns with our own tool support.

3.6 Evaluation of Exploratory Studies

Table 3.6 provides an overview over the four studies with regard to our evaluation model presented in in table 3.3.

All four of our studies confirmed an excellent level of documentation for the analysis process. This documentation includes traces from each model element back to individual stakeholder statements and documentation. The documentation also includes the reasoning for most interpretations in short memos. A researcher not involved with the coding of the data
Table 3.6: Study Evaluation.

<table>
<thead>
<tr>
<th></th>
<th>Medical Image Diagnostics</th>
<th>Railway Systems</th>
<th>Human Resource Development</th>
<th>Qualitative Research</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Documentation</strong></td>
<td>- Partial Traceability</td>
<td>- Full Traceability</td>
<td>- Full Traceability</td>
<td>- Full Traceability</td>
</tr>
<tr>
<td></td>
<td>- Memos</td>
<td>- Memos</td>
<td>- Memos</td>
<td>- Memos</td>
</tr>
<tr>
<td><strong>Completeness</strong></td>
<td>-</td>
<td>-</td>
<td>- Expert Survey</td>
<td>- Expert Survey</td>
</tr>
<tr>
<td><strong>Consistency</strong></td>
<td>-</td>
<td>- Problems</td>
<td>- Problems</td>
<td>- Problems</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Systematically</td>
<td>Systematically</td>
<td>Systematically</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Documented</td>
<td>Documented</td>
<td>Documented</td>
</tr>
<tr>
<td></td>
<td></td>
<td>and Resolved</td>
<td>and Resolved</td>
<td>and Resolved</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Expert Survey</td>
<td>- Expert Survey</td>
</tr>
<tr>
<td><strong>Input Types</strong></td>
<td>- Expert Interviews</td>
<td>- Expert Interviews</td>
<td>- Expert Interviews</td>
<td>- Expert Interviews</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Norms</td>
<td>- Workshops</td>
<td>- Workshops</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Drawings</td>
<td>- Drawings</td>
</tr>
<tr>
<td><strong>Model Types</strong></td>
<td>- Feature Model</td>
<td>- Conceptual Model</td>
<td>- Conceptual Model</td>
<td>- Process Model</td>
</tr>
<tr>
<td></td>
<td>- DSL</td>
<td>- Glossary</td>
<td>- Conceptual Model</td>
<td>- SRS</td>
</tr>
<tr>
<td></td>
<td>- Conceptual Model</td>
<td></td>
<td>- Conceptual Model</td>
<td>- Conceptual Model</td>
</tr>
</tbody>
</table>
could easily assess the relevance of different aspects to the group of interviewed experts.

The first two studies did not yield an explicit evaluation of the resulting artifacts against a fixed reference point or through an objective third party. However, these studies were useful in exploring the viability of the approach and identifying pitfalls that had to be addressed by our method. They also confirmed our assumptions that, using QDAcity-RE, it is possible to use the code system as a unified model that connects different target artifacts through inter-model traces which makes navigating between the different artifacts easier and also fosters inter-model consistency. To this end, the first study derived a DSL, a feature model, as well as a conceptual model, from the same code system, while the second study did the same for a conceptual model and a glossary. The fourth project also contributed toward this dimension of evaluation by combining the extraction of a RS in natural language with the creation of a process model and a conceptual model from the same data set using the same code system. In this last case, the code system had to be adapted with a predefined structure on the top two levels to make part of it specific to NL requirements documents.

The artifact quality concerning completeness and consistency of the created models was evaluated through surveys with domain experts conducted within the 3rd and 4th study as well as through a comparison with an ontology that has been established independently from our research.

All of the four studies used expert interviews as a means of data collection, supplemented by a variety of other materials such as drawings, norms and regulations, formal documentation, and workshop transcripts. While our studies indicate that all of these types of materials can be analyzed using the same method it is our finding that our method provides more value the less structured the data is. This correlates with our finding, that a bottom-up or middle-out approach suits our method better than a top-down approach. This also coincides with the most common use of QDA in qualitative research that, to a significant extent, can be considered inductive theory building.

However, we also found indications in the literature, that similar approaches to ours some-
times benefit from more structured input data. For instance, Rashid et al. used Incident Fault Trees (IFT) \[101\] to elicit security requirements \[157\]. Rashid et al. chose IFTs over other types of more unstructured documentation with the goal to keep the focus on explanation rather than description. The value of a GT based method in this scenario stems in large part from the constant comparison method across a multi-incident analysis, and less from the in-depth analysis of a single incident.

3.7 Conclusion

We presented four exploratory cases for qualitative theory building, refining our method, which is presented in the following chapter 4.

During our exploratory research, we found that the coding procedure supported the structuring and analysis of qualitative data for conceptual domain modeling. Important concepts became apparent already early in the coding process. This was also the case for data that emphasized process descriptions, which interviews tend to favor. The participating domain experts primarily gave an account of their domain from a process point of view. Through the development of concepts and categories, the structural aspects emerged and could be further investigated through theoretical sampling. Inconsistencies could be investigated by comparing the respective data fragments and notes could be taken in code memos about questions that need to be asked in the next interview and about the different options of interpretation. This was especially important for integrating company-specific descriptions of HR development into a consistent domain model.

The systematic coding procedure and the writing of memos make the process of domain analysis traceable, but coding and modeling decisions are still interpretive and therefore depend on the analyst’s experience and expertise. What to code and how to develop concepts into categories is a difficult task for which there is no one simple solution. We found that abstracting too early in the process or focusing too much on the domain model while coding can make later changes more difficult. However, the analysis method provides more guidance to
a novice analyst for extracting a domain model. Systematic coding helped us to engage with the domain to be analyzed, where previous domain knowledge was limited.

Although theoretical sensitivity also depends upon the researcher’s level of experience in qualitative research and the phenomenon under study, it develops further during the research process and can be enhanced using techniques for questioning the data or systematically analyzing a word or phrase and comparing different incidents [39, 83, 108]. This suggests that while a requirements engineer still benefits from his or her experience in the domain under study, a systematic analysis procedure can support him or her to develop theoretical sensitivity with regard to domain analysis.

The practices of constant comparison, theoretical sensitivity, and questioning of the data can also help to prevent experienced analysts from prejudiced misconceptions. On the other hand, we experienced the coding process as time-consuming and requiring a high cognitive effort, in accordance with many of the authors of related work.

We’ve shown, that QDAcity-RE integrates multiple different types of input data well (interview transcripts, workshop transcripts, documentation, etc.) and that the code system may serve as a unified model for other artifacts besides the conceptual model (behavioral models, feature models, DSLs, etc.). However, we found that the less structured the input data is, the more value exists in the application of our method.

Our studies suggest that our analysis method favors a bottom-up or middle-out approach and provides less additional value for a top-down analysis approach.
QDAcity-RE [107] is a method for domain analysis. The analysis process codified by this method has the goal of creating a code system through iterative refinement from which the domain model is derived.

The code system is a unified model that bridges the gap between stakeholder materials in natural language and more formal models like requirement engineering artifacts. The code system is described in chapter 4.3, and subsequently the process is detailed in chapter 4.4.

4.1 Method Overview

The domain analysis is performed in an iterative fashion. The main artifact of the analysis, the code system, is incrementally refined until so-called saturation is reached. Saturation, that is sufficient completeness, is reached when the code system does not change significantly with the addition of another iteration of stakeholder sampling, data gathering and analysis.

Each of the iterations consists of the following three steps:
1. Stakeholder sampling

2. Data gathering

3. Data analysis

The sampling of new data is driven by gaps and inconsistencies explicitly documented in the current state of the code system representing the results from all previous iterations. While the means of data gathering, such as interviews, workshops, and legacy documentation, are not exclusive to our method, there are specific characteristics of data and techniques of data gathering that QDAcity-RE suits more than others. These characteristics are discussed in chapter 4.4.2.

The analysis is then performed by qualitative coding of the data in three coding steps, called open, axial, and selective coding.

The goal of our analysis process is to make the previously implicit and largely undocumented interpretations and decisions made during domain analysis explicit. The manifestation of this explicit documentation is the main artifact of our method: the code system.

Figure 4.1 provides an overview of how to perform domain analysis using our method.

4.2 Method Context

Our method is primarily aimed at environments where products have a long life-cycle and thorough documentation is needed or even mandated. If it is foreseeable that the system will be replaced by something completely different in the near future, the benefits of our method may not outweigh the increased effort it requires. However if the product will evolve, then the documentation and the traces back to stakeholder statements are a major benefit. The documentation and traceability make it easy to verify if specific concepts are still relevant and why they were modeled in the first place.

Furthermore, in a context where the main sources of information are expert interviews or other highly unstructured information sources, our method is significantly more helpful
Figure 4.1: The QDAcity-RE Process for Structural Domain Modeling

when compared with situations in which the information sources are already highly structured, for example, situations with existing specifications. In principle, all types of data gathering are supported, and the sources can be combined and correlated. To combine different types of data using data triangulation is recommended.

Our method focuses on the data gathering and analysis process. It is assumed that the project vision and project scope have already been defined, although the scope may be refined through the iterative analysis process, if necessary.

4.3 The Code System

The code system is a hierarchical structure of codes. It represents a model that captures concepts, categories, their properties and interactions. The code system is produced as a result of the coding process of QDA, which is detailed in section 4.4.3.

Each code in the code system describes a concept with evidence in the gathered data. When codes are first created during the analysis process, they are loosely coupled. With an increasing number of iterations, the code system is formed as a hierarchical structure of these codes as nodes. Each code has a label, a definition, and instructions on its intended use as well as a
description differentiating it from other codes and possible misinterpretations. In addition to this information, which is typically documented in a codebook [124], each code contains meta information on which kind of entity it is (activity, actor, etc.).

The code system bridges the gap between natural language text containing stakeholder information and requirements engineering artifacts like the analysis domain model. Each code is linked to one or more text segments in the gathered data. These instances of a code are called a coding, and the length of the coded text segment is called the unit of coding (sentence, paragraph, multi-paragraph, etc.).

Since the code system provides a holistic view of the phenomenon, it acts as a common denominator between multiple models describing different views on the domain (i.e. structural, behavioral, data, communication, etc.). This strengthens inter-model consistency. In our exploratory studies, for instance, we derived both a UML class diagram as well as a BPMN diagram from one code system, and we derived a feature model, class diagram, and domain specific language (DSL) from another code system with all model elements traceable to the code system, linking to elements from various model types.

An example of a code system excerpt from one of our studies with its traces to associated stakeholder statements and parts of the model is presented in figure 4.2. The granularity of the traces back to the stakeholder interviews can be varied through the unit of coding. A
typical unit of coding ranges from a part of one sentence to multiple paragraphs.

4.4 The QDAcity-RE Process

The analyst starts the domain analysis process by constructing a broad initial interview guideline.

This initial guideline evolves with each iteration of data gathering and analysis to close knowledge gaps identified through previous iterations and to resolve inconsistencies. Increasing the specificity of the guideline does not necessarily lead to a structured interview. Still, structured interviews or even questionnaires can cover information that requires a larger empirical sample. For this, they can provide valuable supplementary material to strengthen the validity of the findings through method or data triangulation. Triangulation is a term describing a set of practices used to vary different aspects of the analysis to gain insights on the phenomenon from different perspectives. If different perspectives on the phenomenon lead to the same conclusion, the analysis result is believed to be more credible. The increased credibility is assumed because the analysis is grounded in different types of data (data triangulation), or because the data was analyzed through different types of activities (method triangulation) or by different investigators (investigator triangulation). A fourth form of triangulation is considered when the results were verified by people with an external perspective on the research project using the same data (theory triangulation).

While triangulation is not a required part of the QDAcity-RE method, it is a practice that lends itself well to domain modeling as well and can easily be integrated.

4.4.1 Stakeholder Sampling

With the interview guideline ready, the analyst then performs stakeholder sampling to find those individuals who are best able to discuss the topics of the interview guideline. The sampling strategy is not fixed for the entirety of the analysis, but only for the next iteration. It has to be performed with consideration for what the gaps, inconsistencies, or novel insights
are that are present in the current state of the code system.

The sampling strategy is called theoretical sampling because it draws on the current state of the theory on how the phenomenon could be modeled as supported by the evidence already collected and analyzed at the time of sampling. This information is codified in the code system. Theoretical sampling promotes a more flexible and agile way of sampling, as opposed to defining the data gathering process a-priori based on assumptions about the domain and the different stakeholders. The data gathering is not performed in a predefined order.

In qualitative research, theoretical sampling is often considered the ideal way to shield the outcome from being influenced by preconceptions [194] and to highlight information gaps [30].

4.4.2 Data Gathering

During the data gathering phase of each iteration, the analyst extracts and documents unstructured information from stakeholders’ both explicit and implicit knowledge. The goal is to document as much information as possible in a way that allows for a structured analysis leading to consolidation with other materials and abstraction into uniquely identifiable pieces of information that describe specific parts of the domain.

QDAcity-RE can be paired with a wide range of methods for collecting unstructured data from stakeholders, such as workshops, interviews, observations, surveys, or creativity techniques such as brainstorming. Our method treats all data the same and unifies the information content of different media types in a single artifact, the code system, which encompasses the consolidated information from different input artifacts.

All of these data gathering methods can be used in our method, however, interviews take the most prominent role. The coding of transcribed interviews using QDA can provide the highest added value. We have found that our method provides the highest value in a context, where the majority of the information is available only in unstructured form. Interviews document the stakeholders’ thoughts in a way that supports an unbiased analysis through a third-
party analyst. Through the analysis, the information within the interviews gets structured and becomes easier to navigate. In contrast, workshops, for example, often require more structured moderation in order to coordinate a larger group of stakeholders. This added structure, however, lessens the value of QDA, because the analyst is most likely to just follow this structure in his analysis, preventing a possibly more natural structure to emerge from the data.

Interviews and workshops are transcribed and can be coded together with legacy documentation and regulatory texts.

Throughout the whole process, the analyst has to be open to new ideas emerging by letting the gathering process be steered in large parts by the stakeholders. This concept is called theoretical sensitivity and it helps to identify what is significant to the interviewee without being biased by preconceived notions.

Within the scope of this work, we use interview transcripts, marketing material, natural language documentation, regulatory documents, and photographed hand-drawn illustrations as input data for our exploratory projects.

4.4.3 Data Analysis

The analysis of the gathered data is driven by the coding process. During coding, the gathered data is annotated to highlight the most insightful parts of the text, resulting in a code system. The code system is a hierarchically structured set of codes, representing common concepts, that connects unstructured data to structured RE artifacts, such as the domain model and the glossary. The code system thus ensures inter-model consistency.

The coding process consists of the following three activities, which are performed in sequence.

1. Open coding

2. Axial coding
3. Selective coding

These three activities are performed during each iteration of data gathering and analysis until a stopping criterion, the so-called \textit{theoretical saturation}, is reached. Reaching the stopping criterion indicates sufficient completeness of the analysis results.

It is important to limit the amount of data that is added in each iteration. Especially the first coding step, open coding, becomes more difficult to perform as the amount of new data in each iteration increases. In a research context, typical increments add new data collected from 1-5 interviews for each iteration. For our exploratory projects, we added one interview per iteration because of the relatively small scope of each project.

During the first stage of coding, open coding, the analyst creates an unsorted list of labels and assigns each label to one or more text segments. These labels are called \textit{codes} and the portion of data that has been coded is called a \textit{coding}. The granularity of the coded segment is called the \textit{unit of coding} and may vary from single words, sentences to multiple paragraphs or pages.

Codes are referred to as in-vivo codes \cite{30, 39}, if their name is directly mentioned in the unstructured data, but codes can also be abstractions from the original material. Because a domain model should represent the domain terminology \cite{117}, in-vivo codes should be the most common codes. Synonyms should be documented with the code and ultimately in a glossary. It is common, that in the beginning hundreds of open codes are created on a multitude of abstraction levels. We advise to first generate specific concepts for smaller units of coding and then to combine them during the abstraction process. Specifically, when coding within the context of domain analysis, it is important to make all aspects of a phenomenon explicit in separate codes. A code “employee attends development measure” is not easily mapped into a domain model, because it includes several aspects: the actor “employee”, the activity “attending development measure”, and the event “development measure”. In addition, the analyst should be careful to describe activities with verbs and not with nouns in order to distinguish them from events.
The extracted codes are then structured hierarchically by grouping them into categories during axial coding to form a map of concepts supported by the analyzed documents. Categories represent the aspects central to the domain and are described further with regard to their properties and context through constant comparison and questioning. The data fragments indicating the properties should also be coded. Both structural and dynamic aspects can be developed into categories. However, if the purpose of the analysis is clear, such as the extraction of a conceptual domain model, the analyst may focus on aspects that are central for the analysis and investigate these first. During the axial coding step, the code system language (see section 4.5) is used to define the types of relationships that may be modeled within the code system.

The last step in a coding iteration, selective coding, helps model only aspects within the scope of the project. The selection and focus on a few high-level phenomena reflect what is central to the domain, what belongs to it to support the central concepts, and what is not part of the code system and consequently not significant for the domain.

During this coding step, core categories are chosen, which holistically describe the studied phenomenon. All other codes have to be subsumed by a core category. The code system should describe core categories in all of the following five dimensions to be considered complete:

1. Actions & strategies
2. Consequences
3. Causal condition
4. Contextual condition
5. Structural condition
This is borrowed from social science research, where these dimensions form the cornerstone of the coding paradigm [40]. The coding paradigm aims at increasing the systematization of the coding process [195].

Codes that do not fit in any core category are considered not relevant. While in research this criterion of relevance is highly influenced by the research question as well as the domain, for domain modeling it is purely dependent on the domain since the question constituting the reason for the analysis is always “What are the concepts in this domain and what are the relations between them?”.

In theory building research, all categories of the code system will ultimately be subsumed under a single core category. However, selecting a single core category does not make sense for the purpose of domain analysis, because a domain model should give a complete representation of the domain [18]. The phenomena, i.e. entities, which are central to the domain have already been identified as being important by developing them into categories.

4.4.4 Iterative Refinement

After the initial collection and analysis of data, the analyst enters an iterative process repeating the concurrent collection and analysis of data, where the sampling of new data should be sensitive to how the code system evolves.

After each iteration of data collection and analysis, the analyst should reevaluate what data should be collected next based on the current state of the emerging domain theory. The current state of the code system and lacking description of the core categories thereby drives the sampling process for the next iteration of data gathering and analysis.

New iterations are performed until saturation reaches a defined level. Full saturation is reached when through the additional gathering of materials selected through theoretical sampling, no new codes in the code system emerge, and the definitions of existing codes remain stable.

Although the concept of theoretical saturation is frequently suggested [20, 80, 65], the
metrics for measuring saturation are rarely documented. An overview of the problem of
data saturation is presented by Francis et al. [67]. They propose to start with an initial analy-
sis sample and a stopping criterion, which is the number of consecutive interviews that have
to be analyzed following the initial analysis sample, without new themes emerging from the
data. Both measures have to be defined a-priori, depending on the complexity of the studied
phenomenon. They conclude, that a 10 + 3 (initial + stopping) rule for their saturation crite-

rion may be regarded as a reasonable value if there are no specific indications on the required
sample size within the problem domain.

Whatever saturation metric may be considered adequate in a specific case, it is imperative
in any case that the measure be clearly documented and consistently measured. Although
the analysis process can be used without measuring saturation, we advise defining an explicit
criterion that is actually measurable and to track this metric throughout the lifespan of the
project.

Using the concept of theoretical saturation yields a metric for the quality criterion of com-
pleteness. Further, the hierarchical structure of the code system assists the analyst in identi-
fying conflicts and contradictions. This hierarchical structure is a direct result of the coding
process and since concepts describing the same semantic entity will be located in close prox-
imity within the tree structure, inconsistencies will be easier to identify.

During each new iteration, the analyst is required to look for evidence or contrary indica-
tions of newly emerging codes in already coded documents. He or she is also required to look
looking for evidence of established codes in the new data and to identify new concepts that
previously were not prevalent. This behavior is called constant comparison.

The code system is further refined iteratively while finding support for a theory, establish-
ing new codes, or combining and eliminating codes that are not sufficiently supported by the
data.

Throughout the whole analysis process, the documentation of the data gathering and anal-
ysis process plays a vital role. One common practice for facilitating such documentation
in qualitative research is memo writing. Memos can be attached either to specific codes or as project memos on the analysis process to the whole code system or specific documents. Memos are thus important for describing the meaning of different concepts within the domain and explaining the decisions within the analyst’s mental process. In current tool support for QDA, memos are also the only way of describing more expressive relationships between concepts. This is one aspect we want to improve about current qualitative research processes: We want to make information which researchers usually only write up informally, explicit, and machine processable by using more convenient and reliable means than natural language processing (NLP).

4.5 Codesystem Language

To facilitate the transformation from code system to conceptual model, the codes within the code system need to be typed in accordance with a code system language.

As presented in section 4.3, the code system represents a unified model where the conceptual model is merely one possible presentation. Although our vision of a single unified model partly motivates our code system language, we have only experimented with its use for conceptual and certain types of behavioral domain modeling. Our vision is that this meta model will be extended and adapted to different use-cases.

Reuter investigated different ways in which pre-defined structured knowledge, such as a type-system, or meta-model, for codes, can be used to aid the development of new code systems in domains that are, in part, already well-understood [161]. Using the code system language is a restriction in the freedom of applying and structuring codes as the process is usually described for qualitative research. However, also in a research context Salinger et al., among others, found that the lack of any restrictions or structural framework can be a hindrance in creating theories in an acceptably efficient manner [172]. The solution can be some form of predefined coding scheme and naming conventions, or as we present, a meta model. In fact, in the discipline of method engineering, having constructs of a method explicitly represented
in a meta model is encouraged [183].

The model of the code system language is presented in figure 4.3.
Figure 4.3: Code System Language (CSL).
Conceptually there are two distinct types of codes. Codes that describe concepts/aspects and codes that describe relationships. We call the latter relationship codes. Even though relationships between codes can be modeled in the meta attributes of any code (that is not a relationship code), without the relationship itself being a code in the code system. However, sometimes it is necessary to trace the origin of a relationship back to segments of data. In this case, it makes sense to also model the relationship as a code.

Consequently the CSL attributes of type Relationship are mutually exclusive with the types Concept, Property and Aspect.

Further, if a code describes an Aspect, then it can either be of type Concept or Property, not both.

For the mapping to a conceptual domain model, not all facets of the CSL are required, but we’ve used aspects such as Activity and Process to generate behavioral views on the code system.

More constraints on the application of the CSL to a code system are presented in table 4.1.

We’ve evaluated the CSL using the data from the exploratory studies, as well as from a data set on inner source [26], which a student was tasked with creating a conceptual domain model using the interview data in the context of a small scale research project within the scope on a course on research methods.

Our vision is that through the consistent use of the CSL, all code systems created within QDAcity would be interconnected, which may allow our system to make helpful suggestions or even allow to answer questions beyond the individual research projects. In a future extension of our tool, we conceive that users will be able to define their own meta model as an extension of the CSL to represent their domain more specifically and eventually share this extension with colleagues or students working in the same field.

The CSL and its mapping to a UML class model have in part been supported through Master’s theses by Kunz and Salow [116, 173].

71
### Table 4.1: Application Rules for CSL

<table>
<thead>
<tr>
<th>Condition</th>
<th>Application Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label = = (Category OR Concept)</td>
<td>Aspect cannot be Activity</td>
</tr>
<tr>
<td>Label = = (Category OR Property)</td>
<td>no is part of allowed</td>
</tr>
<tr>
<td>Label = = Concept</td>
<td>at least one of (is a, is part of)</td>
</tr>
<tr>
<td>Aspect = = (Activity OR Process)</td>
<td>at least one of (is consequence of, causes)</td>
</tr>
<tr>
<td>Aspect = = (Object OR Actor OR Place)</td>
<td>no is consequence allowed</td>
</tr>
<tr>
<td></td>
<td>no causes allowed</td>
</tr>
<tr>
<td></td>
<td>no influences allowed</td>
</tr>
<tr>
<td>Aspect = = Actor</td>
<td>mandatory performs</td>
</tr>
<tr>
<td>Aspect = = Actor</td>
<td>no performs allowed</td>
</tr>
<tr>
<td>Label = = Property</td>
<td>mandatory is related to</td>
</tr>
<tr>
<td>AND Aspect = = (Object OR Actor OR Place)</td>
<td></td>
</tr>
<tr>
<td>Label = = Property</td>
<td>mandatory influences</td>
</tr>
<tr>
<td>AND Aspect = = (Activity OR Process)</td>
<td>no is related to allowed</td>
</tr>
</tbody>
</table>
4.5.1 Mapping to Conceptual Model

A mapping rule consists of the following three components.

- The type of code (CODE or RELATION)
- A condition that has to be met for the rule to fire
- An action describing what type of model element to create

If the code is not of type CODE, then the condition relies only on the CSL attribute of the code in question. If it is of type RELATION, then it also draws on CSL attributes of the Source and the Target of the relation.

The mapping from CSL to a UML class diagram is shown in table 4.2.
Table 4.2: Mapping from CSL to UML Class Diagram.

<table>
<thead>
<tr>
<th>Type of Code</th>
<th>Condition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>CODE</td>
<td>(Code.CSLAttribute == Category)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Code.CSLAttribute == is-a) &amp;&amp;</td>
<td></td>
</tr>
<tr>
<td>RELATION</td>
<td>((Source == Category)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>((Target == Category)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Code.CSLAttribute == is-part-of) &amp;&amp;</td>
<td>Create aggregation</td>
</tr>
<tr>
<td></td>
<td>((Source == Category)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>((Target == Category)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Code.CSLAttribute == is-related-to) &amp;&amp;</td>
<td>Create directed association</td>
</tr>
<tr>
<td></td>
<td>((Source == Category)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>((Target == Category)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Code.CSLAttribute == is-related-to) &amp;&amp;</td>
<td>Create class field</td>
</tr>
<tr>
<td></td>
<td>((Source == Category)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Target == Property) &amp;&amp;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>((Target == Actor)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Code.CSLAttribute == influences) &amp;&amp;</td>
<td>Create class method</td>
</tr>
<tr>
<td>RELATION</td>
<td>((Source == Category)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Target == Property)</td>
<td></td>
</tr>
</tbody>
</table>
5

QDAcity:

A QDA Tool Supporting QDAcity-RE

5.1 Motivation

While the exploratory studies showed the general feasibility of the QDAcity-RE method, it also revealed clearly, that the time consuming nature of the method is significantly exacerbated through the lack of tool support and the consequential need for manual documentation of all required analysis steps. Further, the comprehensive documentation, which the method produces, is only of value if it remains consistent with changes throughout the whole process. Maintaining this consistency as a manual task is highly error-prone.

We, therefore, decided to implement our own software supporting the qualitative analysis, allowing us to further process the analysis artifacts and implement additional views on the data. This leads to a consistent universal model of the data in the code system and a specific
visual representation for the conceptual model in form of a UML class diagram. This way, if any of the data is changed, either during the traditional QDA process or in the UML view, the change can be documented and reflected consistently in all relevant components.

We further decided to emphasize the collaborative aspect of QDA and research in general, by enabling users to share project access and even work concurrently on the same data.

A peripheral aspect we pursued with QDAcity is its use case for teaching support, which we have successfully used extensively in our own teaching of QDA. This was supported through a Masters thesis by Ammari [5]. However, since this aspect of our tool is largely independent of its use case for domain modeling it is not elaborated here.

5.2 Architecture and Design

QDAcity is a single-page web application with a JavaScript (JS) client running in the browser and three backend services: the Google App Engine (GAE) backend service, the real-time collaboration service (RTCS), and the auth proxy service.

We decided to build our tool as a progressive web app (PWA). PWAs are built on a set of technologies to make them behave like a native app on a variety of different runtime environments, without having to port the application for each platform [16, 125]. PWAs, like native applications, usually make use of a local database to provide functionality even without a connection to the server. Our implementation of this feature is elaborated in section 5.3 on the offline experience.

An overview of the distribution of the different execution environments used for QDAcity and how they are interconnected is presented in the deployment diagram in figure 5.1.

As illustrated in figure 5.1, the client running in the user’s browser has two ways of communicating with the backend: either through HTTPS requests, which are routed through the service worker or through a websocket connection, where the RTCS acts as a proxy to the GAE backend service and ensures consistency between all connected users of a project who work concurrently.
We describe the GAE backend service, RTCS, auth proxy service, and the frontend client in the following subsections.

5.2.1 GAE Backend Service

The GAE backend service provides a RESTful API to the client. The service is written in Java 8 and runs on Google App Engine \(^1\) standard environment. It is designed to be horizontally scalable and running on instances with less than 300MB of memory. It is connected to a serverless NoSQL document database in the Google Cloud infrastructure, the Cloud

---

\(^1\)https://cloud.google.com/appengine/
Firestore in Datastore mode\(^2\), where all application data is being stored. To minimize vendor lock-in, JDO\(^3\) is used for most object-mapping from our data model to Google Cloud Datastore.

The REST API is created with Google Cloud Endpoints, which automatically creates a discovery document for the API which is used by the client library, as well as the service worker of the client component, which acts as a proxy for the backend system.

Besides managing CRUD operations on the database, the GAE backend service is also home of the metrics component, which can be used to calculate intercoder agreement metrics using the F-Measure\([202]\), Fleiss’ kappa coefficient\([64]\), and Krippendorff’s alpha coefficient\([113]\).

We also implemented a saturation metric based on the project’s edit history. As Guest et al. point out, even though the concept of theoretical saturation is frequently described as the gold standard for determining the sample size for purposive samples in qualitative studies, the operationalization of it frequently remains vague\([80]\). This correlates with our subjective observation. We hypothesize that this is partly due to lacking tool support. For the QDAcity-RE method, we integrated an operationalization of this metric into our software to automate the generation of the saturation metric. We chose to base our implementation on the suggested operationalization of theoretical saturation by Francis et al\([67]\). Our implementation is based on change logs generated through user interaction with the data. The changes are classified into categories like *code edited*, and our algorithm is highly configurable on how each category, or a specific type of change, should be weighted, if considered at all. A more detailed description of the metrics component can be found in section 5.4.

### 5.2.2 Real Time Collaboration Service

In our pursuit to enable a collaborative workflow on a QDAcity project, we faced the following challenges:

\(^2\)https://cloud.google.com/datastore/
\(^3\)https://db.apache.org/jdo/
1. App engine standard environment for Java did not support websocket connections at the time of implementation.

2. There was expected code sharing between client code and server code. However, the client code was written in JavaScript and the backend in Java.

To address the first issue, we decided to implement the RTCS on its own infrastructure separate from the GAE backend service running on GAE.

To gain full control over the runtime environment, we decided to utilize Compute Engine instances as the infrastructure. This way we could manage our own virtual machines, yet the cloud platform would still allow us to scale our instances using a load balancer. The GCP load balancer supports session affinity to make sure that clients connected to one instance would continually be routed to the same instance for the duration of the session. This is important because otherwise a websocket connection could not be maintained between one specific instance of the RTCS and one user.

To address the second issue, the obvious choice for the programming language of the RTCS is JavaScript running in a node.js environment.

We therefore configured our servers with an nginx reverse-proxy, pm2 process manager, Express web framework, and connecting to clients using the socket.io library using the node.js runtime.

When a user opens a specific project, he or she opens a websocket connection to the RTCS, which will then open a so-called room (a communication channel) for that project and put the user’s connection in it. Other user’s opening the same project, will join the same room, and leave it when the connection is closed or lost.

---

4 https://cloud.google.com/compute/
5 https://nodejs.org
6 https://nginx.org
7 http://pm2.keymetrics.io/
8 https://expressjs.com/
9 https://socket.io/
The RTCS uses the same editor library as the client and holds the authoritative copy of each document. When a user mutates the shared document, a change operation is created which is sent over the network to the RTCS, which then creates a lock on the document, applies the change operation to its own document, and persists the changed document through an update request to the GAE backend service via an HTTPS request. It then sends the change operation to all other connected clients and an acknowledgment message to the client that initiated the change. The lock is then released.

To make this service horizontally scalable, the information on which users are in which room, and the locks on the documents are stored in an in-memory Redis database\textsuperscript{10} provided as a service by redislabs\textsuperscript{11}. This way, it is inconsequential to which instance each user connects to. If another user working on the same project is connected to any other instance of the RTCS, changes to the documents will be distributed correctly to all users currently connected to the projects as well as all instances of the RTCS that are responsible for serving those users.

The implementation of the RTCS was supported through a Master’s thesis by Mischke\textsuperscript{[132]}.

5.2.3 Auth Proxy Service

While the authentication system of QDAcity works through issuing its own JWT tokens\textsuperscript{102} generated and validated using an RSA key pair, the system allows for federated user identity management. This way a user can sign in to a third-party service, such as Google, Facebook, or Twitter to authenticate themselves using their user name and password for the third party service to receive a token containing claims to the user identity. This token is then sent to our main backend service which will validate the claims and issue its own JWT token. Consecutive requests will then be authenticated cryptographically using the self-issued JWT instead of validating the access token from the federated identity provider again.

For any third party authentication provider that supports the OAuth 2 standard, our client

\textsuperscript{10}https://redis.io/
\textsuperscript{11}https://redislabs.com/
can handle the authentication for retrieval of the authentication provider’s token by itself, and the auth proxy is not needed.

However, if the authentication provider only supports OAuth 1, then the request needs to be signed with a secret. This secret should, of course, not be visible to the client. Hence, the request to the authentication provider is first made to our auth proxy service. The auth proxy then signs the request and forwards it to the third party authentication provider. On successful authentication, the token is then handed back to the client. This workflow is illustrated in Figure 5.2.

![Figure 5.2: Successful Authentication in QDAcity With Federated Identity Provider](image)

The auth proxy service is written in JavaScript for the node.js environment and runs on Google App Engine standard environment as a separate service. Since the auth-proxy is only used a single time on login using a federated identity provider that does not support OAuth 2, it is idle most of the time. This was one reason for choosing the app engine standard envi-
ronment since it supports automatic scaling to zero. When the service is not being used, all instances are shut down, and if a user does need it for the login procedure, a fresh instance can be spun up in a matter of seconds and respond to the request. This design decision trades a slightly less responsive user experience in an infrequent scenario for a leaner cost structure. This may be changed through a simple re-configuration of a single YAML file describing the instance type of the service.

5.2.4 Frontend Client

The client running in the browser is written in JavaScript using the React framework. As a single-page app, dynamic routing is handled by react-router. In addition to the JSX syntax of react, which already puts a components’ JavaScript logic in the same file as the HTML representation of the component, we also made use of a CSS-in-JS framework, namely styled-components, so the logic, HTML structure and styling information of a component are not scattered in multiple files.

The client is written in the 8th edition of the JavaScript specification - ECMAScript 8 (ES 8), also called ECMAScript 2017. To add support for older browsers, the source code is transpiled during the build process using babel into ES 5 code.

During the build process, the code is also bundled from over 350 individual JavaScript classes and hundreds of required third-party NPM modules into a single JavaScript package using Webpack.

The application supports multiple locales using react-intl and is currently available in English and German language. All user-facing strings are defined with a default text in the source code and are extracted during the transpilation process into a text file for the English language version using a babel plugin we wrote. This text file was then used as a basis for

---

12 https://reactjs.org/
13 https://www.styled-components.com/
14 https://babeljs.io/
15 https://webpack.js.org/
16 https://github.com/yahoo/react-intl
the translation. All language text files can be written in a simple syntax we defined, and are automatically transformed into the JSON format required by our localization framework in the continuous deployment (CD) pipeline.

A test-job in our CD pipeline will check with each build if there are user-facing strings used in the source code that are not present in all translation files and whether there are translations defined that are not (or no longer) used in the source code.

The localization of the app was supported by the Bachelor’s thesis of Obermeier [146].

**UML Editor**

To integrate conceptual modeling with coding in QDAcity, we integrated a visual editor into the CodingEditor component. The main components involved in this integration are shown in figure 5.3.

![Figure 5.3: UML Editor Components](image)

The implementation is based on the *mxGraph* library [17], which is also the foundation of *draw.io*. We chose mxGraph due to its permissive license, active community, detailed documentation, and the absence of dependencies to large JS libraries such as *lodash* or *jQuery*.

[17]https://github.com/jgraph/mxgraph
We have encapsulated all interaction with the mxGraph library in our GraphView component to create a much smaller interface and retain the option to more easily swap out the library to another or our own implementation. The GraphView component is responsible for drawing the diagram and exposes an interface for adding and removing nodes and edges to the UMLEditor component. For the visual representation for the nodes, we defined our own React components and set them as nodes in the graph.

The UMLEditor implements the facade pattern for the subsystem responsible for visual modeling in diagrams. It is a React component that is positioned hierarchically directly under the CodingEditor.

When a code changes, the CodingEditor triggers the UMLEditor with a codeUpdated(code) call. First, the ConsistencyManager checks via the CodesystemLanguageMapper whether the changed code should be mapped to the diagram based on the code system language (CSL). The ConsistencyManager also checks whether this is a change from the current state. This improves efficiency since not all changes to a code necessarily lead to a need to redraw the diagram. Also, the CSL can in the future be configurable by the user, and we foresee the case where based on a code system with a specific CSL, mappings to different visualizations are possible. We, therefore, decided to encapsulate the logic for mapping codes to diagram nodes into its own component, the CodesystemLanguageMapper. If the ConsistencyManager decides, that a change in the visual representation of the diagram is needed, the UMLEditor will request the change through the GraphView interface.

The scenario where a code is changed that requires a new class to be added is presented in figure 5.4.

The ConsistencyManager is also responsible for making sure, that relationships between the nodes are drawn and deleted as specified in the CSL.

While the CSL itself is drawn from the DB and can easily be replaced or extended, the mapping rules are currently all hardcoded into the source code. For a possible future extension with user-defined CSLs and custom mappings to various diagram types, the definition
Adding a code as new class

Figure 5.4: Communication for Adding a Code as New Class

of rules would need to be refactored.

The implementation of the UML editor was supported through a Master’s thesis by Loos [123].

5.3 Offline Experience

While a web app is well suited to facilitate collaboration between multiple users and has the benefit of not requiring a manual installation on the client system, a clear drawback compared to traditional desktop applications is the requirement for a stable network connection. We developed QDAcity as a PWA and used browser support for service workers ¹⁸ to enable limited offline functionality.

The service worker specification is managed by the Service Worker Working Group as part of the World Wide Web Consortium (W3C) and has wide browser support.

Not all functionality of QDAcity is supported in an offline state. For instance, new users

¹⁸https://www.w3.org/TR/service-workers/
can not be registered, projects can not be created, and metrics (such as inter-rater agreement) can not be calculated. Instead, we only focused on basic operations of the CodingEditor component, like changing the code system, applying codings to text segments, and remove codings from text segments.

A service worker is a special form of a web worker and hence runs in its own thread not blocking the main thread or other web workers.

The service worker acts as a proxy to the GAE backend service. All HTTPS requests are routed through the service worker, which relays the request to the backend service, assuming network availability. The response of the GAE backend service is, in return, also sent through the service worker. In the simplest case, all requests and responses are forwarded unaltered, as well as any error messages such as bad requests, or time-outs. However, the service worker can also cache objects and generate a response to the request itself when the network is unavailable.

In the case where the service worker generates a response locally, the service worker can keep track of the request it has handled through the Cache API which is part of the service worker specification and implemented as a key-value store in the different browsers. This is necessary in many cases since, for the frontend client, there is no difference between a response from the server and a response from the service worker.

To avoid boilerplate code for the routing of requests, we used a JS library from Google called Workbox\(^{19}\).

Workbox makes it easy to configure simple caching strategies. For example for GET requests under a specific URL, the service worker should always cache the last response which is repeatedly sent for any instances of the same request that follow after a connection loss. However, any updates to project data should at a later point be synchronized with the backend DB, where conflicts may arise from changes that could have been made by collaborators on the same data. The conflict resolution is described in more detail in section 5.3.1.

\(^{19}\text{https://developers.google.com/web/tools/workbox}\)
Some user interactions in the CodingEditor are not handled through HTTPS requests but through a websocket connection to the RTCS. These can not be handled by default using the Workbox library and our `SyncHandler` component because a service worker can not establish a websocket connection. Hence, the frontend does need to be aware of its current connectivity status. Communication between the main thread and the service worker is possible through the `postMessage` interface implemented by the browser. If the frontend is in offline mode, messages that would normally be sent to the RTCS are then sent to the service worker. The response is respectively expected as a message from the service worker in the same form it would come from the RTCS. Here, it was of particular benefit to using JS as the programming language in the RTCS, allowing for code re-use.

While in offline mode, the UI informs the user of the connectivity status, and the user is shown indications of which data was changed locally only and needs to be synced at some later time.

When the frontend client re-establishes a connection to both the GAE backend and the RTCS, it will switch to online mode and send a message to the service worker triggering the synchronization process. The service worker will then work through the stack of objects flagged as out of sync while offline. The service worker will first compare the last version before going offline with the current state in the backend. If these are not identical, the service worker will send a message to the frontend client signaling a conflict between the current state in the backend (theirs) and the most recent offline version (ours). The frontend presents these options to the user, who can choose between ours, theirs, the original. The user can also manually edit the data to resolve the conflict.

Resolved conflicts for actions that are normally handled by the RTCS are immediately synced with all currently connects users.

The implementation of the service worker was supported through a Master’s thesis by Knauer [110].
5.3.1 Conflict Resolution

When two or more users collaboratively work in parallel on a project while connected to our RTCS, the consistency across all users is ensured through a binary semaphore that is requested for updates on a specific resource by the RTCS. When one of the collaborators, however, uses QDAcity in offline mode. This synchronization technique no longer works, and conflicts may arise when the collaborator working offline synchronizes his work with all collaborators when the network connection is re-established.

The resolution of conflicts was supported through a Master’s thesis by Isayev [98]

Components

The main components involved in the synchronization process are shown in figure 5.5. The communication between the subsystem UI and the service worker is realized through the postMessage interface implemented by the browser. Communication between the SWCommunicator and the RTCS is facilitated through messages on a websocket connection, while the SyncHandler and GAE Backend Service communicate through HTTPS requests.

The SyncHandler is responsible for keeping track of operations performed locally in an IndexedDB\(^{20}\). When re-establishing a connection, it will process the list of locally modified objects, compares them to the current version in the backend, and tries to merge automatically, if possible. If any conflicts are detected, the SyncHandler delegates their resolution to the user by means of the UI components for conflict resolution. During the synchronization process, which is controlled by the service worker components, the SWCommunicator in the UI subsystem is also triggered by the SyncHandler though the message interface. The SWCommunicator will then update operations performed by the RTCS which the SyncHandler can not handle itself, since it can not establish a connection to the RTCS. The RTCS will then request a lock on the resource, update the GAE backend service and broadcasts the change to all connected users as well as an acknowledgment for a successful update to the re-

questing client. On any error, no change is applied, and the requesting client may retry their request.

**Three-Way Merge**

When the same object (text or code) has been updated in parallel in different ways, we perform a three-way merge to identify the changes that can be merged automatically and those where user interaction is required. This decision is purely based on the three versions of the artifact, not considering the history of change operations as tracked in a change log. A scenario where a conflict arises due to the same data having been modified by two different users in parallel is presented in figure 5.6.

For all paragraphs, we compare the three versions (ours, theirs, original). There are then five cases to consider:

1. All three versions of the paragraph are identical. The paragraph in the merged version will be unchanged.
2. *Original* and *theirs* are the identical, *ours* is different. The paragraph in the merged version will be equal to our version.

3. *Original* and *ours* are identical, *theirs* is different. The paragraph in the merged version will be equal to their version.

4. *Ours* and *theirs* are identical, *original* is different. The paragraph in the merged version will be equal to both our and their version.

5. All three versions of the paragraph are different. The edit distance is calculated (see next section), and a manual conflict resolution is triggered.

The same also applies to different attributes of a code, say the color, or the name.

Each paragraph of text in a document within a QDAcity project has a unique identifier. New or removed paragraphs can therefore easily be detected.

**Edit Distance**

To make it easier for the user to identify what has changed in the conflicting versions compared to the original we identify changes on a more fine granular level. We want to visually highlight specific changes to words or characters in a string. To identify those changes we
need to calculate the minimum edit distance [162]. The minimum edit distance between two strings is defined as the minimum number of operations needed to change one string to the other [141]. The possible operations we consider here are insertion, deletion, and substitutions of text.

We operationalized the minimum edit distance using the Levenshtein distance [118], based on the algorithm described by Wagner and Fischer [206].

We apply the algorithm on the granularity of words, rather than characters. For this, we first tokenize the paragraph, when both versions are not equal.

Consider a text $A$ with length $|A|$ and a text $B$ with length $|B|$, then the distance between the first $i$ tokens of text $A$ and the first $j$ tokens of text $B$ is defined in equation 5.1. The Levenshtein distance between the two full texts is $\text{lev}_{A,B}(|A|, |B|)$.

$$\text{lev}_{A,B}(i,j) = \begin{cases} 
\max(i,j) & \text{if } \min(i,j) = 0, \\
\text{lev}_{A,B}(i-1,j) + 1 & \text{if } \min(i,j) \neq 0 \\
\text{lev}_{A,B}(i,j-1) + 1 & \\
\text{lev}_{A,B}(i-1,j-1) + 1_{(A_i \neq B_j)} & \end{cases}$$

(5.1)

Where $1_{(A_i \neq B_j)}$ is the indicator function for $A_i \neq B_j$ as defined in equation 5.2.

$$1_{(A_i \neq B_j)} = \begin{cases} 
1 & \text{if } A_i \neq B_j \\
0 & \text{otherwise} \\
\end{cases}$$

(5.2)

We start by populating a matrix with the dimensions $(|A|+1)\times(|B|+1)$ with the case where $\min(i,j) = 0$. The result for two example strings $A$=“some random text” and $B$=“some more random text” is shown in table 5.1.

We then go through the matrix row by row, calculating each cell with the formula in equation 5.1 based on the value in the cells to the left, top-left, and top.

For instance, since both $A_1$ and $B_1$ have the same value (the string “some”), the indicator
function $1(A_i \neq B_j)$ equals zero. This makes the last of the three cases for $\min(i, j) \neq 0$ minimal and thus $lev(1, 1) = lev(0, 0) = 0$.

The fully populated matrix of the example is shown in table 5.2.

The arrows in table 5.2 indicate on which of the cases in equation 5.1 each cell’s value is based on. The cell on the bottom-right of the matrix shows the total number of operations needed to transform $A$ into $B$. In our example, the number of operations needed is one.

The type of operation(s) needed to make the strings equivalent can be determined by following the links from the bottom-right cell along the arrows. When a cell value is higher than its predecessor, then the type of operation can be determined by the row, column, and the direction of the predecessor. The three possible types of operations are determined as follows.

- If the predecessor is on top, then the word $B_j$ was inserted.
• If the predecessor is to the top-left, then $A_i$ was substituted with $B_j$.
• If the predecessor is to the left, then $A_i$ was deleted.

5.4 Agreement and Saturation Metrics

We have implemented three different methods for measuring inter-rater agreement (also known as inter-coder agreement and inter-rater reliability), which are presented in section 5.4.1. We further developed a highly configurable way of measuring saturation in a QDAcity project based on change logs of user interactions with the data as described in section 5.4.2.

The implementation of the metrics in our tool was supported through a Master’s thesis by Schöpe [178).

5.4.1 Inter-Rater Agreement

Inter-rater agreement is a measure indicating how similar two independent coders (raters) analyze the same data. Having the same data analyzed by multiple coders and comparing the result of the analysis is an established practice in qualitative research and provides evidence for investigator triangulation. We expect the implementation in QDAcity to serve both the research use case and the domain modeling use case equally.

The simplest form of inter-rater agreement for corpus annotation is observed agreement as the quotient of the number agreed ratings and the sum of all items to be rated [8].

However, this measure can not be compared across different contexts, because the amount of agreement that is to be expected by chance can differ widely, and this is not considered by simple observed agreement. Consider a classification of three classifiers with discrete uniform distribution, then the agreement of two coders applying these randomly is expected to be 33.3%. However, the same random coders would score 50% agreement, if the classification system was reduced to two categories. In the context of QDA, this means, that the number of codes in the code system has a large impact on the expected agreement, due to agreement by chance.

93
Further, if one classifier occurs significantly more frequent than the other, the expected agreement can be significantly higher [59].

Common measures for inter-rater agreement of corpus annotation that do consider the agreement by chance include Cohen’s kappa coefficient [36], Scott’s pi [179], and Krippendorff’s alpha coefficient [113].

Cohen’s kappa and Scott’s pi are related, as both of them are computed using the following general formula.

$$\pi, \kappa = \frac{P_o - P_e}{1 - P_e}$$

(5.3)

Where $P_o$ is the observed agreement and $P_e$ is the expected agreement by chance.

The difference between $\pi$ and $\kappa$ lies in the calculation of $P_e$. While $\pi$ uses squared arithmetic means of the proportions of each category compared to all ratings, $\kappa$ instead uses squared geometric means. Both Cohen’s $\kappa$ and Scott’s $\pi$ are restricted for measurements of agreement on binary or nominal data between no more than two raters. Krippendorff’s $\alpha$ also supports other types of ratings such as ordinal, polar, or ratio measurements.

**Fleiss’ Kappa**

An extension of Scott’s $\pi$ is Fleiss’ $\kappa$ [64], which allows for more than two raters. We chose to implement this more general measure since, for the use case of QDAcity, the calculation of agreement between more than two coders is a common scenario. The general form of Fleiss’ kappa is identical to equation 5.3.

To compute the expected agreement by chance $P_e$, we first need to calculate $p_j$ for each code $j$ as follows.

$$p_j = \frac{1}{Nn} \sum_{i=1}^{N} n_{ij}$$

(5.4)

Where $N$ is the total number of items to rate. In our case this is, depending on the unit of coding considered for agreement, the number of paragraphs, or the number of sentences.
The number of coders is denoted as $n$, and $n_{ij}$ denotes the number of coders who applied code $j$ to item $i$.

The expected agreement by chance is then defined as in equation 5.5

$$P_e = \sum_{j=1}^{k} p_j^2$$  \hspace{1cm} (5.5)

Where $k$ is the number of codes in the code system.

$P_o$ is defined as the arithmetic mean of all $P_i$’s (equation 5.7), where $P_i$ denotes the amount of agreement between all coders on a specific unit of coding $i$, as shown in equation 5.6.

$$P_i = \frac{1}{n(n-1)} \sum_{j=1}^{k} n_{ij}(n_{ij} - 1)$$  \hspace{1cm} (5.6)

$$P_o = \frac{1}{N} \sum_{i=1}^{N} P_i$$  \hspace{1cm} (5.7)

Fleiss’ $\kappa$ only calculates agreement on a categorical rating, meaning each unit of coding can only be assigned one category (code). However, in QDAcity-RE it is common that many codes are applied to the same text segment. We, therefore, process each code separately. For each unit of coding we consider the rating of all raters on the categories code applied or code not applied.

Krippendorff’s Alpha

Another common way of measuring inter-rater reliability, which accounts for agreement by chance (or in this case disagreement), is Krippendorff’s alpha coefficient. Krippendorff’s alpha is a generalization of multiple other agreement metrics, allowing for more types of rating (binary, nominal, ordinal, interval, etc.) between two or more raters, and allowing for unequal sampling sizes [112].

The general form of $\alpha$ is presented in equation 5.8, with $D_e$ denoting the expected disagreement between raters and $D_o$ the observed disagreement between raters.
\[ \alpha = 1 - \frac{D_o}{D_e} \] (5.8)

The formulas for observed and expected disagreement are shown in equation 5.9 and equation 5.10, respectively.

\[ D_o = \frac{1}{n} \sum_{v=1,v'=1}^{V} \theta_{vv'} \text{metric} \delta_{vv'}^2 \] (5.9)

\[ D_e = \frac{1}{n(n-1)} \sum_{v=1,v'=1}^{V} n_v n_{v'} \text{metric} \delta_{vv'}^2 \] (5.10)

The difference function \( \text{metric} \delta_{vv'} \) used in both formulas is dependent on the type of data that is being rated. The type of rating performed within the scope of this thesis using QDAcity-RE is the application of a set of codes in a code system to segments of text. Hence, we use the difference function for nominal data as shown in equation 5.11,

\[ \delta_{\text{nominal}}(v, v') = \begin{cases} 
0 & \text{if } v = v' \\
1 & \text{if } v \neq v'
\end{cases} \] (5.11)

To measure inter-rater agreement using \( \alpha \) requires the computation of a reliability data matrix and a coincidence matrix. The form of the reliability data matrix is presented in table 5.3, where \( v_{ij} \) denotes the value set by rater \( i \) for unit of coding \( j \). Note, that because in our use case ratings are not categorical and multiple codes can be assigned to the same unit of coding, the reliability matrix is computed for each code, with the possible values code set and code not set.

In the following, we use the notation of \( R \) for the number of raters, and \( N \) the number of possible units of coding (sentences or paragraphs).

From the reliability matrix, we then derive the coincidence matrices for both the observed and expected coincidences. The coincidence matrix contains frequencies of matches between
the raters. The matrix is always square with all possible values in both dimensions of the matrix. The coincidence matrix for observed coincidences is presented in table 5.4.

The elements of the matrix $o_{ij}$ are calculated as in equation 5.12.

$$o_{v'v} = \sum_{u=1}^{N} \frac{r_u}{m_u - 1} = o_{v'v}$$  \hspace{1cm} (5.12)

Where $r_u$ is the number of possible combinations of two raters where one rater applied code $v$ and the other code $v'$ to a unit of coding $u$. Note, that for $r_u$ all possible combinations, regardless of the order are counted, so both $vv'$-pairs, and $v'v$-pairs count as separate pairs. $m_u$ denotes the number of raters who rated the unit of coding $u$. Through this mechanism, Krippendorff’s alpha handles the case when not all units of coding are coded by all raters.

The categorical rating required for the computation of $\alpha$ leads to the same approach we took for $\kappa$, calculating the agreement for each code separately, and aggregated a general agreement value from those.
Recall, Precision and F-Measure

The information retrieval measures recall and precision may also be used as an inter-rater agreement metric [92].

Consider two coders, who decided which codes to apply to a specific text segment. For each code $c$ in the code system, there are four possible classifications for the different combinations of coding for this segment.

- True positive (TP): Both coders applied $c$
- True negative (TN): Both coders did not apply $c$
- False positive (FP): Coder A did not apply $c$ but coder B did do so
- False negative (FN): Coder A did apply $c$ but coder B did not

Of all the items in the units of coding under consideration, the instances where coder A applied the code $c$ is considered the set of relevant items ($RI$), and the items, and the instances where coder B applied the code are the selected items ($SI$).

The four cases are illustrated in figure 5.7.

![Figure 5.7: Selected and Relevant Items](image)
Recall is then defined as the number of items where both coders agreed to apply the code divided by the number of times coder A applied the code, as shown in equation 5.13.

\[ Recall = \frac{TP}{RI} = \frac{TP}{TP + FN} \]  

(5.13)

Precision is defined as the number of items where both coders agreed to apply the code divided by the number of times coder B applied the code, as shown in equation 5.14.

\[ Precision = \frac{TP}{SI} = \frac{TP}{TP + FP} \]  

(5.14)

To an extent, recall and precision are inversely correlated. Imagine coder B wants to maximize the recall score while he is coding the data. A sure way to achieve this goal would be to indiscriminately apply the code to all units of coding. This however likely degrades the precision score by maximizing the number of selected items, and thus the denominator of equation 5.14. Conversely, if coder B applies the code especially carefully only in instances where the confidence is extremely high, then precision may be high due to a low false positive rate, but some of the instances that coder A did code may be missed, hence increasing the false negative rate.

That is why it makes sense to not view each of these measures isolated, but always in relation to each other. The F-Measure, also known as the F1-Score, is defined as the harmonic mean between the two values, as shown in equation 5.15.

\[ F_1 = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision} \]  

(5.15)

**Implementation Details**

In QDAcity, a user can request a copy of a project revision that contains all documents and code system data of the revision except for the coding data. The owner of the original project can then, at any time, trigger an agreement report which calculates inter-rater agreement between the original state of the revision and all re-coded copies of the revision.
The agreement is then computed as described above, depending on the selected agreement metric. The results are then presented per document that has been re-coded and as an aggregate.

The computation is deferred to a task queue in the GCP project. The computation is highly parallelized with one task being created for each combination of rater, document, and code. The tasks are consumed from the queue at a defined rate by all instances of the GAE backend service.

5.4.2 Saturation

While saturation is a concept widely used as a stopping criterion in qualitative research, the operationalization of it is mostly unclear. In fact, we have had a reviewer of one of our papers in peer-review criticize our operationalization of saturation because, for them, saturation would be “more of a feeling”, than a concrete measurement. However, we argue that any explicit operationalization makes a stronger case for the reliability of the analysis than a vague feeling of being done. We concur with Celli et al. that “evidence of saturation must be given in the presentation of the data and discussed via the forms in which it was recognized during the analysis” [24].

Saturation Interval

Saturation is measured changes over fixed time intervals (iterations) $T_0$ to $T_n$. At the end of each saturation interval, the current state of the project is evaluated based on its history of evolving over the span of multiple intervals. The granularity of the time intervals $T_n$ can vary.

We have considered two types of triggers which define the start of a new saturation interval $T_{n+1}$. Both were implemented in the backend, however, only one was integrated into the current release.

The user action that marks the end of the current interval and the calculation of a new saturation value is the creation of a new project revision on the project dashboard. A project
revision is a labeled version of a state of the project defined by a set of documents, a code system, and the codings applied to sections in the document. A typical scope of one revision is the addition of new material or the conclusion of a defined subset of the analysis. Users can create new revisions at any time. Therefore, the measurement of saturation as presented here is only meaningful if contextualized with the magnitude of the incremental change from one revision to the next.

A second method of defining the length of $T_n$, which was implemented in the backend, is based on the number of codings applied since the beginning of the interval. For instance, we could define that after every 20 codes that are applied to a text segment, we would measure saturation based on the changes that occurred since the start of the interval, as described above. Within our own projects, the number of codings was approximately linearly correlated with the number of revisions created, so it might be a good substitute for measuring “work performed” for the definition of the length of the saturation interval. However, we expect that finding the right number of codings performed in each interval is equally context-dependent as finding the right granularity for creating new project revisions. Due to time constraints, we chose not to integrate this way of defining the saturation interval.

Saturation implementation

Saturation in QDAcity is based on change logs which are maintained if certain changes to a project’s data are triggered through user interaction. For each change log entry, we log the object type of the change and the type of change performed on the object. The object type can be any of the following: DOCUMENT, CODE, CODEBOOK_ENTRY and CODE_RELATIONSHIP. Subtypes exist for the elements of the codebook entries and the attributes of Codes (Color, Name, etc.). The types of change we log are CREATED, DELETED, APPLIED, and MODIFIED. However, not all possible combinations occur in our implementation. A codebook entry, for instance, can not be created or deleted on its own. Instead, it only exists as an attribute of a code, which can be created or deleted. Still, we wanted to
track changes to the code system independently from changes to other code properties, like its color or name. Also, the object type CODE is the only one which can have the change type APPLIED associated with it, meaning a text segment has been coded with this code. We denote the set of valid combinations of object type and change type as the set of change categories $\mathbb{K}$.

Saturation is calculated for a specific change category $c$ as in equation 5.16.

$$S_{T_n}^c = 1 - A_{T_n}^c$$

(5.16)

Where $A_{T_n}^c$ is the number of changes of change category $c \in \mathbb{K}$ within the interval $T_n$ in relation to the total number of changes of category $c$ over all intervals $[0..n]$, as shown in equation 5.17. $C_{T_n}^c$ denotes the number of changes in time interval $T_n$.

$$A_{T_n}^c = \frac{C_{T_n}^c}{\sum_{i=0}^n C_{T_i}^c}$$

(5.17)

The weighted average of saturation of all changes is defined as in equation 5.18, where $w_c$ is the weight for change category $c \in \mathbb{K}$.

$$S_{T_n}^* = \sum_{c \in \mathbb{K}} S_{T_n}^c \frac{w_c}{\sum_{d \in \mathbb{K}} w_d}$$

(5.18)

Although QDAcity provides a default setting, the operationalization of saturation is highly dependent on the context. For instance, whether changing the color of a code in of itself is a meaningful change within the analysis or not depends on the use of color during the analysis. If the code color is used to visualize the code being a subcode to a specific category code, then a change in color can be considered merely a proxy for the relocation of the code in the hierarchy, which is covered by its own change type category. Or if color is used as a measure of the maturity of a code, this is a type of meta-information one might not want to consider in the saturation metric. In QDAcity, the weights can therefore be configured by the user for each project individually.
The length of the time intervals $T_n$ is configurable as a number of project revisions. The QDAcity default for the length of $T_n$ is three project revisions. A project owner can also determine a minimum number of iterations (project revisions) necessary before saturation is calculated. Before the minimum number of iterations is reached, the saturation is considered to be 0%. The default configuration is that at least 3 project revisions must exist.

Further, a threshold can be defined, delimiting the number of changes acceptable for a change category $c$ within the last interval $T_n$. With this threshold, it can, for instance, be defined that 10% of the total changes to code memos may happen within the interval $T_n$, and saturation $S^c$ could be considered to be reached for the change category of code memo modifications.

The adjusted saturation for change category $c$ and interval $T_n$ and a configured minimum portion $m_c$ of changes to have happened before $T_n$ then becomes:

$$
\bar{S}_{T_n} = \min(1, \frac{S_{T_n}}{m_c})
$$

Consequently, the weighted adjusted saturation function used in QDAcity is defined as in equation 5.20.

$$
\bar{S}_{T_n}^* = \sum_{c \in K} \min(1, \frac{S_{T_n}}{m_c}) \ast \left( \frac{w_c}{\sum_{d \in K} w_d} \right)
$$

Saturation is computed asynchronously in a deferred task pushed to the task queue of the GCP project. The instances servicing these tasks are the same that service user-facing requests and scale with the combined workload. The benefit of this is increased responsiveness when creating a new revision, with the drawback that the update of saturation happens eventually sometime after the revision is shown as created to the user.
5.5 CD Pipeline

For qdacity.com we have set up a continuous deployment (CD) pipeline using GitLab CI\(^{21}\). This pipeline is presented in figure 5.8.

![Figure 5.8: The QDAcity.com CD Pipeline](chart)

Jobs within one stage are executed in parallel and independent from each other, while artifacts may be passed on from one stage to jobs in a later stage.

**Build Phase**

The build artifacts from the build phase are the WAR folder build with Maven\(^{22}\) from the job Backend Build, and the JavaScript bundle built with Webpack from the job Frontend Build. The build stage also installs all necessary dependencies from NPM\(^{23}\) and caches them for future jobs. For the job RTCS Build, the installation of all node dependencies is all that is necessary in the build phase.

**Test Phase**

The jobs in the test phase consume the build artifacts produced in the previous stage. The unit test job creates a code coverage reports as an output artifact and the acceptance test job

---

\(^{21}\)https://about.gitlab.com/product/continuous-integration/

\(^{22}\)https://maven.apache.org/

\(^{23}\)https://www.npmjs.com/
creates log files from the Java development server as well as the RTCS development server set to the debug level.

The job *Acceptance Tests* runs a docker container built on a selenium base image, where the development servers have been scripted to start. Then, a suite of selenium tests is being run through the chrome driver testing from a user’s perspective using a virtual frame buffer. After the first test suite ran successfully, the development servers are terminated and a suite of selenium tests is run against the service worker which is expected to have been installed during the first suite of tests. This simulates the user experience while the user is offline. The development servers are then restarted for the third suite of tests, which test the synchronization of resources cached in the browser with the data in the live database, simulating a reconnect of the user, and triggering various scenarios for automatic merges and merge conflicts.

Further, the babel plugin is tested on whether a supported set of ways to import and use the localization library are recognized correctly, whether all uses of the library contain all required parameters, and if all defined IDs of localized strings are present in all translations.

**Deploy Phase**

In the deploy phase, the GitLab runner authenticates itself towards the GCP using service account credentials stored as secret variables and then deploys both the *GAE backend service* which contains the Java backend and the static bundled JavaScript client as well as the Auth Proxy to the GAE standard environment using a maven plugin. The RTCS is deployed using the gcloud CLI, which copies the source files and dependencies to a dedicated compute engine instance and then restarts the service.

**Release Phase**

The optional release Phase of the pipeline is only triggered manually, and renames static assets to include the version number in their path, and makes the names comply with a more aggressive caching policy. Also, the configuration file of the main app is changed, so the ser-
vice never shuts down, even without any load. Because of the aggressive caching policy, a
version that has once been released can no longer be reliably updated, because old versions of
some artifacts may still be cached somewhere in the network infrastructure.
Validation

To validate our hypothesis 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.

Our research design was driven by the following research question.

**Research question:** 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, and how do the traces assist the analysis?

To identify potential limitations of our experiment design and apply corrective measures where possible, we performed multiple peer debriefing [79, 189] sessions 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.
6.1 Experiment Design

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. The other group was tasked to perform an analysis of the same data using a strategy utilizing strategic reading \([50, 51]\). The participant sampling is described in more detail in section 6.1.2.

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.

Since the population of our experiment groups was small (nine participants), we can not 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 considers the following variables:

**Independent variable:**

- Method used for analysis

**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
The experiment was conducted over a four-week period. The participants were not given all interview transcripts at once. Instead, we divided the data up into chunks and each week handed them one or two additional interview transcripts. This way, the iterative nature of our analysis method could be enforced.

The order in which the interview data was handed out to participants was the same in which the data was 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 diary study 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. The study diary approach is described in section 6.3.1.

The timeline of one week of the study is outlined in figure 6.1.

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 [81]. 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 towards the end of the study and also to include questions that were brought up by the entries in the study diary.

6.1.1 Data on Target Domain

The data we used for the evaluation of the analysis phase of the method is the same as in our exploratory study on HR development [107]. 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 baseline 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 interview transcripts ranged in length from five to fifteen pages each.

6.1.2 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 the search for recruits within our courses and through Facebook groups for students of the respective degree program at our university. As an 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 on a five-point scale from 'none at all' to having 'used it in multiple studies’

• Whether they have taken the course on conceptual modeling or UML at our university or equivalent courses at another university

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.

The responses with regard to the self-assessed competency in domain modeling, as well as the responses for the self-assessed competency in QDA, are shown in figure 6.2. The rating was performed on a five-point scale ranging from None in both cases to Used in multiple studies for QDA, and Expert for domain modeling competency. 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 self-assessed competency for domain modeling is 1.55, while the average for QDA is significantly higher at 2. We assume the overall assessment of QDA competence be-
Table 6.1: Demographic Averages by Group

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>QDAcity</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample size</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Age</td>
<td>29.6</td>
<td>28</td>
</tr>
<tr>
<td>Gender</td>
<td>40% m 60% f</td>
<td>75% m 25% f</td>
</tr>
<tr>
<td>Work experience (years)</td>
<td>0.3</td>
<td>0.125</td>
</tr>
<tr>
<td>Competence in modeling</td>
<td>1.8</td>
<td>2.25</td>
</tr>
<tr>
<td>Competence in QDA</td>
<td>1.4</td>
<td>1.75</td>
</tr>
</tbody>
</table>

The learning 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 were 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. The average self-assessed competence for domain modeling in the QDAcity group was 1.8, while the average in the control group was 2.25.

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 are presented in table 6.1. 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 6.1 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.

Both groups were given the same task of creating a conceptual domain model from interview data provided to them. The independent variable between the two groups was the method for analyzing the qualitative data input.

At the beginning of the study, we held kickoff meetings with both groups separately, where each was given a 90-minute introduction to 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.

The two groups are described in the subsequent sections 6.1.3 and 6.1.4.

6.1.3 **Control Group**

The control group was instructed to use an analysis method described by Diaz using a technique called *strategic reading* [50]. This method, similarly 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 [129], and has been suggested for design science research [50, 51]. 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.

• **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 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 root-cause analysis in this context is a mind map, rather than a UML diagram. The original motivation was to guide researchers in 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.

The analysis method used by the control group is also tool-supported. It uses *Mendeley*¹ as a reading platform and *MindMeister*² as a modeling tool for mind maps. Both tools are connected using the Google Chrome plugin called *DScaffolding*³.

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.

Diaz describes the flow of information between Mindmeister and Mendeley as the “Pur-
pose Pipe” and the flow from Mendeley to Mindmeister as the “Annotation Pipe”, as de-

---

¹https://www.mendeley.com
²https://www.mindmeister.com
³https://chrome.google.com/webstore/detail/dscaffolding/hkgmnnjalpmapogadkngkgbbgdljnlne
picted in figure 6.3. 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.

We chose the method for the control group specifically due to it being tool-supported and yielding a similar level of traceability. However, the philosophy of the tool support for the control group was entirely different in the sense that it purposefully only connected existing tools whereas the tool-support for the QDAcity-RE method 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 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.

As an additional step, to arrive at a model in the same notation within both groups, the control group was asked to further translate the findings in their 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.
6.1.4 QDAcity Group

The QDAcity group was tasked with creating a conceptual model using the QDAcity-RE method supported by the QDAcity tool.

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 consider the current state of analysis sufficiently accurate and complete.

During the analysis process analysts annotate the input material such as interviews, workshop transcripts, etc. 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 \[107\].

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.

Each week the participants were required to perform another iteration of coding, refining the code system using the constant comparison method.

At the end of each iteration, the code system was evaluated using the coding paradigm, to identify areas that need further investigation. Even though in the given experimental scenario this information could not be used to drive further sampling, as it would normally do, this still serves as a reflection mechanism for the analyst influencing further reading and analysis of source materials.

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 codebook 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 the most frequently co-located codes with a percentage of the text area of overlap. This helps to find potentially missed relationships between concepts.

6.1.5 Evaluation Methods

We analyzed the results of the experiment in two dimensions: One dimension (a) is 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 [107]. The other dimension (b) is 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 section 6.2 for (a), and section 6.3 for (b).

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

Figure 6.4 presents an overview of our evaluation strategy.
Figure 6.4: Evaluation overview
6.2 Artifact Evaluation

6.2.1 Evaluation Method

Lindland, Sindre, & Solvberg [122] 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 significantly 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 [115], which for two samples is similar to a Wilcoxon test [212] in that it compares the ranks of values for each variable rather than the actual values.

The Kruskal-Wallis test [115] 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 non-parametric test was the small population of our sample. Normal distribution of all relevant features of both our groups can not 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.
6.2.2 Artifact Evaluation 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. The results of this comparison are shown in Table 6.2 for the control group and in Table 6.3 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 the QDAcity-RE method 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.

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.

### Table 6.2: Artifact Comparison with Expert Solution - Control Group

<table>
<thead>
<tr>
<th>Participant</th>
<th>Recall(%)</th>
<th>Precision(%)</th>
<th>F-measure(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average</td>
<td>29.688</td>
<td>57.981</td>
<td>38.160</td>
</tr>
<tr>
<td>A1</td>
<td>41.667</td>
<td>58.824</td>
<td>48.780</td>
</tr>
<tr>
<td>A2</td>
<td>39.583</td>
<td>59.375</td>
<td>47.500</td>
</tr>
<tr>
<td>A3</td>
<td>16.667</td>
<td>47.059</td>
<td>24.615</td>
</tr>
<tr>
<td>A4</td>
<td>20.833</td>
<td>66.667</td>
<td>31.746</td>
</tr>
</tbody>
</table>

### Table 6.3: Artifact Comparison with Expert Solution - QDAcity Group

<table>
<thead>
<tr>
<th>Participant</th>
<th>Recall(%)</th>
<th>Precision(%)</th>
<th>F-measure(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average</td>
<td>48.958</td>
<td>55.258</td>
<td>49.474</td>
</tr>
<tr>
<td>B1</td>
<td>66.667</td>
<td>47.059</td>
<td>55.172</td>
</tr>
<tr>
<td>B2</td>
<td>43.750</td>
<td>32.308</td>
<td>37.168</td>
</tr>
<tr>
<td>B3</td>
<td>43.750</td>
<td>58.333</td>
<td>50.000</td>
</tr>
<tr>
<td>B4</td>
<td>41.667</td>
<td>83.333</td>
<td>55.556</td>
</tr>
<tr>
<td>B5</td>
<td>56.250</td>
<td>52.941</td>
<td>54.545</td>
</tr>
</tbody>
</table>
model for the purpose of avoiding idiosyncrasies of single companies. As an example of this, participants’ 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 is an artifact of a different modeling style, 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 table 6.2 and table 6.3, 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 downwards.

**Statistical Evaluation**

Figure 6.5 shows a boxplot of the precision, recall and f-measure [202] by group, and the descriptive statistics are also presented in tabular form in table 6.4, where the column for f1 stands for f-measure (also called F1-Score).

<table>
<thead>
<tr>
<th>Measure</th>
<th>QDAcity</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rec.</td>
<td>Prec.</td>
</tr>
<tr>
<td>Min.</td>
<td>41.7</td>
<td>32.3</td>
</tr>
<tr>
<td>Max.</td>
<td>66.7</td>
<td>83.3</td>
</tr>
<tr>
<td>Mean</td>
<td>50.4</td>
<td>54.8</td>
</tr>
<tr>
<td>Std.Dev</td>
<td>10.8</td>
<td>18.7</td>
</tr>
</tbody>
</table>

The null- and research-hypotheses of our Kruskal-Wallis test is as follows:
Figure 6.5: Boxplot of Recall, Precision and F-Measure by Group
$H_0$: The probabilities distributions in both groups are the same.

$H_1$: 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 $\alpha$.

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

The H-value is computed as follows:

$$H = \frac{12}{n(n+1)} \sum_i \frac{R_i^2}{n_i} - 3(n+1) \quad (6.1)$$

with $R_i$ being the sum of all ranks within group $i$, $n_i$ 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 6.5.

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 6.5: Ranking for H-Test.

<table>
<thead>
<tr>
<th>Measure</th>
<th>Sum of Ranks</th>
<th>Mean Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>QDACITY</td>
<td>CONTROL</td>
</tr>
<tr>
<td>Recall</td>
<td>34.5</td>
<td>10.5</td>
</tr>
<tr>
<td>Precision</td>
<td>21.5</td>
<td>23.5</td>
</tr>
<tr>
<td>F-Measure</td>
<td>33</td>
<td>12</td>
</tr>
</tbody>
</table>

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 6.6.

The $\chi^2$ approximation for the critical value of H for a significance level of 0.05 and one
Table 6.6: Kruskal-Wallis H-Test results

<table>
<thead>
<tr>
<th>Measurement</th>
<th>H-Value</th>
<th>p-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall</td>
<td>5.507</td>
<td>0.0189</td>
</tr>
<tr>
<td>Precision</td>
<td>0.741</td>
<td>0.3893</td>
</tr>
<tr>
<td>F-Measure</td>
<td>3.84</td>
<td>0.05</td>
</tr>
</tbody>
</table>

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 6.6.

With an $\alpha$ 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 in 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.

6.3 Process Evaluation

6.3.1 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.

Diary Study

Diary studies have been used in computer science and information systems research to study a variety of phenomena such as the contribution of particular demographics to open source [47], or the activities of novice professional software developers [14].

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 [14], or interviews [2], adding method- and data-triangulation [81].

The study diary was conducted using an online form containing 11 questions, 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 the 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 questions are presented in appendix A.2.
The ratings within the diary study were all on a seven-point 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 section 6.3.2.

Interviews

We performed 38 in-person and two online interviews. 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 [81] 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 primary coder. 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 one additional coder coded all codes under the category process→analysis process, the other two additional coders 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 was 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. After resolving these issues in a discussion, all coders agreed they had a common understanding of the data.

A third round of inter-rater reliability was added after three quarters of the interviews had been coded, covering data from the first three and a half weeks of the study. 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 figure 6.6. 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 section 6.3.2.
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 different types of user interactions with the data.

We operationalized saturation of our analysis based on suggestions of Francis et al. [67]. 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 was documented automatically by QDAcity, which we used for our analysis.

The results of the saturation measurement are presented in section 6.3.2.

6.3.2 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 was 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 marked with a \([B]\) were made by a participant of the QDAcity group.

We present the main results of the study diary and the results from the interview analysis within the following two subsections.
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 figure 6.7, and the self-assessment of the confidence in the resulting artifacts is shown in figure 6.8.

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 weeks two and four, where the qualitative data we gathered offers some explanation. In week two, the average rating in the con-
trol group went up, while that in the QDAcity group went down. Our 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 to create 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 off 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 figure 6.9. 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 figure 6.10.

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 the 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 re-use of colors, information that may be visible at 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.
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, wasn’t 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 a 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 through a single tool. This allowed for more frequent switching back and forth between assigning codes and modeling concepts and their relationships. 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 (figure 6.10).
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 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 the 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 towards 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 an interesting insight into 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 improving slightly during the study, the qualitative feedback in the interviews suggests a significant change in attitude towards 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 wasn’t 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 towards 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]

**Saturation**

The saturation based on all changes, as well as the three categories of changes are presented in figure 6.11. 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.

### 6.4 Discussion

Strategic reading and the QDAcity-RE method both draw on the idea that besides creating a documentation artifact, the process of annotation also improves the comprehension of the analyzed text [144, 147, 148]. The integration of documentation as a side product of the analysis process worked in both cases.

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 intended 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.

6.5 Limitations

Within this study, we only evaluated the analysis phase of the QDAcity-RE method. Iterative data gathering and analysis is an important cornerstone of the QDAcity-RE method and should be further evaluated in detail. We attribute the successful application of the QDAcity-RE method in our exploratory studies 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 enforced an iterative aspect into our experiment by not giving participants access to all source data right away.

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 a stronger negative impact on the 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 type of data was 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 effect 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 commonly done, has been criticized for making the transferability of the results to professionals inappropriate, and thus lowering the external validity of the research [54, 185]. 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 [199].

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. 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.
Conclusion

We present and evaluate a novel approach to domain analysis, called QDAcity-RE, by adapting qualitative research methods.

In our approach, the research process of *theory building* facilitates domain analysis within the requirements elicitation phase of a software development project.

We show how an iterative process of concurrent data collection and analysis can be applied to requirements engineering, including open, axial, and selective coding of qualitative data.

Our method inherently produces traceability of requirements back to original statements by stakeholders (pre-RS traceability), which does not have to be created and maintained separately after the fact. The traces are documented in an intermediate analysis artifact called the *code system* which evolves iteratively with the analysis process. The code system acts as a universal model from which a variety of artifacts, describing both behavioral and structural aspects, can be derived. This thesis focuses on the creation of conceptual domain models using QDAcity-RE, but peripherally also demonstrates this attribute through the generation
of behavioral models and a software requirements specification. The visualization of parts of the code system as a UML class diagram is facilitated through a code system language (CSL).

We show that by applying QDA to domain analysis, structural elements and relationships needed to derive a UML class diagram can be extracted from a code system based on interviews with domain experts. Constant comparison and theoretical sampling assist in integrating differing domain descriptions into an abstract model.

While the analysis process still requires interpretations and modeling decisions, our method provides more guidance than existing domain analysis approaches and a thorough documentation of these decisions. In addition, codes and memos ensure traceability between the original data and the derived model and assist in connecting several RE artifacts ensuring a high degree of inter-model consistency.

QDAcity-RE was iteratively created and during its development continuously evaluated in four exploratory case studies in the domains of medical imaging diagnostics, railway systems, HR development, and qualitative research.

Our method was also successfully applied by our colleagues Harutyunyan et al. to elicit requirements for FLOSS governance tools [85].

We further validated the analysis method of QDAcity-RE in an experimental setup, comparing it to a control group using a method for conceptual modeling based on strategic reading with a similar level of traceability [50, 51]. The control group method, similarly to QDAcity-RE has roots in analysis for research purposes.

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, needing only 56.79% of the time the average participant in the control group needed.
We explain part of this discrepancy through the more tight 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. Participants were developing strategies for switching less often 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 than in the control group. The difference for recall and f-measure were evaluated as being statistically significant with an \( \alpha \) of 0.05. The difference for precision was evaluated as not statistically significant.

The models created with QDAcity-RE and the QDAcity tool 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.

The number of nine participants in our experiment was limited, making generalization to a larger population difficult. But we gathered in-depth data on the experience of the participants which revealed significantly differing themes in the 40 interviews we analyzed as well as in the study diary the participants kept. Also, the characteristic difference of the models in the QDAcity group being more detailed throughout the experiment was 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, was 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 long running or require a lot of collaboration in which more detailed and traceable documentation is particularly helpful.
We were guided by two separate frameworks of rigor, depending on the research method chosen. These frameworks guided our research design and execution and were also used for its evaluation in terms of research rigor.

We evaluated the qualitative theory building part of the research, the four exploratory cases, within the context of the naturalistic research paradigm. Specifically, we used the framework of trustworthiness as defined by Guba and Lincoln [79, 120, 121]. The framework with its four categories credibility, transferability, dependability, and confirmability, is presented in section 7.1.1. The evaluation of our qualitative theory building work based on this paradigm is presented in section 7.1.3.

Experimental studies are the epitome of the rationalistic research paradigm. Consequently, we evaluate our experiment in terms of statistical conclusion validity, internal validity, construct validity, and external validity [180]. The four categories of validity, as well as the criteria of objectivity, and reliability, are explained in section 7.1.2. An evaluation of our experiments based on the rationalistic paradigm is presented in section 7.1.4.

Since within the experiments we evaluated both quantitative and qualitative data, we also evaluated our validation work through the lens of traditionally qualitative research. We used Guba and Lincoln’s trustworthiness framework for this evaluation [79, 120, 121], as we did for the exploratory studies. The evaluation of our experiments based on the naturalistic paradigm is presented in section 7.1.5.

Some of the practices for rigorous research bridge the two paradigms, like the recommendation for “reconciling qualitative and quantitative data” in mixed-methods research as part of a triangulation effort [150]. With two parts of this thesis in theory building and theory testing, the thesis as a whole follows this practice, and the mixed-methods design of our experiment also allowed us to explain quantitatively identified phenomena more closely using the qualitative data of 40 semi-structured interviews with our participants.
Table 7.1: Evaluation Criteria by Research Paradigm. Adapted From [79]

<table>
<thead>
<tr>
<th>Objective</th>
<th>Naturalistic paradigm</th>
<th>Rationalistic paradigm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Truth value</td>
<td>Credibility</td>
<td>Internal validity</td>
</tr>
<tr>
<td>Generalizability</td>
<td>Transferability</td>
<td>External validity</td>
</tr>
<tr>
<td>Consistency</td>
<td>Dependability</td>
<td>Reliability</td>
</tr>
<tr>
<td>Neutrality</td>
<td>Confirmability</td>
<td>Objectivity</td>
</tr>
</tbody>
</table>

Table 7.1 provides a comparative overview of terminology from both the naturalistic and rationalistic paradigm, with regard to their evaluation framework. Any study, qualitative, quantitative, or mixed methods should address these four objectives. However, depending on the paradigm, the measures of assessment differ, as well as the methods recommended for achieving the objectives.

Internal validity and credibility are both measures of the truth value and depend largely on choosing a study design which ensures that the results plausibly follow from the implementation of the study.

Transferability and external validity are both concerned with the generalizability of the findings. Qualitative studies usually have a strong focus on theoretical generalizability, which is a measure of how well findings are transferable to different contexts, rather than statistical generalization to a larger population [57, 127].

To achieve the objective of consistency, the research must be based on stable measurements with the appropriate instruments, which make the study replicable. Both dependability and reliability fall in this category.

Confirmability and objectivity both are criteria that require neutrality, meaning that researcher bias should be avoided.

Two criteria within the rationalistic paradigm which are not in table 7.1, are statistical conclusion validity and construct validity. While the objective of statistical conclusion validity may be categorized as truth value, similar to internal validity, it also requires reliable measurements (objective: consistency). Construct validity is a measure of generalizability with regard
to the higher order constructs. However, with its concern for the appropriate operationalization of the constructs, it also influences reliable measurement (objective: consistency).

Figure 7.1 visualizes the criteria used in our evaluation of research rigor in the two parts of this dissertation, the qualitative studies used for theory building and the controlled experiment used for theory validation.
Credibility

Credibility refers to the truth value of the findings [79]. Hence, it is closely related to the rationalistic concept of internal validity. However, while internal validity determines truth of the findings as the strength of the inference of a causal relationship between two constructs, and often within an abstracted context, such as a laboratory experiment, the trustworthiness criterion of credibility requires researchers to employ tactics of ensuring confidence in the plausibility of the results while maintaining a more holistic stance.

Practices recommended by Guba to ensure credibility include prolonged engagement, persistent observation, peer debriefing, triangulation, member checks, establishing structural corroboration or coherence, and establishing referential adequacy [79].

Prolonged engagement means spending enough time with the studied phenomenon to reflect on potential biases and allow participants to adjust to the situation of being studied [79].

Persistent observation requires “extended interaction with a situation” in order to identify typical and atypical responses and assess the relevance of the data gathered to the research [79].

Peer debriefing requires the researcher to present the research design and results to his peers to be scrutinized and questioned. This ensures that the methods are adequate for the research question and allow for the research design to be improved based on constructive criticism.

Triangulation is a means of corroborating findings through the use of different data, methods, and investigators, to ensure that the results of the analysis are not rooted in a systemic bias of one of these particular dimensions [81, 48].

Member checks require the researcher to mirror the research results back to participants of the study to ensure that the results reflect their understanding. This can be performed as an evaluative measure at the end, but also as a measure to steer the course of investigation when performed with intermediate or preliminary results.
Establishing structural corroboration or coherence means that all interpretations of the analysis need to be checked against all data gathered to identify conflicting perspectives [79].

Establishing referential adequacy involves the gathering of additional data after the fieldwork has concluded to compare it with the study results [79].

Transferability

Transferability describes the generalizability of the research findings to different contexts [79].

*Purposive sampling* is recommended as the most impactful strategy to ensure transferability by selecting a sample that includes different settings in various relevant dimensions.

Dependability

*Dependability* “is concerned with the stability of data” [79]. The analysis results should be consistent and repeatable. To ensure consistency in the data Guba recommends *method triangulation, investigator triangulation, peer debriefing* and the establishing of an *audit trail* [79].

Confirmability

*Confirmability* is the degree to which influence due to researcher bias was avoided, and the results are truly grounded in the data gathered from the participants. The soundness of the interpretation that is part of the analysis needs to be checked.

Methods of addressing confirmability include *investigator triangulation*, one of the four types of triangulation as it is also recommended for *credibility*.

Guba further recommends to *practice reflexivity*, which requires the researcher to “systematically and rigorously reveal their methodology and themselves as the instrument of data generation” [167]. This practice should be directed at the participants as much as to oneself.
7.1.2 Rationalistic Paradigm

Validity in general, within the rationalistic paradigm, is the degree of truth in an inference [180].

Within the rationalistic paradigm, and specifically for experiments, Campbell describes validity as consisting of two major criteria: internal and external validity [25], where internal validity determines the strength of the inference from the independent to the dependent variable, and external validity determines generalizability.

After introducing these criteria, Campbell quickly noted that the two can easily become incompatible, stating that “the controls required for internal validity often tend to jeopardize representativeness” [25].

The two criteria for validity were later complemented with two additional criteria for validity: statistical conclusion validity, and construct validity [38].

All four terms are defined in the following subsections, as well as the terms reliability and objectivity. An extensive list of potential threats to each of the four dimensions of validity and suggested measures to increase the strength of each can be found in Shadish [180], we will discuss the relevant issues to our research while discussing our experiment in section 7.1.4.

Statistical Conclusion Validity

Statistical conclusion validity is “the validity of inferences about the correlation (covariation) between treatment and outcome” [180]. Statistical conclusion validity is closely related to and, in the case of quantitative experiments, a necessary condition for internal validity, requiring the correct use of statistics to infer a statistically significant correlation between the independent and dependent variable.

Internal Validity

The degree of internal validity is defined as the answer to the question “did in fact the experimental stimulus make some significant difference in this specific instance?”[25] Internal
validity builds on top of *statistical conclusion validity* and determines the strength of the inference of causality from the observed correlation [180].

**Construct Validity**

*Construct validity* is “the validity of inferences about the higher-order constructs that represent sampling particulars” [180]. As such it is a measure of generalizability with regards to the constructs. High construct validity allows for inference from the operationalization in the experimental setting to the desired abstract concepts the research is supposed to represent.

While some constructs and their measurements such as length or weight are usually without need for interpretation, other constructs such as quality, or performance of some object, human perception, or sentiment, are less clearly defined. *Construct validity* requires that the constructs under investigation are clearly and correctly defined as well as operationalized, to allow the inference from the observed measurement to the construct under investigation.

**External Validity**

External validity begs the question “to what populations, settings, and variables can this effect be generalized?”[25]. Like *construct validity* it is a measure of generalization, however, in particular, the generalization to a different population, setting, outcome, or a variation of treatment.

**Reliability**

*Reliability* is the degree, to which the measurement instruments are consistent [79, 66].

*Reliability* is a prerequisite to *statistical conclusion validity* since, without reliable and repeatable measurements, inferences about the correlation between the independent and dependent variable become invalid.
Measures improving the reliability of measurements include an increased number of measurements and better training of raters applying the measurement [180].

Objectivity

*Objectivity* is the extent to which the results are free from researcher bias [79, 66]. Objectivity is also a prerequisite to statistical conclusion validity, since a biased research design may confound the observed correlation between the independent and dependent variable.

Practices, such as strict adherence to methods, triangulation, and randomization, play a vital role in ensuring objectivity. Still, Sandelowski et al. write about objectivity in quantitative research synthesis, that “objectivity continues to reside in the defense and documentation of largely subjective judgments” [174].

7.1.3 Analysis of Rigor in Exploratory Cases

In this section, we discuss the four exploratory cases which facilitated the iterative development of our method and a preliminary evaluation. The study design and results were presented in detail in chapter 3. Since this was a purely qualitative study, we discuss rigor within the framework of the naturalistic research paradigm.

Figure 7.2 presents an overview of the measures taken within our research to ensure rigor within the naturalistic framework.

Credibility

Our confidence in the credibility of our four exploratory studies is strengthened by the fact that they were carried out over a period of six months each, and over the total range of three years. We believe, that for our research this constitutes adequate prolonged engagement and persistent observation.
With four different cases over three years, we committed to *prolonged engagement*. Opportunity to reflect on biases and adjust during a study, or for the next one, was plenty. And the iterative nature of our approach allowed for adjustments. With regard to letting participants adjust, this was not possible in every instance, since some of the experts we interviewed were only available for a single interview. However, we addressed this when possible with a follow-up interview. In one of the studies, we sat in a longer running series of meetings, where we only observed, took notes, and reflected to the participants what we had observed.

Although *persistent observation* is not focused on the time span of a study, it still benefits from the prolonged engagement we committed to. Further, the relevance of findings to our core category was also assessed regularly within the iterative coding process during *selective coding*. The coding process as a whole allowed us to identify conflicts and irregularities and to
follow up on them in the course of the study, thus identifying typical and atypical responses.

We also employed extensive peer debriefing [189] sessions with researchers within our group, gathering feedback for defining the research question, data gathering and analysis. Within such a session, we laid out our design, our plans, and our findings to the critique and questions from our peers. This detachment from the fieldwork and insight from researchers external to the project proved extremely valuable. The combination with our iterative approach of four independent studies was especially helpful to address any feedback, even if it may be difficult to adjust the currently running study.

We made extensive use of triangulation to ensure, that our findings were not the product of potential biases. We employed data triangulation by not only analyzing the expert interviews, but corroborating information with supplemental material from other sources such as existing documentation and requirements, workshop transcripts, and field notes.

Having multiple coders corroborating each other’s work strengthens the objectivity of the results [93]. In each of the four studies, we had at least two investigators independently analyzing the data, corroborating, and challenging each other’s work. This naturally provides investigator triangulation. The peer debriefing sessions provide a limited amount of theory triangulation where the theory needs to be corroborated by a researcher external to the project. Method triangulation was only used in terms of the different methods for data gathering, using observation, interviews, and gathering of public data.

To guard against researcher bias, member checks can be used to confirm that the results of the analysis reflect the understanding and experience of the participants. During the data gathering, we used reflective listening to confirm that we understood our participants correctly. We also conducted follow-up interviews with some participants to clarify our understanding of their perspective. After the iterative data gathering analysis had concluded, the resulting artifacts were then sent out to the participants with survey questions for the evaluation. The iterative nature of our studies made the feedback we received impactful. According to Guba, using member checks “is the single most important action inquirers can take, for it
goes to the heart of the credibility criterion” [79].

Of the practices described so far, all but member checks are required while performing the study and guiding its future direction. Member checks were also used in this regard, and additionally as an evaluation method after the data gathering and analysis concluded.

We achieved establishing structural corroboration or coherence by following the constant comparison method during coding of the data. After each interview, we cross-checked all new findings with previously gathered data to ensure that we didn’t miss any evidence for the new concept, and specifically looked for contradictions by following advice on attention on negative cases, or negative case analysis, for disconfirming evidence [120, 42].

While during the studies we made use of data triangulation as explained above, to achieve referential adequacy we also compared the results of the case within the HR domain against an ontology on the very same topic of HR development, that was published independently of our research. While the other practices are vital for guiding the research process, establishing referential adequacy is important to judge the results against a point of comparison. In the validation part of our work, we also generated more artifacts for comparison in this particular study.

**Transferability**

*Purposive sampling* is one of the recommended practices to strengthen transferability. Within the four studies, we aimed for theoretical sampling as a frequently recommended purposive sampling strategy. However, we were frequently constricted due to the availability of possible participants. In the larger scope of the four studies, we ensured that they take place in different domains, and with a different focus on a specific aspect of the method that was under development. We are therefore confident that the overall sample of data we gathered within all four studies encompasses a wide variety of contexts. With this, the study results should easily translate to new domains as well. We did not experience any impact of the type of domain towards the ability to execute our method as intended within the context of each
study.

We further believe, that our extensive use of triangulation helps our case for transferability.

Dependability

Previously, in section 7.1.3 on credibility, we laid out how we approached method triangulation, investigator triangulation, and peer debriefing. These practices also contribute to dependability.

Our decisions for the design of our research have been documented and published in our research article on the method [107], and various final thesis that contributed to our exploratory cases [116, 176, 177, 131]. All interviews were transcribed and coded. Through the coding, the analysis is traceable to sections of the transcribed expert interviews in the project file of the QDA. All additional meta data necessary for the derivation of our final artifacts was documented in spreadsheets and code memos, constituting our audit trail.

Confirmability

We also practiced reflexivity. We explained the purpose of our research to all participants and were open about our epistemological stance. Since most of our data gathering during the exploratory study relied on interviews, we were aware of the influence of the interviewer as an instrument of data generation. Therefore, all studies had at least two investigators, and the interview guidelines were always cross-checked by both. We also conducted member checks to reduce researcher bias and accurately document the phenomenon the participant’s point of view.

We also performed peer debriefing, and an early draft of our journal article on the exploratory cases and the development of the method was critiqued in a writer’s workshop [89] to collect feedback from our peers.
7.1.4 Analysis of Rigor in Experiments (Rationalistic)

In this section, we discuss the rigor of the controlled experiments presented in chapter 6 within the rationalistic research paradigm. Since the experiment had a mixed-methods design, analyzing both quantitative and qualitative data, another discussion of research rigor within the naturalistic paradigm is presented separately in section 7.1.5.

Figure 7.3 presents an overview of the measures taken within our research to ensure rigor within the rationalistic framework.
Figure 7.3: Measures Ensuring Rigor in Theory Validation (rationalistic)
Table 7.2: Threats to Statistical Conclusion Validity

<table>
<thead>
<tr>
<th>Threat</th>
<th>Consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low statistical power</td>
<td>Non-significance of inference may be incorrectly assumed.</td>
</tr>
<tr>
<td>Unreliability of treatment implemen-tation</td>
<td>Effects may be underestimated if treatment is not fully implemented equally for all participants.</td>
</tr>
<tr>
<td>Extraneous variance in the experimental setting</td>
<td>May introduce confounding factors.</td>
</tr>
<tr>
<td>Inaccurate effect size estimation</td>
<td>Measurements may be skewed due to outliers.</td>
</tr>
</tbody>
</table>

Statistical Conclusion Validity

Table 7.2 presents an overview of the threats to validity discussed in this chapter, and their potential consequences.

One of the most significant threats to statistical conclusion validity is low statistical power. We have set the type I error rate for rejection of a true null hypothesis at $\alpha = 0.05$. This value is recommended practice in a social science setting [180, 41].

A significant factor that may lead to low statistical power is an inadequate sample size. To further increase the statistical power, an increase of the sample size is a recommended measure, however, this was not feasible due to resource constraints. A larger sample would have required more funding, but the most significant reason for this constraint was that we accepted the trade-off between statistical power and a qualitative in-depth analysis. Because we wanted to interview each participant between each iteration, the effort required per participant increased manyfold compared to a purely statistical evaluation of the results. While the experiment itself lasted four weeks, the analysis of all gathered data lasted over 6 months. We feel validated in the decision of weighing a potentially larger sample size against gather-
ing more in-depth qualitative data by the many times the qualitative data yielded meaningful insights into possible explanations for quantitatively observed phenomena. Within our mixed-methods approach, the two forms of data gathering and analysis complemented each other.

For small sample sizes, a stratified assignment is recommended [180, 128] to ensure that both groups have equal representation for the known and significant covariates. We applied this extensively, however, as the number of controlled covariates was significant with regard to the total number of participants, it was not possible to create matched communities and sample participants randomly, as suggested by Gail [70]. Instead, we applied stratified sampling of participants into two groups, and then applied randomization to the assignment of treatment to a group to prevent researcher bias. The stratified assignment, considering factors such as the self-assessed competency in different dimensions, helped reduce the threat of heterogeneity of units (respondents) [180].

Another threat to statistical conclusion validity is unreliability of treatment implementation. Since both groups learned a new method respectively, this problem is not exclusive to the treatment group so it is not a bias towards one group or the other. However, we observed incorrect application of the method in both groups, and have addressed the anticipated issue through four introductory sessions (two for each group), each three hours long. The iterative nature of our experiment and the semi-structured interview after each iteration allowed us to correct misunderstandings and incorrect application of the method. Within each of the interview sessions, we asked participants to present a brief walkthrough of their process and explain the resulting intermediate and final artifacts.

Shadish further mentions extraneous variance in the experimental setting as a potential threat, especially in field experiments [180]. Since our experiment is indeed not a laboratory experiment, we were exposed to this threat. Participants performed their work at their own convenience within their private environment. We believe the following measures of our experiment design help mitigate this threat. We required each participant to keep a study
diary, documenting each work session. We further had the ability due to our mixed-methods approach, to inquire about potential covariates in the environment of the participants, and to document them. All environmental aspects that were reported to us were temporary, like travel, or a busy week within their educational setting. We always followed up on these issues in the following week. The fact that our experiment was conducted in four iterations, where each iteration had a duration of one week, helped mitigate the influence of such temporary effects.

The threat of inaccurate effect size estimation \[180\] can be caused by having outliers in one group or the other. “Outliers inflate the standard error of the sample mean, which, in turn, lowers power” \[211\]. We addressed this issue by choosing the Kruskal-Wallis test \[115\] as an evaluation method, an ANOVA that is based on ranks.

**Internal Validity**

Threats to internal validity are any effects that might be responsible for the observed correlation other than the cause-effect relationship between the independent and the dependent variable.

Table 7.3 presents an overview of the threats to validity discussed in this chapter and their potential consequences.

Eliminating the threat of ambiguous temporal precedence \[180\] is a necessary requirement for any measure of internal validity. It requires making sure that the cause precedes the measured effect. This is not a concern in our experimental setup, since the measured effects in the resulting artifact were a direct result of the method to learn and apply, which was the independent variable.

Threats to internal validity through selection were mitigated by stratified assignment of participants. We considered possible experience in the domain to be analyzed, work experience in IT, as well as the self-assessed competencies in domain modeling and QDA, apart from general demographic characteristics such as gender and age. The assignment of treat-
<table>
<thead>
<tr>
<th>Threat</th>
<th>Consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambiguous temporal precedence</td>
<td>Uncertainty, which variable is cause and which is effect.</td>
</tr>
<tr>
<td>History</td>
<td>Confounding factors due to concurrent external events.</td>
</tr>
<tr>
<td>Maturation</td>
<td>Confounding factors due to natural changes.</td>
</tr>
<tr>
<td>Regression artifacts</td>
<td>Regression to the mean may be confused with a treatment effect for previously extreme measurements.</td>
</tr>
<tr>
<td>Attrition</td>
<td>Participants dropping out may skew results if the dropout rate is correlated with the observed effects.</td>
</tr>
<tr>
<td>Testing effects</td>
<td>Results of a test may influence participants’ test in future exposure to the same test, independently of treatment.</td>
</tr>
<tr>
<td>Changing instrumentation</td>
<td>Confounding factors due to changes in the measurement instruments.</td>
</tr>
</tbody>
</table>
ment was random.

The history threat to internal validity refers to potential outside events that might lead to the measured outcome, such as changes in the life situation of a participant [180]. A proposed measure to counteract this threat is making sure that the data schedule is the same for both groups, rather than testing one group first and then the other or having longer or shorter time-spans between measurements for one group [136]. Also, similar life experiences from outside the study can be expected due to a homogeneous and stratified sample. We followed this advice, however, since we could not observe our participants during the complete four-week duration of the experiment, this threat can not be eliminated.

Maturation as a threat to internal validity is defined as “natural changes that would occur even in the absence of treatment” [180]. Since the study required multiple sessions every week for four weeks, the effects of becoming less concentrated during a long session are less relevant and assumed equal in both groups. However, with a duration of four weeks, the study was not long enough for significant changes to the cognitive ability of our participants could have an impact due to aging. A serious threat concerning maturation would have been carry-over effects after spending time with the data to be analyzed using any method, which is why a within-subject design was not possible, and we opted for a between-subject design.

The threat of regression artifacts concerns the effect that participants with an extreme measurement, are likely to have a less extreme measurement if tested again (regression to the mean [191, 142, 136]). With the study diary and the interviews, we had an element of a repeated-measures design, mitigating this effect. The score on the final artifact was the result of four weeks of cumulative work, so the quality of the artifact is not the result of performance in a single session.

Attrition is another common threat to internal validity [180]. It occurs when participants drop out of the study before it concluded. All participants were free to withdraw from the experiment at any time, and one participant did make use of this right. This meant that the differential of our finely tuned controlled variables between both groups slightly grew. We
believe the homogeneity of both groups within the considered variables remained good, with exception of the ratio of male to female participants, which we believe to be a less significant factor than the self-assessed competencies.

*Testing effects* are prevalent in our design, however, we do not consider this a threat to validity in the sense that motivation to learn the respective method was a desired effect in both groups. The significant impact of *testing effects* also required our between-subject design.

*Changing instrumentation* is another potential threat to *internal validity* [180], which we eliminated since instrumentation remained constant throughout the study.

**Construct Validity**

Table 7.4 presents an overview of the threats to validity discussed in this section, and their potential consequences.

A fundamental threat to *construct validity* is *inadequate explication of constructs* [180]. We have explicated the independent, dependent, and controlled variables in our experiment in section 6.1 and we were also explicit about the properties of our method we wanted to validate through the experiment.

*Construct confounding* is a threat to *construct validity*, which means that representations within the sample of the experiment are rarely pure manifestations of a single construct [180]. This applies especially to studies with human participants. We mitigated this effect through consideration of the controlled variables. However, it is impossible to eliminate all possible dimensions of above or below average performance in our experiment, so this threat could be mitigated but not eliminated.

*Monomethod bias* refers a the threat to *construct validity* which arises due to a single method of measuring. We addressed this through our mixed methods approach of statistical analysis, study diary, and semi-structured interviews.

Since our independent variable is binary, threats to *confounding constructs with levels of constructs* [180] are not relevant to the independent variable. The dependent variable, conversely,
### Table 7.4: Threats to Construct Validity

<table>
<thead>
<tr>
<th>Threat</th>
<th>Consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inadequate explication of constructs</td>
<td>Inference from operationalization and construct may be incorrect.</td>
</tr>
<tr>
<td>Monomethod bias</td>
<td>Single method may introduce bias towards a particular result.</td>
</tr>
<tr>
<td>Confounding constructs with levels of constructs</td>
<td>Generalizations from partial operationalization of a construct may be invalid.</td>
</tr>
<tr>
<td>Treatment-sensitive factorial structure</td>
<td>Being exposed to treatment may cause one group to experience the test itself differently.</td>
</tr>
<tr>
<td>Reactive self-report changes</td>
<td>Participants may be motivated to self-report inaccurately for a perceived benefit.</td>
</tr>
<tr>
<td>Compensatory equalization</td>
<td>Participants in the control group may receive compensatory assistance for not being in the treatment group.</td>
</tr>
<tr>
<td>Reactivity to the experimental situation</td>
<td>Environmental effects of the experiment may be part of the treatment construct.</td>
</tr>
<tr>
<td>Experimenter expectancies</td>
<td>Participants may wish to conform to perceived expectations.</td>
</tr>
<tr>
<td>Compensatory rivalry</td>
<td>Motivation may differ based on the assigned group.</td>
</tr>
<tr>
<td>Treatment diffusion</td>
<td>Participants being exposed to treatment they were not assigned to invalidates potential inferences.</td>
</tr>
</tbody>
</table>
is explicitly measured continuously and then ranked for the statistical analysis to eliminate the effect of outliers.

We believe our measurements, the study diary, the interviews, and the evaluation of the final artifact, are independent of the treatment. This eliminates the threat of treatment-sensitive factorial structure, which occurs when the instrumentation changes based on exposure to treatment [180].

We offered no incentive to participants that would make an assignment to either group seem more desirable. Both methods were previously unknown to all participants, and financial compensation was equal in both groups. This eliminates the threat of reactive self-report changes, which occurs if participants try to appear more suitable for a particular group. This effect sometimes does not disappear after assignment [180]. The lacking benefit of being in either group also eliminated the threat of compensatory equalization [180] where the control group is treated unusually beneficial as compensation for not having the benefit of the better treatment. Without a clear benefit to the participants of one group, there was also no reason for resentful demoralization [180] of being in the less desirable group.

Reactivity to the experimental situation, may trigger participants to react in a way, that they assume is expected from the experimenter, even if it may contradict their instruction [180]. This is also referred to as the good subject effect [148, 166]. Recommended measures to mitigate this threat include making the dependent variable less obvious and performing measurements on it after the conclusion of the participation. Participants were aware that we were evaluating the quality of their final artifacts. An incentive to produce a “good” final result was desirable in both groups. However, we did not reveal our method of measurement. The quantitative measurement did indeed occur after the active participation of subjects. The qualitative data gathered after each iteration, however, needed to be collected during the participation. We further followed guidance to reduce the threat of reactivity to the experimental situation by standardizing part of our interaction with subjects through the use of a study diary, which remained constant during the experiment. The guarantee of anonymity and
confidentiality also helped to reduce the participants’ desire to display the good subject effect.

The related threat of *experimenter expectancies* \(^{180}\) may lead to participants performing better or worse due to the researcher’s corresponding expectation. This phenomenon, also known as the pygmalion effect has been shown in educational research \(^{68}\) and corporate settings \(^{56}\), however in our setting participants never received direct evaluative feedback during the active participation. We were still conscious of this problem during the interviews. Our instruction during the interviews was limited to correcting false application of the respective method, not the results. While participants were performing the tasks or filling out the diary we were not present.

There was no *compensatory rivalry* \(^{180}\) that we could observe. Participants did not know each other, to the best of our knowledge, or had any indication of the performance of the other group.

*Treatment diffusion* \(^{180}\), where some participants receive both treatments did not occur since we only instructed each participant in the use of the one method respective to their group. It is conceivable but highly unlikely that any participant found out about the method of the other group and studied it without being instructed to do so.

**External Validity**

*External validity* should be evaluated in four types of generalization: Interaction of the causal relationship *with units*, *over treatment variations*, *with outcomes*, and *with setting* \(^{180}\).

Table 7.5 presents an overview of the threats to validity discussed in this chapter and their potential consequences.

With regards to generalization to other units, we would expect similar results with student participants from a similar educational background. We observed the learning curve for the method in the QDAcity group to be similar to the learning curve we observed with students applying QDA for research purposes in our course on research methods. In this different setting, we could observe hundreds of students who participated in homework on QDA
<table>
<thead>
<tr>
<th>Threat</th>
<th>Consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interaction of causal relationship over units</td>
<td>Observed effects may not hold for other types of participants.</td>
</tr>
<tr>
<td>Interaction of causal relationship over treatment variations</td>
<td>Observed effects may not be generalizeable to variations of treatment.</td>
</tr>
<tr>
<td>Interaction of causal relationship with settings</td>
<td>Observed effects may not hold in settings other than the experimental setting.</td>
</tr>
<tr>
<td>Interaction of causal relationship with outcomes</td>
<td>Measurement of outcome may be too specific for generalization.</td>
</tr>
</tbody>
</table>

using the same tool that the QDAcity group used for their analysis. Whether results from experimentation with students may be generalized to professionals is a point of contention \[54, 185\]. However, as Tichy points out, experiments with student subjects in software engineering may show trends that hold in a professional setting \[199\]. In fact, we are only interested in the differential of both groups. And while we believe the overall performance would increase with participants more experienced in conceptual modeling, this should have a similar effect in both groups.

A significant factor for the generalization to other units is *sampling bias*. Recruits may, as a group, be significantly different from the larger population depending on their motivation for participation. We believe the financial compensation was just high enough to find a sufficient sample but low enough so it was not the single motivation for our participants. Many participants stated their interest in research and were curious about the experience of participation. An above-average curiosity might have been present in our sample. How impactful this factor would be to the outcome when generalizing to a less motivated population we can not estimate. However, when comparing both groups’ self-portrayal in the semi-structured interviews, we observed no difference in this characteristic due to the independent variable.

For the generalization to a larger population, we believe the small sample size has a signifi-
cant negative effect.

A significant aspect regarding the interaction of causal relationship over treatment variations is the duration of treatment. Even though the experiment was performed over multiple weeks, the situation could change if the respective method of each group had been employed for a more extended period of time. We are confident that some trends in the data we gathered would continue over an extended period. The gap of effort expended widened throughout the experiment so that the control group spent significantly more time each week as the experiment continued. This correlates with qualitative data we gathered from the interviews, that the mind map was becoming more unwieldy the more data was processed. This effect existed also for the code system but was less prominently displayed by participants. Also, the learning curve seemed slightly more challenging in the QDAcity group from what we could gather in the interviews, and this is consistent with the diary data on the confidence rating, which was significantly lower in week two, but then caught up to the level of the control group. This suggests to us, that the benefits of our method were mostly beginning to materialize in the second half of the experiment.

A potential threat to external validity regarding the variation of treatment can also occur as a direct effect of scaling. For example, when social interactions in larger communities differ from those in smaller communities. While we don’t expect this effect directly in our context, we did encounter evidence that the situation would be different if multiple analysts would collaborate on the same project. Participants of both groups have mentioned in the interviews, that the method they were using might be of particular benefit when collaborating with multiple participants. The collaboration effects on our method need further study.

Another aspect of external validity is the generalization of the outcome. This requires openness and anticipation of questions that are likely to be asked about the outcome of the study [180]. We believe this aspect is addressed sufficiently by our experimental design through the inclusion of open-ended questions in the study diary, and regular semi-structured interviews.
Questions about the interaction of causal relationship with settings have a single focus within the context of our research question since the target setting is highly focused on the engineers’ understanding a domain with the purpose of system development. In contrast, the setting in our experiment was participants performing the analysis at home. We are confident that the setting is less relevant to our research question than the generalizability with regard to units, outcomes, and treatment variation.

Reliability

The reliability of our results was ensured by triangulation. Three types of triangulation were employed.

Investigator triangulation was achieved through regular peer debriefing, as well as through three sessions of inter-rater reliability.

Method triangulation was achieved through the mixed-method design, which also yielded multiple types of data to be analyzed, providing data triangulation.

Repeating measurement is one of the recommended practice for strengthening reliability [180]. The diary study followed a repeated-measures design, with the same questions being answered after each work session, leading to more reliable aggregated data.

The iterative nature of our experiment design also helped us gathering more reliable data than in a much shorter laboratory experiment. If the deviation of any particular participant was significant, regarding either the artifacts or the process, we were able to follow up on this in the next interview session. This way, we documented factors leading to the deviation. All of the identified factors were independent of the measurement instruments.

We believe the sample size was sufficient for identifying trends over the period of four weeks, and the trends were corroborated by the qualitative interview data.

The measures we described about statistical conclusion validity also helped the reliability of our measurements. In the chapter on statistical conclusion validity we laid out how we addressed the threats of inadequate sample size, heterogeneity of units (respondents), reliability
of treatment implementation, extraneous variance in the experimental setting, and inaccurate effect size estimation.

Objectivity

Investigator triangulation through peer debriefing also helped to limit researcher bias, and thus strengthen objectivity.

We controlled for possible covariates by surveying potential participants on the relevant confounding factors and then used stratified sampling to ensure a similar distribution of all variables across both groups. Objectivity was then ensured by randomization of the applied treatment.

Both groups received an equal amount of guidance being introduced to the methods and corrected when deviations from the respective method were reported in the weekly interviews.

The measures we described in the previous sections on statistical conclusion validity, internal validity, and construct validity show that we followed methodological guidelines to avoid any effect that would impede the inference of a correlation between the observed effects all the way to the underlying construct we aimed to measure.

7.1.5 Analysis of Rigor in Experiments (Naturalistic)

In this section, we discuss the rigor of the controlled experiments presented in chapter 6 within the naturalistic research paradigm. Since the experiment had a mixed-methods design analyzing both quantitative and qualitative data, another discussion of research rigor within the rationalistic paradigm is presented separately in section 7.1.4.

Figure 7.4 presents an overview of the measures taken within our research to ensure rigor within the naturalistic framework.
Credibility

To ensure credibility within our experiments, we employed extensive peer debriefing [189]. The design was laid out for and critiqued by four researchers. Each interview guideline was discussed with one colleague and if necessary revised as a consequence.

We also debriefed three colleagues on the coding procedure, who agreed to code a portion of the 40 interviews for inter-rater agreement. Thus, we also had investigator triangulation.

Data triangulation was achieved through the inclusion of qualitative and quantitative data, the gathering and analysis of which also required us to make use of method triangulation. The only type of triangulation we did not employ was theory triangulation since we consider all researchers involved with the analysis of data to be internal to the study.

Prolonged engagement was not as strong as in our exploratory studies, but in the context of a controlled experiment, the iterative setting over multiple weeks allowed the participants adequate time to adjust to the setting of the study. It also allowed us to follow up on issues that established themselves as critical for an individual participant and observe changes over
time.

The “extended interaction with a situation” necessary for persistent observation was given by the repeated gathering of qualitative data from multiple participants per group. This allowed us to identify both typical and atypical responses over time, and across participants.

To establish structural corroboration or coherence we used the constant comparison method while coding, which required us to compare any new findings with previously analyzed data and confirm or disconfirm any previous analysis results through new evidence. We explicitly gave attention to negative cases, to identify differences within the experiences of our participants.

Referential adequacy was established by our baseline model which was previously evaluated by domain experts, against which we compared results from both groups.

Transfearability

Our purposive sampling and stratified assignment contribute to transferability. However, in contrast to the exploratory studies, where we aimed for a diverse sample, for the purpose of the controlled experiment we strived for a more homogeneous sample.

The mixed-methods approach also inherently provides triangulation, which further strengthens transferability due to more reliable results.

Guba further recommends collecting data that leads to a thick description, which describes the context of the findings, enabling a comparison to any different context to which transferability of the findings may be considered [79]. The context within a controlled experiment is naturally more abstracted than in a case study, and the description of it is not as detailed. Still, we believe that within the methodological framework we were able to collect thick descriptive data through the diary study and interviews, which contextualizes and explains the quantitatively observed evidence.

A threat to the theoretical generalizability, and thus transferability is the abstracted nature of a controlled experiment. We believe the trade-off we consciously made here to enable a
direct analysis of potential correlations and to provide a reliable point of comparison paid off. In return, we accept that the context of the environment during the experiment was more abstracted than in a case study.

Within the interviews, participants hypothesized on potential contexts, that would benefit from the methods they were employing during the study. These include contexts, where collaboration between multiple stakeholders is of high importance, and projects that run for long periods of time, where limitations in long-term memory require more detailed documentation.

**Dependability**

As recommended by Guba, we performed *method triangulation*, *investigator triangulation*, and *peer debriefing* [79].

*Method triangulation* was established through the mixed-methods approach combining statistical analysis with qualitative interviews and a diary study.

*Investigator triangulation* was established through three rounds of inter-rater reliability.

Further, we documented all steps from recruitment, assignment, planning, and execution of interviews, the diary entries, as well as intermediate and final artifacts the participants were working on. This thorough documentation constitutes our *audit trail* [79].

The *audit trail* was used in our *peer debriefing* sessions, where we presented our planning documents and design decisions to researchers within our group and had them question, critique, and improve our research design.

**Confirmability**

Within the experiment we also *practiced reflexivity*, as recommended by Guba [79]. The purpose of the study was explained to all participants, and they were given an informed consent form (see appendix A.1) explaining the study.

Both groups received an in-depth introduction into the study, where they were encouraged
to ask questions, and where the purpose of the study and our epistemological stance was made clear.

During the interviews we made sure, there was always room to deviate from our interview guidelines to follow aspects that our participants felt were important.

Investigator triangulation was used to reduce researcher bias and thus strengthen confirmability. This was achieved through the three inter-rater reliability sessions. The peer debriefing, as well as the random assignment of treatment to one group further contributes to strengthening confirmability.

7.1.6 Conclusion

We evaluated both the theory building and the theory validation of our work with the respectively appropriate frameworks for research rigor.

We evaluated the qualitative theory building work within the framework of trustworthiness [79, 120]. We presented strong evidence that our research methodology fostered all four categories of this framework. We implemented most practices recommended by Guba [79].

The mixed-methods theory validation work was evaluated both in terms of trustworthiness and traditional rationalistic validities. For the qualitative aspects of the experiment, we implemented many of the same practices we employed for theory building. The mixed methods approach itself did yield exceptional method triangulation, unearthing explanations for many observations that would otherwise be lost.

We considered all potential threats to validity mentioned by Shadish [180] with regard to our experiment, and followed mitigating advice where appropriate and feasible.

7.2 Future Research

Beyond the scope of the research presented in this thesis, we consider the following areas for potential future research of particular importance.
7.2.1 Extension to Other Types of Artifacts

Within our exploratory studies, we already analyzed the capability of the QDAcity-RE method to be used for the creation of various types of artifacts besides conceptual domain models. Using QDAcity-RE, we have created process models, DSLs, glossaries, and natural language RS. However, the efficacy of the method for these artifacts was never validated by us, as we did for conceptual models with the controlled experiment presented in this thesis.

Among the artifacts QDAcity-RE could be extended to, the creation of a natural language RS based on the method is expected to be of the highest utility. A follow-up project to the work presented in this thesis, extending the QDAcity-RE method in this regard, has already started.

7.2.2 Validate Inter-Model Consistency

Dependent on the extensions of the method to other artifacts (section 7.2.1), one of our expected benefits, an improvement on inter-model consistency, could be more thoroughly investigated in future research. It is our vision, that the code system can function as a unified model where all other artifacts downstream of the requirements elicitation can be derived from.

Through the code system, all artifact elements in different models would therefore be interconnected, meaning changes performed in one artifact would automatically be mirrored in all other impacted artifacts. This would ensure consistency among them.

The final thesis by Schmitt, performed under our supervision, was done on the topic of integrating multiple views into the code system [177], and we consider it preliminary work on this research area.

7.2.3 Impact on Change Management

With all original stakeholder material being traceable from the requirements, it is another promising avenue for research to investigate the impact of applying QDAcity-RE to change
management of requirements.

Of particular focus might be the possibility to identify dismissed alternatives to the current version of a model element and the documentation of how potential conflicts were resolved. Through QDAcity-RE, the genesis of each requirement, and all potentially conflicting stakeholder statements that are relevant can be easily identified through traces.

7.2.4 Supporting QDA Through NLP

The process of QDA, in QDAcity-RE or otherwise, is labor intensive. We performed early studies on supporting the human analyst through NLP [104, 105]. Our vision is a recommender system that provides suggestions for autocoding. Barik et al., also implemented a prototype for automated knowledge extraction as a means to extend GT, and attest to its potential [10]. Other research into autocoding by Crowston et al. has also shown promise with a rules-based approach [43, 44]. A combination of the two approaches could prove beneficial and is suggested for future research.

Besides autocoding, we also investigated the idea of extracting relationships between the codes. Besides the relationship implied by the hierarchy of codes, NLP techniques for relationship extraction were explored through a master’s thesis from Hofmann, which was supervised by us [91].

Both of these approaches remain promising avenues for future research.

7.2.5 Investigation of Alternatives to the Coding Paradigm

Our experimentation revealed, that the coding paradigm was difficult to use for our study participants. And while benefits in the form of guidance where to focus attention were described by some participants, other participants were unable to utilize it as intended.

We suggest that future research should focus on creating and evaluating possible alternatives, or develop a more guided approach, possibly assisted through NLP, that makes the utilization of the coding paradigm easier.
7.2.6 Quality Metrics

QDAcity already supports metrics for saturation and inter-rater reliability. However, with the trace links established, further metrics on the resulting artifact are conceivable. Ananin [6] investigated possible metrics for conceptual models created with QDAcity, but further research would be required for a sensible integration into the QDAcity-RE process, and the QDAcity tool.

Also, further research to refine our configurable metric for saturation might yield useful results to improve the saturation metric or make it more flexible to unusual project contexts not yet considered.

7.2.7 Project Specific CSL

Under our supervision, Reuter [161] already investigated the use of domain knowledge in the creation of a code system in a novel project within a familiar domain, or research area. While the pre-structuring of data based on previous knowledge runs contrary to some popular QDA methodologies, it is common practice to speed up parts of the research that are repetitive.

Following up on this topic, an investigation of possible user defined CSLs that might be shared within a community of researchers, or a specific research group, would be of interest. This would provide a common framework, which is extendable to the contexts of a specific project. This work could impact QDA in research, as well as domain modeling for RE, since the situation of understanding a new sub-domain within an already well-known field occurs similarly in both types of analysis activities.
References


conference on software architecture workshops (ICSAW) (pp. 141–148): IEEE Computer Society.


A.1  Informed Consent

Following the ethical guidelines of our institution and good scientific practice [186] we acquired informed consent from every participant of the experiment by informing them about the purpose of the study, the use of the gathered data and their right to withdraw their participation at any time without giving a reason. The following form was signed prior to the start of the experiment.

**Research study on conceptual modeling**

*Consent Form*

This informed consent form is for the Open Source Research Group and who we are inviting to participate in our research titled “Domain Modeling using Qualitative Data Analysis” The study is led by Andreas Kaufmann and supervised by Dr. Dirk Riehle at the Open Source Research Group of the Friedrich Alexander University Erlangen Nürnberg (FAU). It is funded, in part, by Deutsche Forschungsgesellschaft (DFG) grant RI 2147/7-1.
This Informed Consent Form has two parts:

- Information Sheet (to share information about the study with you)
- Certificate of Consent (for signatures if you choose to participate)

You will be given a copy of the full Informed Consent Form

**Part I: Information Sheet**

**Introduction**

We developed a novel method for conceptual domain modeling, called QDAcity-RE, and invite you to participate in a field experiment for the evaluation of this method compared to an existing method for conceptual domain modeling. We will describe our research, and what your participation would entail, in this document. If there is anything that you do not understand, or if you have any kind of question regarding our research and the planned study, please ask us. You are also free to consult any third party on any of the information given here.

**Purpose of the research**

Defining the right requirements for a software project is an important factor for its success. To identify adequate requirements, a deep understanding of the target domain has to be developed. To facilitate such an understanding, and providing the means to share it among stakeholders, domain modeling can be employed. Unfortunately the process of domain modeling frequently relies a lot on intuition and experience of the analyst. Our method attempts to formalize the process to ensure rigor in its execution and also create traceability from developments artifacts beyond the defined requirements to stakeholder material the specifications were built upon (so-called Pre-RS-Traceability).

The study we are inviting you to participate in aims to evaluate the method compared to an existing method.

**Procedures and intervention**

The evaluation takes place in a field experiment. We will divide participants into two groups. One using QDAcity-RE and one using a method based on strategic reading. We will inform you in which group you can participate after signup to the study concluded. You will be instructed on the respective method by us and are encouraged to seek guidance w.r.t. the method.

Each week you will receive a set of one to two interview transcripts from expert interviews we already conducted. Your task will be to apply the respective method to
analyze the unstructured data with the purpose of creating a conceptual model using the assigned method. The analysis will take place without our direct supervision on your own time. You will be asked to document the time spent on specific parts of the analysis. During the full duration of the experiment you are not permitted to share any information about the data or your analysis artifacts with anyone except us. After each week we will have a one on one meeting and we will ask you to answer a few questions regarding the method and your experience.

Voluntary Participation

Your participation in this research is entirely voluntary. It is your choice whether to participate or not. If you choose not to participate this will not affect your status, rights and obligations as a student at FAU in any way. In particular your participation, or refusal of participation has no bearing on your participation or evaluation in any of the courses offered by the Open Source Research Group.

Duration

The study takes place over a period of five weeks. A potential extension may be warranted due to unforeseen circumstances such as illness. We expect the average effort for you to be around 8-10 hours per week. The study will take place in June / July of 2019.

Risk

We do not anticipate any immediate risks to you participating in this study other than those encountered in day-to-day life.

Benefits

Potential benefits are learning about HR development through the analysis of expert interviews, learning, and applying a new method for conceptual modeling, and if you are interested in a research career, it might be a benefit to experience such an experiment as a participant. We are happy to share information about questions you might have about the research process with you.

Other than these potential benefits, you have the benefit of a guaranteed compensation in the agreed-upon lump sum payment.

Compensation

For your completed participation in the study you will receive a compensation for your time of 500EUR.

Confidentiality
The information that we collect from this research project will be kept private to the Open Source Research Group. We intent, however, to publish aggregated statistics in scientific publications. We also may publish quotes from the interviews. Any information about you in textual form will have a number identifier on it instead of your name. The only people with access to an encrypted key file linking your identifier to your name will be Andreas Kaufmann and Dirk Riehle. The audio recordings of the interviews will be stored as-is private to the Open Source Research Group.

Right to refuse or withdraw

You do not have to take part in this research if you do not wish to do so, and choosing to participate will not affect your student status in any way. You may stop participating in the study at any time that you wish without impact on your relationship with the university or the Open Source Research Group. By doing so, you forfeit your claim to compensation.

Who to contact

If you wish to ask questions about the study or this form, you may contact any of the following persons responsible for planning and execution of this study:

Andreas Kaufmann: andreas.kaufmann@fau.de
Dr. Dirk Riehle: dirk.riehle@fau.de

The ombudsperson of FAU for good scientific practice is:

Dr. Dietmar Fey: dietmar.fey@informatik.uni-erlangen.de

Part II: Certificate of Consent

I have been invited to participate in research about conceptual domain modeling. I have read the foregoing information, or it has been read to me. I have had the opportunity to ask questions about it and any questions I have been asked have been answered to my satisfaction. I consent voluntarily to be a participant in this study.

Your Name: ____________________________________________________________

Your Signature: ______________________ Date (DD.MM.YYYY): ________________

In addition to agreeing to participate, I also consent to having the interviews tape-recorded.

Your Signature: ______________________ Date (DD.MM.YYYY): ________________
### General Information

<table>
<thead>
<tr>
<th><strong>What is your participant ID?</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Time spent</strong></td>
</tr>
<tr>
<td>Total number of hours. Use decimal point, not comma for fractions of an hour</td>
</tr>
<tr>
<td><strong>Description of tasks</strong></td>
</tr>
<tr>
<td>Describe all tasks performed in this session</td>
</tr>
<tr>
<td><strong>Session Goal</strong></td>
</tr>
<tr>
<td>What was the goal in this work session?</td>
</tr>
</tbody>
</table>

### Reflection on results

<table>
<thead>
<tr>
<th><strong>Resulting artifacts</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>What were the most significant changes to the artifacts of the study, most notably the domain model?</td>
</tr>
<tr>
<td><strong>Novel insights</strong></td>
</tr>
<tr>
<td>Was there any surprising or new insights into the domain?</td>
</tr>
<tr>
<td><strong>Confidence in results</strong></td>
</tr>
<tr>
<td>Please rate your confidence in whether the current state of your domain model represents the domain well? Please rate on a 7 point scale.</td>
</tr>
<tr>
<td>Reflection on process</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>Rating success of session</td>
</tr>
<tr>
<td>Please explain why you chose rating in the previous question.</td>
</tr>
<tr>
<td>Problems / Challenges</td>
</tr>
<tr>
<td>Finish</td>
</tr>
<tr>
<td>Additional comments</td>
</tr>
</tbody>
</table>
B

Conceptual Models
Figure B.1: Domain Model for HR-Development Exploratory Study (Expert Solution)
Figure B.2: Conceptual Model of Participant A1
Figure B.5: Conceptual Model of Participant A4
Figure B.6: Conceptual Model of Participant B1
Figure B.7: Conceptual Model of Participant B2
Figure B.9: Conceptual Model of Participant B4
Figure B.10: Conceptual Model of Participant B5
The code book presented in table C.2 emerged through analysis of 40 interviews with participants in our experiment. We followed the code book structure by MacQueen et al. [124], defining each code and giving explicit instructions when to use the code and when not to use the code. The latter was mostly used for disambiguation with another code, and was used optionally. The codebook was refined through three iterations of inter-rater agreement. The hierarchy of the code system is color coded as shown in table C.1.

Table C.1: Color Coding of Code System Hierarchy

<table>
<thead>
<tr>
<th>Color</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Level 1</td>
</tr>
<tr>
<td></td>
<td>Level 2</td>
</tr>
<tr>
<td></td>
<td>Level 3</td>
</tr>
<tr>
<td></td>
<td>Level 4</td>
</tr>
<tr>
<td></td>
<td>Level 5</td>
</tr>
<tr>
<td></td>
<td>Level 6</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>artifact description</td>
<td>Description of any of the resulting artifacts. Final or intermediate</td>
</tr>
<tr>
<td>interview</td>
<td>Description of the expert interviews</td>
</tr>
<tr>
<td>interview content is clear</td>
<td>The expert interview is clear in its content</td>
</tr>
<tr>
<td>interview is a difficult read</td>
<td>Interview is difficult to understand</td>
</tr>
<tr>
<td>interview is incomplete</td>
<td>The interview is described as incomplete, some information is missing</td>
</tr>
<tr>
<td>interview is insightful</td>
<td>The interview is described as insightful</td>
</tr>
<tr>
<td>structure of code system</td>
<td>Description of the structure of the code system</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>use of relationship types</td>
<td>Description on which relationship types the participant chooses</td>
</tr>
<tr>
<td>domain description</td>
<td>Description of the domain of HR development</td>
</tr>
<tr>
<td>method description</td>
<td>Description of the method to use in each respective group</td>
</tr>
<tr>
<td>description is incomplete</td>
<td>The description of the method is incomplete</td>
</tr>
<tr>
<td>description is accessible</td>
<td>The description of the method is easily understandable</td>
</tr>
<tr>
<td>description is actionable</td>
<td>The learned method can be put into practice.</td>
</tr>
<tr>
<td>description is clear</td>
<td>The description of the method is unambiguous and clear. Opposites should also be coded with this code</td>
</tr>
<tr>
<td>description of explicate</td>
<td>The explicate problem template is described as helpful</td>
</tr>
<tr>
<td>problem template is helpful</td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>description is motivating</td>
<td>The description is written in a way that is motivating</td>
</tr>
<tr>
<td>method is confusing</td>
<td>The method description is confusing, which makes it unclear how it is supposed to work</td>
</tr>
<tr>
<td>code system language</td>
<td>The code system language is confusing</td>
</tr>
<tr>
<td>method is misunderstood</td>
<td>Participant did not understand how the method is supposed to work</td>
</tr>
<tr>
<td>purpose unclear</td>
<td>The purpose of the method is not clear</td>
</tr>
<tr>
<td>terminology</td>
<td>Terminology of the method is not clear</td>
</tr>
<tr>
<td>motivation makes sense</td>
<td>The motivation of the method is clear</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>prior experience</td>
<td>Experience from before starting the participation in the experiment</td>
</tr>
<tr>
<td>HR experience</td>
<td>Experience in the domain of human resource</td>
</tr>
<tr>
<td>QDA experience</td>
<td>Experience with QDA, apart from the study</td>
</tr>
<tr>
<td>UML and modeling experience</td>
<td>Experience with creating class diagrams and UML</td>
</tr>
<tr>
<td>inferring from experience</td>
<td>The analysis has been guided by prior experiences in the domain</td>
</tr>
<tr>
<td>process</td>
<td>Category code for everything related to the process of reading, analyzing, documenting.</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>analysis process</td>
<td>The analysis process covering all stages from reading, annotating, creating the intermediate artifact and the UML model</td>
</tr>
<tr>
<td>UML modeling</td>
<td>The process of creating the final artifact of the study in either group: the UML class diagram.</td>
</tr>
<tr>
<td>finding relationships between concepts</td>
<td>The process of defining relationships between concepts</td>
</tr>
<tr>
<td>integration of UML modeling</td>
<td>Description of how the UML modeling is integrated into the method</td>
</tr>
<tr>
<td>UML and intermediate artifact intertwined</td>
<td>UML modeling is done in lockstep with creating the intermediate artifact</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>position in the process</td>
<td>The order within the analysis process, where the UML modeling is performed</td>
</tr>
<tr>
<td>UML modeling at the end</td>
<td>The UML model is created after the intermediate artifact</td>
</tr>
<tr>
<td>relationship of intermediate artifact to UML</td>
<td>The relationship of the code system or the mind map to the UML model</td>
</tr>
<tr>
<td>using the metamodel tab</td>
<td>The process of defining the metamodel attributes needed for mapping codes to UML in the properties window of the code rather than using the graphical modeler</td>
</tr>
<tr>
<td>behavioural aspects</td>
<td>Aspects of behavioural modeling</td>
</tr>
<tr>
<td>changes over time</td>
<td>Changes to the way the participant experienced the application of the method from week to week</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>changing granularity of annotation</td>
<td>The unit of coding was deliberately changed</td>
</tr>
<tr>
<td>evolution of model</td>
<td>Changes to either of the models. This includes the intermediate artifacts as well as the final UML model</td>
</tr>
<tr>
<td>increased efficiency</td>
<td>Increased efficiency of method application, due to experience, as compared to earlier weeks</td>
</tr>
<tr>
<td>initial approach</td>
<td>Description of the initial approach a participant took.</td>
</tr>
<tr>
<td>more confidence</td>
<td>Increased confidence in the application of the process or the quality of the resulting artifacts as compared to earlier weeks.</td>
</tr>
<tr>
<td>more detail</td>
<td>The artifacts becoming more detailed over time</td>
</tr>
<tr>
<td>changing reading purpose</td>
<td>The process of changing the reading purposes as defined in Mindmeister as one of the seven colors to tag</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>coding</td>
<td>The process of creating new codes, changing them, rearranging them and applying them to text segments</td>
</tr>
<tr>
<td>axial coding</td>
<td>The process of setting codes in relation to each other</td>
</tr>
<tr>
<td>open coding</td>
<td>The process of creating new codes as they emerge from the text</td>
</tr>
<tr>
<td>selective coding</td>
<td>The process of setting all codes in relation to the core phenomenon w.r.t. the coding paradigm and discarding those that are no longer necessary</td>
</tr>
<tr>
<td>context switch: reading/modeling</td>
<td>The problem of going from reading to modeling with the intermediate artifacts</td>
</tr>
<tr>
<td>deviations from method</td>
<td>Deviations in the application of the method from how it is described in the respective papers</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>familiarize with tooling</td>
<td>The process of getting to know the tools to be used during the study.</td>
</tr>
<tr>
<td>identifying knowledge gaps</td>
<td>The process of finding gaps in the current models that need more explanation</td>
</tr>
<tr>
<td>open questions</td>
<td>Things that are unclear about the domain at the current stage</td>
</tr>
<tr>
<td>method assessment</td>
<td>Positive or negative assessment of how well the method works for its purpose</td>
</tr>
<tr>
<td>coding paradigm is confusing</td>
<td>Aspects of the coding paradigm are unclear</td>
</tr>
<tr>
<td>coding paradigm is helpful</td>
<td>The coding paradigm is intended to help participants uncover knowledge gaps and guide the analysis</td>
</tr>
</tbody>
</table>

230
<table>
<thead>
<tr>
<th>Code</th>
<th>Full definition</th>
<th>When (not) to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>control group method is similar to coding</td>
<td>Statement that the method using strategic reading is similar to coding</td>
<td>When a participant with prior experience in QDA describes the method of the control group to be similar to coding</td>
</tr>
<tr>
<td>ease of application</td>
<td>The ease with which the method could be applied</td>
<td>Applies to any aspects of the method a participant describes as particularly easy or difficult to use. Both artifacts and process steps</td>
</tr>
<tr>
<td>effort required</td>
<td>The effort required to perform the method as intended</td>
<td>When participants describe the effort they expend, either in time, or qualifying that they had to put in a certain amount of effort to complete the task of analysis</td>
</tr>
<tr>
<td>helpful for collaboration</td>
<td>The method is being described as particularly helpful when collaborating with others</td>
<td>Participants describe explicitly, or implicitly, that certain aspects of the method would yield additional benefits when used in a collaborative setting</td>
</tr>
<tr>
<td>helpful for longer running projects</td>
<td>The method is helpful if a project is run over an extended timeframe, making documentation more important</td>
<td>When participants describe that they assess one of the strengths of the method to be that in longer running projects you can not work purely from memory to remember what you analyzed months in the past</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>method helpful in guiding the process</td>
<td>The method is helping participants by guiding them through the analysis process</td>
<td>Mentions when following the method proved beneficial, or was perceived as giving helpful and clear instructions. Comparisons to ad-hoc modeling or other methods. Do not use when: When the subject is the method description rather than the method application</td>
</tr>
<tr>
<td>template giving focus</td>
<td>The explicate problem template is described as giving focus</td>
<td>When participants describe the template useful in a way that guides their analysis</td>
</tr>
<tr>
<td>tool is helpful</td>
<td>Descriptions of how the tool helps, makes the analysis better or more efficient</td>
<td>When specific attributes of the software are praised as being particularly helpful</td>
</tr>
<tr>
<td>saturation</td>
<td>The state where the resulting artifacts are stable, and further gathering of data would no longer lead to significant changes</td>
<td>When participants talk explicitly about saturation, or implicitly about their feeling of how close their artifacts are to a final state</td>
</tr>
<tr>
<td>working from memory</td>
<td>The practice of not using documentation as much but rather working from memory</td>
<td>When a participant describes, that he or she does not need to look up something in the documentation but rather working from memory. This is frequently connected to descriptions of the time between interviews being short</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>challenges</td>
<td>Problems and challenges that participants encountered.</td>
<td>Any type of problem or challenge with applying the method. More specific sub categories should be favored over this code. Do not use when: When any of the sub categories apply</td>
</tr>
<tr>
<td>restructuring intermediate artifact</td>
<td>The problem of changing an existing hierarchy of the intermediate artifact. This could apply to either the code system or the mind map</td>
<td>When a participant mentions it as problematic to change the existing hierarchy or lists some challenges that are implicated thereby. Do not use when: When changing the hierarchy is problematic due to tooling issues, this code should not be used. Instead &quot;challenges-&gt;tooling problems&quot; should be applied.</td>
</tr>
<tr>
<td>finding good categories</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UML modeling is hard</td>
<td>The process of UML modeling is described as challenging.</td>
<td>Participants feel uncomfortable with UML, or they find it challenging to model in UML. This is likely highly correlated with a lack of experience with UML, but is more general encompassing any mention of UML modeling being a particularly difficult phase of the method.</td>
</tr>
<tr>
<td>artifacts are complex</td>
<td>Any of the artifacts (UML / mind map / code system) are difficult to handle due to their complexity</td>
<td>When participants describe that working with an artifact is difficult due to it being too complex. Do not use when: When strategies of dealing with this complexity are described</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>consistency between interviews</td>
<td>The practice of ensuring, that the domain model describes the domain in a way that tries to be consistent with all interviews</td>
<td>When the problem of ensuring consistency between interviews or identifying inconsistencies is described</td>
</tr>
<tr>
<td>determining relevance of concepts</td>
<td>The challenge of determining what is relevant for the domain</td>
<td>When it is described how to determine the relevance of a concept. This can be in any of the stages of the analysis: the annotating, the intermediate artifact, or the UML model</td>
</tr>
<tr>
<td>finding good categories</td>
<td>The problem of grouping concepts together in categories</td>
<td>When the problem of grouping concepts together in a hierarchy, either in the mind map or the code system is mentioned</td>
</tr>
<tr>
<td>finding the right category for annotation</td>
<td>The problem of finding the right concept for annotation when a text segment is deemed relevant</td>
<td>When a participant describes it as a challenge to select one of the existing categories or concepts or create an appropriate new one for a new annotation. The concept could be a code or a node in the mind map</td>
</tr>
<tr>
<td>granularity of analysis</td>
<td>The problem of choosing the right level of granularity in the analysis</td>
<td>When a participant is unsure what granularity to use in the analysis, or strategies of choosing the right granularity. Do not use when: When any of the sub categories apply.</td>
</tr>
<tr>
<td>granularity in model</td>
<td>The problem of choosing the right level of granularity in the UML model, or the intermediate artifact</td>
<td>When the problem of including broader or more specific concepts in the UML model or the intermediate artifact is discussed</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>granularity of annotations</td>
<td>The problem of choosing the right level of granularity in the annotation</td>
<td>When the problem of including broader or more specific concepts in the annotation is discussed. Do not use when: When any of the sub-categories apply</td>
</tr>
<tr>
<td>knowing when to apply codes</td>
<td>Having difficulty with the fundamental aspect of applying codes to text segments of QDA (open coding)</td>
<td>When the experienced problem points implicitly or explicitly to the problem of mastering the fundamentals of QDA with respect to applying codes to text segments (coding / open coding). Do not use when: When it is about the process description, this should be coded with the open coding tag in the process dimension of the code system. This code (knowing when to apply codes) is specifically about problems with this part of the process.</td>
</tr>
<tr>
<td>lack of experience</td>
<td>Missing experience that impedes efficient completion of tasks relevant to the study</td>
<td>When the participant self-proclaims a lack of experience. Also implicit evidence of lacking experience, for instance due to consistently false interpretation of UML notation. Specific sub-categories should be favored over this code. Do not use when: When a sub-category applies</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>lack of domain knowledge</td>
<td>Lack of knowledge in HR</td>
<td>When a participant describes that they know little (or too little) about the domain</td>
</tr>
<tr>
<td>lack of experience in UML</td>
<td>Lack of experience in UML notation</td>
<td>When a participant describes they don’t know UML well, or this becomes apparent in their use of the notation</td>
</tr>
<tr>
<td>lack of method competence</td>
<td>Lack of knowledge in the correct application of the method</td>
<td>When the lack of competence in the method to use. When a participant describes they are unsure how to apply the method. Do not use when: When the confusion stems from an inadequate method description. There is a separate code for this.</td>
</tr>
<tr>
<td>large amount of data</td>
<td>The amount of information in the data to analyze is overwhelming</td>
<td>Participants describe that they struggle due to the amount of new information</td>
</tr>
<tr>
<td>method is too complex</td>
<td>Description of the complexity of the method being problematic.</td>
<td>When a participant describes that he or she has trouble following the method due to its complexity</td>
</tr>
<tr>
<td>method is overkill for use-case</td>
<td>Following the method is described as being too burdensome in the specific trial being conducted, but not necessarily unbeneficial in other circumstances</td>
<td>When participants describe that they feel the benefit is not worth the effort in a specific context. When participants describe other more efficient ways to achieve the desired goal, but recognize the benefits which may be more important in other use cases</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>method restricting creativity</td>
<td>The method being restrictive in a way that prevent good and creative ideas of the analyst to work</td>
<td>When a participant describes that he or she has ideas how to do the analysis more efficient, but these strategies don’t work due to a too restrictive method framework</td>
</tr>
<tr>
<td>purpose of analysis</td>
<td>Understanding the purpose of the produced artifacts.</td>
<td>When participants describe their understanding of the purpose of the artifacts, or if it is implicitly apparent that a participant is lacking a proper understanding of the purpose for a domain model. Do not use when: When the topic is the purpose of the study.</td>
</tr>
<tr>
<td>tagging implicit concepts</td>
<td>The problem of specific concepts that are needed in the conceptual model are ubiquitous but never explicitly discussed.</td>
<td>When a participant describes it as hard to tag specific text sections when he or she has the feeling this concept is just everywhere. An example of this would be the employee</td>
</tr>
<tr>
<td>time constraints</td>
<td>The problem of not having enough time for the analysis</td>
<td>When a participant describes that they could not finish what they planned, or explicitly say they did not have enough time. Also if an assertion is made that with more time the results would be better. This could also appear in the context, that the method is too time consuming. Do not use when: When any of the subcategories apply</td>
</tr>
</tbody>
</table>

237
<table>
<thead>
<tr>
<th>Code</th>
<th>Full definition</th>
<th>When (not) to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>tooling problems</td>
<td>Challenges and problems that relate to the use of the respective tools.</td>
<td>When bugs are described, or aspects of a tool that are restrictive and impede the analysis. Do not use when: When any of the subcategories apply</td>
</tr>
<tr>
<td>&quot;little tricks&quot;</td>
<td>A term used by one participant to describe ways of working around perceived flaws in the tooling</td>
<td>When participants describe that they've used workarounds for something they found inconvenient</td>
</tr>
<tr>
<td>QDAcity bugs and feature requests</td>
<td>Bugs in the QDAcity tool that require a workaround or prevent intended use of the software, or features that would help the analysis process that are not present in the current version of the software</td>
<td>When participants describe that QDAcity did not work as expected due to a fault in the software, or the process could be more easily executed with a new feature that is not present. Do not use when: When wrong use of the software is the reason that the software did not react as expected by the participant</td>
</tr>
<tr>
<td>accidental changes</td>
<td>The tool allows for changes to be made unintentionally.</td>
<td>When a participant describes that they did changes in the tool, which were unintentionally, but not due to a bug but rather accidental usage of an intended and properly working feature. Do not use when: When the topic is about bugs in the software</td>
</tr>
<tr>
<td>exporting annotated interviews</td>
<td>The problem of exporting annotated interviews</td>
<td>When the participant describes problems related to exporting the annotated text files. This is mostly related to the group using Mendeley</td>
</tr>
</tbody>
</table>

238
<table>
<thead>
<tr>
<th>Code</th>
<th>Full definition</th>
<th>When (not) to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>seeing connections between different parts of the mind map</td>
<td>The problem of drawing connections between concepts in the mind map, that are not represented in the hierarchical structure.</td>
<td>When the representation of non-hierarchical relationships in the mind map is described as problematic</td>
</tr>
<tr>
<td>seven colors in mendeley</td>
<td>In mendeley there are only seven colors possible for annotation.</td>
<td>When the restriction to seven colors in mendeley is described as problematic. Do not use when: When any of the subcategories apply</td>
</tr>
<tr>
<td>frequency of changing reading purpose</td>
<td>The problem of having to change reading purposes frequently due to the limitation of seven colors</td>
<td>Use for mentions or references to frequency of changes to the reading purposes (both frequent and infrequent).</td>
</tr>
<tr>
<td>keeping track of reused colors</td>
<td>The problem that arises when reading purposes are changed and assigned a color that was already used in the same document</td>
<td>When participants describe confusion due to the reuse of colors, or describe problems with the workaround of having to write the name of the concept into the comment of the annotation</td>
</tr>
<tr>
<td>tool combination</td>
<td>Problems that arise due to the combination of different tools, especially the synchronization between different tools</td>
<td>When the combination of tools is described as problematic. This mostly applies to the group that needed to combine Mendeley as a reading tool with mindmeister and draw.io as modeling tools. Any synchronization issues between separate tools should be coded with this code</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>tool restricting method</td>
<td>The method can not be applied properly due to a limitation of the tooling</td>
<td>When participants describe that any limitation of the tool (bug or feature) prevents a proper application of the method. Do not use when: When the workaround &quot;Linking annotations manually&quot; is applicable. There exists a separate code with that name</td>
</tr>
<tr>
<td>application</td>
<td></td>
<td></td>
</tr>
<tr>
<td>understanding context</td>
<td>The problem of understanding the context of the domain.</td>
<td>When participants describe it as a challenge to understand exactly how a statement is meant in its intended context</td>
</tr>
<tr>
<td>using pre-defined structures</td>
<td>The challenge of working with the structures, templates and meta-models given by the respective method.</td>
<td>When participants describe the use of the code system language, the explicate problem template, or the coding paradigm. Do not use when: Any of the specific subcategories apply</td>
</tr>
<tr>
<td>coding paradigm</td>
<td>The five dimensions of the codign paradigm as described in the QDAcity-RE method: Action and strategies Consequences Causal condition Contextual condition Structural condition</td>
<td>When the challenge of using the five dimensions of the coding paradigm are discussed. Do not use when: The topic of the code system language, or the explicate problem template from the control group</td>
</tr>
<tr>
<td>completeness of coding paradigm</td>
<td>The coding paradigm is intended to cover all relevant dimensions to describe the domain</td>
<td>When the completeness of the coding paradigm is discussed, or specific dimensions are suggested to be missing</td>
</tr>
<tr>
<td>explicate problem template</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>explicate problem template confusing</td>
<td>Aspects of the explicate problem template are unclear.</td>
<td>Participants describe the template for the mind map confusing. Do not use when: When any of the subcategories apply</td>
</tr>
<tr>
<td>defining the problem</td>
<td>Problems with the dimension of set problem statement</td>
<td>When participants mention the template dimension of set problem statement as a point of confusion</td>
</tr>
<tr>
<td>explicate problem template is complete</td>
<td>Completeness of the Mindmeister template</td>
<td>When the completeness or incompleteness of the explicate problem template is discussed</td>
</tr>
<tr>
<td>node description adequacy</td>
<td>Descriptions of the nodes in the template being sufficient or not sufficient</td>
<td>When participants describe the node descriptions in Mindmeister as insufficient or as helpful and adequate</td>
</tr>
<tr>
<td>strategies</td>
<td>Strategies in the method application that participants employ</td>
<td>Any analysis strategy that participants came up with to address the encountered challenges. Do not use: When any of the sub categories apply</td>
</tr>
<tr>
<td>linking annotations manually</td>
<td>When the tools would not link annotations automatically for the group using strategic reading the solution is to link annotations manually with nodes in the mind map</td>
<td>When participants describe linking text segments in Mendeley manually to nodes in Mindmeister</td>
</tr>
<tr>
<td>annotating broad concepts first</td>
<td>Annotating broader concepts first</td>
<td>When a participant mentions that he or her annotated broad concepts first in order to refine the annotations to more fine grained concepts later. This is a contrary code to &quot;annotating detailed concepts first&quot;</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>annotating detailed concepts first</td>
<td>The practice of finding evidence for more detailed concepts first and then building them into categories later</td>
<td>When participants describe that they were looking for the more specific concepts first. This is the contrary code to &quot;annotating broad concepts first&quot;</td>
</tr>
<tr>
<td>attention to negative cases</td>
<td>The practice of actively looking for contradictions between new data, and already coded data and the current state of analysis</td>
<td>When participants describe they actively looked for contradictions</td>
</tr>
<tr>
<td>defining common terminology</td>
<td>The practice of defining terminology that best reflects the domain from the potentially distinct terminology used in different interviews</td>
<td>When participants describe how they chose to name a concepts when conflicting terminology was used in different interviews. Documentation of synonyms is another possible use of this code</td>
</tr>
<tr>
<td>emerging from evidence</td>
<td>Concepts or relationships emerge from evidences rather than from pre-defined notions</td>
<td>When a participant describes how concepts or relationships emerge from the data. In other words how the model is grounded in the data</td>
</tr>
<tr>
<td>focus on UML while annotating</td>
<td>While annotating the text the focus is on what might fit into the UML model</td>
<td>When participants describe that their decision what to tag in the text is driven by what they think would be a good addition to the UML model. This may often be in the context of having experience with UML modeling.</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>focus on solutions</td>
<td>Modeling primarily the solutions to a problem, rather than the problem itself</td>
<td>When participants describe, that they have a solutions focused approach in modeling the domain. The problems are less present in the resulting model than the solutions</td>
</tr>
<tr>
<td>focus on stakeholders</td>
<td>Human centered view on the HR development domain</td>
<td>When participants describe that they have a particular focus on identifying stakeholders and analyzing the domain with this focus</td>
</tr>
<tr>
<td>iteratively working</td>
<td>The participant reads the interview multiple times</td>
<td>When the analysis is not done in one reading of the interview but rather reading it multiple times, also if subsequent readings are performed in chunks, or selectively. Do not use when: One of the sub categories applies</td>
</tr>
<tr>
<td>focus on limited number of</td>
<td>During each iteration, the text is read with the specific focus of a limited number of concepts. Includes negative cases</td>
<td>When participants state that they purposefully limit the number of concepts they are focusing on while annotating at any given time. Or if it is stated that this is not the case</td>
</tr>
<tr>
<td>of concepts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>read through before analysis</td>
<td>Before the annotations are applied, the interview is read once from beginning to end</td>
<td>When a participant reads the interview at least once before applying any annotations</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>managing complexity of artifacts</td>
<td>Dealing with the complexity of any of the artifacts. Code system, mind map or UML model</td>
<td>When strategies are described to deal with the complexity, i.e. high number of nodes, lots of relationships, etc. Do not use when: When any of the subcategories apply</td>
</tr>
<tr>
<td>definition of concepts</td>
<td>Giving concepts a proper definition</td>
<td>When practices are described that have the goal to define the concepts used. For instance through a code book or code memo</td>
</tr>
<tr>
<td>naming conventions</td>
<td>Any situation where the name of a code/concept or other types of labels are named in a particular pattern</td>
<td>When participants describe that they used labeling as a way of organizing/structuring their findings or artifacts</td>
</tr>
<tr>
<td>naming of concepts</td>
<td>Strategies for naming concepts</td>
<td>When a participant describes how he or she thinks about naming concepts, or having strategies for naming concepts.</td>
</tr>
<tr>
<td>hierarchical structure helpful</td>
<td>Having a structure in the artifacts that help deal with its complexity</td>
<td>When participants describe the hierarchical structure of an artifact as helpful</td>
</tr>
<tr>
<td>only adding broad concepts to model</td>
<td>The strategy of only adding broad concepts to the model in order to keep the complexity of it manageable</td>
<td>When participants describe that they deliberately not include detailed concepts but only use broader ones to deal with the complexity of the artifact. This code also includes limiting relationships between the broader concepts</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>non–linked annotation</td>
<td>The practice of annotating text without linking it to a specific concept</td>
<td>When a participant tags text without linking it to a specific concept. This is done by using the yellow color in Mendeley. This is only possible for the control group using mendeley</td>
</tr>
<tr>
<td>not switching between tools often</td>
<td>Trying to minimize the number of times a switch between the different tools is necessary</td>
<td>When participants describe strategies for minimizing the switching between tools. This is only relevant for the strategic reading group. For example includes handwritten notes or sketches to avoid tool switching. Do not use when: When note taking etc is used as a general strategy not specifically for minimizing tool switching. There is a separate code under reading strategies called &quot;taking notes&quot; for this</td>
</tr>
<tr>
<td>re-reading annotations</td>
<td>The act of cross checking earlier instances of a code by looking up the already coded instances again</td>
<td>When participants describe how and when they cross-check current findings with earlier instances of tags for a specific concept. Do not use when: When participants go back to the previous interview as a whole, not just the annotations in qdacity or the mind map</td>
</tr>
<tr>
<td>reading strategies</td>
<td>Description of the reading process</td>
<td>The reading process for the interviews. Can also extend to the reading process of the method paper</td>
</tr>
</tbody>
</table>

245
<table>
<thead>
<tr>
<th>Code</th>
<th>Full definition</th>
<th>When (not) to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>ad-hoc summary</td>
<td>Writing an ad-hoc summary.</td>
<td>When participants state that they summarized the content of the reading independently of the prescribed analysis process.</td>
</tr>
<tr>
<td>looking up things online</td>
<td>The practice of looking information up online</td>
<td>When participants state that they used a search engine or a web platform to fill in knowledge gaps that are not covered by the interview or the method paper.</td>
</tr>
<tr>
<td>re-reading sections</td>
<td>The practice of reading sections of text again</td>
<td>When participants describe that they read sections multiple times. Do not use when: When participants describe that they read the interview once in the beginning. There is a separate code for that.</td>
</tr>
<tr>
<td>sticky notes</td>
<td>The practice of using sticky notes in the interview or paper as markers</td>
<td>When participants describe creating bookmarks in the text with sticky notes.</td>
</tr>
<tr>
<td>taking notes</td>
<td>The practice of taking notes while reading</td>
<td>When participants state that they noted some content they find relevant down. Do not use when: When participants wrote summaries of the reading material. There is a separate code for this.</td>
</tr>
<tr>
<td>using memos</td>
<td>The practice of using memos attached to elements of an artifact</td>
<td>When participants describe using code memos or something equivalent.</td>
</tr>
<tr>
<td>Code</td>
<td>Full definition</td>
<td>When (not) to use</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>terminology lookup</td>
<td>The process of looking up terminology of the reading content.</td>
<td>When participants describe that after or during the reading they needed to look up certain terms because they were unfamiliar with them, or unsure how they were used.</td>
</tr>
<tr>
<td>use of color</td>
<td>The use of color for concepts / nodes</td>
<td>Any mention of how color is used or not used for codes or nodes in the mind map.</td>
</tr>
<tr>
<td>using meta codes</td>
<td>Using codes as labels which have their meaning in the analysis process rather than in the description of the domain.</td>
<td>When participants state that they used meta codes as bookmarks to find things that, for instance, they didn’t understand, or find easily later, but that do not describe the domain.</td>
</tr>
</tbody>
</table>