INAUGURAL-DISSERTATION

zur

Erlangung der Doktorwürde

der

Naturwissenschaftlich-Mathematischen Gesamtfakultät

der

Ruprecht-Karls-Universität

Heidelberg

vorgelegt von

Dipl.-Inform.(FH) Christian Kücherer

aus Heidelberg

Tag der mündlichen Prüfung: ....

Domain-specific Adaptation of Requirements Engineering Methods

Gutachterin:	Prof. Dr. Barbara Paech
Betreuer:	Prof. Dr. Alfred Winter

Abstract

[Context and motivation] Requirements are fundamental for the development of software-based information systems (ISs). Stakeholder needs for such ISs are documented as requirements following a requirements engineering (RE) method. Requirements are specific to the *application domain* for which ISs are developed and in which they are used. A *system domain* is represented by ISs that share a minimal set of common requirements to solve similar domain independent problems. Both domain-specific aspects need to be considered explicitly during the specification of ISs. Generic RE methods can be used in different domains, but do not consider explicitly domain-specific details. A solution to this problem is the domain-specific adaptation of RE methods. Domain-specific modeling languages (DSMLs) allow conceptual modeling in specific system domains. Domain ontologies provide formalized domain knowledge of an application domain. **[Objectives]** The objective of this thesis is to investigate, *through the example of the task-oriented RE conceptual framework TORE*, (1) how a generic RE method can be adapted to consider system domain-specifics with the use of a DSML, and (2) how a generic RE method can be adapted to use an application domain ontology. For the system domain adaptation, we use a personal decision support system (PDSS). The PDSS supports a Chief Information Officer (CIO) in decision-making with tasks in information management (IM). For the adaptation to the application domain, we use IM in hospitals represented by the semantic network of information management in hospitals (SNIK) domain ontology. **[Contributions:]** The results of this investigation consist of two method adaptations: first, the system domain-specific DsTORE, and second, the application domain-specific TOREOnto. The contributions of the system domain-specific adaptation DsTORE are fourfold. First, an *as-is domain study* provides details about the information management department of a specific hospital in order to understand the organizational context for the PDSS that will be employed. Second, an exploratory case study shows the extent of task-oriented requirements engineering (TORE) to support the RE specification of a PDSS. Third, the design of DsTORE provides the system domain-specific adaptation of TORE to support the specification of PDSS. Fourth, a case study documents the evaluation of DsTORE. The application domain-specific adaptation TOREOnto consists of three contributions. First, a literature review provides the state of the art regarding the use of domain ontologies in RE, describing nine different usage scenarios of domain ontologies to improve the quality of requirements. Second, the design of TOREOnto provides the application domain-specific adaptation to support the improvement of requirements quality. Third, a case study shows the retrospective evaluation of TOREOnto with RE artifacts created in this thesis. **[Methods:]** The overall research method of this thesis is Design Science according to Wieringa. The problem investigation of domain ontology usage in RE is based on a systematic literature review by Kitchenham and Charters.

Zusammenfassung

[Kontext und Motivation] Anforderungen sind von grundlegender Bedeutung für die Entwicklung softwarebasierter Informationssysteme (ISs) und beschreiben die Bedürfnisse von Interessensgruppen. Anforderungen von ISs werden mit Hilfe von Methoden des Requirements Engineerings (RE) dokumentiert und sind spezifisch für die Anwendungsdomäne, in denen das IS genutzt wird. ISs, die ähnliche Probleme in unterschiedlichen Anwendungsdomänen lösen, haben eine Menge gleichartiger Anforderungen und repräsentieren eine Systemdomäne. Bei der Spezifikation von ISs müssen beide domänenspezifischen Aspekte explizit berücksichtigt werden. Generische RE-Methoden werden in unterschiedlichen Domänen eingesetzt, berücksichtigen allerdings domänenspezifische Details nicht explizit. Eine Lösung für dieses Problem ist die domänenspezifische Anpassung von RE-Methoden. Domänenspezifische Modellierungssprachen (DSMLs) ermöglichen die konzeptuelle Modellierung von ISs einer Systemdomäne. Domänen-Ontologien stellen formalisiertes Wissen über eine Anwendungsdomäne bereit. **[Zielsetzung]** Das Ziel dieser Arbeit ist die Erforschung der domänenspezifischen Anpassung von RE-Methoden am Beispiel der Methode des aufgabenorientierten RE (TORE). Wie kann eine generische RE-Methode angepasst werden, um (1) eine DSML zu nutzen, die Domänen-Besonderheiten des IS berücksichtigt? (2) Ontologien der Anwendungsdomäne zu nutzen? Für die Anpassung an die Systemdomäne nutzen wir ein persönliches Entscheidungsuntersützungssysteme (PDSS), das einen/eine IT-Leiter/in bei Entscheidungsprozessen im Informationsmanagement (IM) unterstützt. Für die Anpassung an die Anwendungsdomäne verwenden wir das IM im Krankenhaus, formal beschrieben durch die SNIK Domänen-Ontologie. **[Beiträge:]** Das Ergebnis dieser Forschung sind zwei Arten von domänenspezifischen Methodenanpassungen: Erstens, die Systemdomänen-Anpassung DsTORE und zweitens die Anwendungsdomänen-Anpassung TOREOnto. Es bestehen vier Beiträge für DsTORE: Erstens beschreibt eine Domänen-Studie den Ist-Zustand der IM-Abteilung eines konkreten Krankenhauses, die den organisatorischen Zusammenhang des einzusetzenden PDSS beleuchtet. Zweitens zeigt eine Fallstudie den Grad der Eignung von TORE für die RE-Spezifikation eines PDSS. Drittens stellt DsTORE die Anpassung von TORE an eine Systemdomäne dar, um die Spezifikation von PDSS zu unterstützen. Viertens wird DsTORE anhand einer Fallstudie evaluiert. Es bestehen drei Beiträge für TOREOnto: Erstens zeigt eine Literaturstudie den Überblick über den aktuellen Stand der Nutzung von Domänen-Ontologien im RE. Dabei beschreiben neun verschiedene Nutzungsszenarien, wie Domänen-Ontolgien genutzt werden um die Qualität von Anforderungen zu verbessern. Zweitens stellt TOREOnto die Anpassung von TORE an eine Anwendungsdomäne dar, um die Nutzung von Domänenontologien zur Verbesserung der Anforderungsqualität zu ermöglichen. Drittens zeigt eine Fallstudie die retrospektive Evaluation von TOREOnto anhand von bestehenden Anforderungsartefakten. **[Methoden:]** Die übergreifende Forschungsmethode dieser Arbeit ist Design Science nach Wieringa. Die Problemanalyse der Nutzung von Domänen-Ontolgien im RE basiert auf einem systematischen Literatur-Überblick nach Kitchenham und Charters.

Acknowledgements

This thesis would not have been possible without the direct or indirect support of many people, to whom this section is dedicated.

I owe my deepest gratitude and appreciation to my supervisor Prof. Dr. Barbara Paech. Without her, this work would not have been possible. Her support and continuous feedback to many intermediate results of this research and her belief in me to complete this thesis has been exceptional. I am deeply grateful for the opportunity to be a part of her research group and to participate in the SNIK research project. I also thank the Institute for Computer Science at the Faculty of Mathematics and Computer Science of Heidelberg University for accepting me as doctoral student and the opportunity to research in such a great environment.

Also, I thank the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) for the grant of the project SNIK: Semantic Network of Information Management in Hospitals, Grant no. 1605/7-1 and 1387/8-1 and their financial support of this thesis during the last three years. Within this research project, my project colleagues from IMISE at the University of Leipzig always supported and gave me constructive feedback on my work. Especially, I thank Prof. Dr. Alfred Winter for acting as a second supervisor, and his research group members Franziska Jahn, Dr. Birgit Schneider, Konrad Höffner, Michael Schaaf, and Sebastian Stäubert. Special thanks to Stefan Smers, the CIO with whom I could apply the DsTORE requirement engineering method, for his valuable contributions, time, cooperation and patience.

Furthermore, I thank my colleagues at the Software Engineering Group Anja Kleebaum, Marcus Seiler, Paul Hübner, and Tom-Michael Hesse for their discussions and feedback on my research work. I thank Leonore Dietrich for several climbing and swimming sessions during the last few years. I also thank Jonathan Griffiths for proof-reading this thesis and giving many language advice. Thanks too to Doris Keidel-Müller, Dr. Willi Springer and Dorothea Heukäufer for their excellent administrative support! Further thanks to the doctoral alumni Dr. Thorsten Merten, Dr. Ulrike Aebelein and Dr. Rumy Proynova for providing wonderful pieces of advice and insights. For the motivation to start and complete this thesis, I thank Christine Matthes and Paul Beetz, Prof. Dr. Ulrike Kutscha, Thereseia and Stefan Eichner, Dr. Uwe Gühl, and Prof. Dr. Andreas Schrader.

I thank my students, who supported my work and this thesis with their ideas, prototypes, discussions, and their B.Sc. and M.Sc.-theses: Manuel Jung, Benedikt Tröster, Tim Bittersohl, Leon Radeck, and in particular my student assistants Anatoli Zeiser and Ewald Rode. I also thank our ERASMUS student Duygu Yagci Dogan from Turkey for her support with the literature review of DSS-RE methods and her work on the characteristics of search engines.

A big thank you additionally goes to Silke Lampson ♡ for her love, relief, and patience with all the many things that I am interested in. Especially, I thank you Silke, for taking over so many tasks, duties and errands to allow me free time for thesis writing. A special thanks to my parents, Dieter and Inge Kücherer, for all their love, effort, advice, time, patience and support over many years. Thanks to my nephew Jochen Clormann who is about to finish his studies in computer science! I am proud of you ☺. I also thank Anne Buschmann for being such a good friend and the chance to spend recreational time in Le Lavandou, and the lovely and wild dog Bonny as our weekend dog to share. I would like to thank all my other friends who supported me with discussions and the enjoyable time spent together during the last three years.

Contents

Part I

Preliminaries

1 Introduction

This chapter introduces the subject of this thesis. Section 1.1 explains the motivation of this research into the domain-specific adaptation of requirements engineering (RE) methods, and gives a brief overview of the examples used to demonstrate the adaptations. Next, the research goal and the design problems are explained in Section 1.2. The thesis' contributions are presented in Section 1.3, followed by a brief description of the research methodology in Section 1.4. Some intermediate results of this thesis' content have been published through peer-reviewed conferences and workshops, which are listed in Section 1.5. Finally, the outline of the thesis is described in Section 1.6.

1.1 Motivation

The specification of requirements is essential for the successful development of software-based information systems (ISs). *Individual requirements (InRs)* describe stakeholders' expectations of an IS. False requirements or requirements affected by quality issues lead to false systems and additional effort during system development [EK08, Boe01, Poh16].

ISs and their requirements are specific to the *application domain* for which they have been developed and in which they are used [Bjø06]. An application domain is anything to which computing may be applied [Bjø06]. More specifically, Bjørner defines an application domain

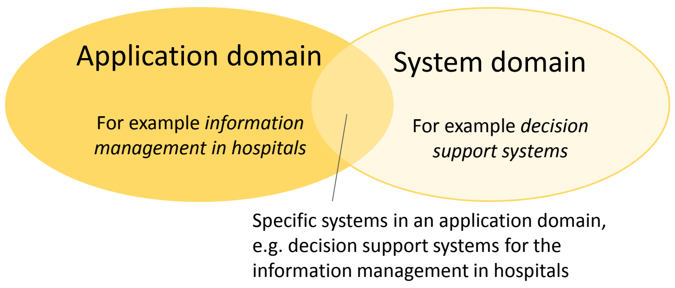

Application domain
For example *information management in hospitals*
System domain
For example *decision support systems*
Specific systems in an application domain, e.g. decision support systems for the information management in hospitals

Figure 1.1: Application domain and system domain

as the application of software within business or industry sectors, such as transportation, the financial services, healthcare or the machining manufacturing sector [Bjø06]. Systems of a similar type are intended to solve similar problems in a similar context with similar technology. Such systems share a minimal set of common requirements, a common terminology, and a common functionality, and represent a *system domain*. Two examples of such systems are decision support systems (DSSs), and enterprise resource planning systems (ERPs). DSSs support organizational decision-making tasks by implementing requirements that relate to the access and aggregation of data, and the formal modeling and representation of decision tasks. DSSs can be used in various application domains, such as business management, medicine, and geography. By contrast, ERPs address requirements that relate to the planning of work processes and interoperability with suppliers and delivery chains. All ISs of a *system domain* are finally deployed and used in a specific *application domain*. For example, a decision support system used to support *project planning* in the information management (IM) department of a hospital is a concrete instance of the system domain DSSs in the application domain IM in hospitals. Figure 1.1 shows the relation of an application domain to a system domain.

Examples of application domains and system domains

application domains: information management in hospitals, financial services, transport and logistic, and the publishing sector.

system domains: decision support systems, enterprise resource planning systems, content management systems, ambient assisted living systems.

Adam et al. under a single *domain* both these aspects, namely system domains and application domains [ADEG09]. As examples of the system domain, the authors investigate social network software and ambient assisted living, and as examples for the application domain, they investigate the office domain, and the transportation sector. Both domain-specific aspects need to be considered explicitly in requirements specifications [ADEG09]. Stakeholders are application domain experts, and they use a terminology specific to their application domain without even mentioning its meaning explicitly [Som09]. By contrast, software engineers are often not domain experts and do not possess application domain-specific knowledge. For this reason, they may not fully understand the characteristics of the application domain in which the IS operates, and often they cannot identify missing or conflicting requirements [Som09]. The *software engineering body of knowledge* [BF14] emphasizes the need for software engineers to use knowledge about the application domain. Domain knowledge helps to detail the communication, specification, and documentation of requirements during all RE activities [Eva04]. Elicited InR must be set in the context of domain knowledge. Typical RE methods are generic and can be applied across various domains. But this implies the problem that current RE methods do not take advantage of using domain-specific details explicitly.

One solution to this problem is the domain-specific adaptation of RE methods. The question is how such domain-specific adaptations of RE methods can be carried out. There are two options for considering domain-specific details. Domain-specific languages (DSLs) are languages tailored to a specific system domain or application domain, depending on the focus of the DSL. DSLs cover not only the well known *domain-specific programming languages*, but also *domain-specific modeling languages (DSMLs)* for conceptual modeling purposes. The creation of a DSL is always motivated by a single purpose and thus an individual adaptation [MHS05]. Domain ontologies provide formalized domain knowledge of an application domain [SBF98]. As ontologies have become widely used in software engineering, their use and impact in the RE discipline has also increased [DVB+15]. Many different approaches use domain knowledge in the form of DOs in order to improve requirements quality. Adapted RE methods that use ontologies can be instantiated multiple times across different contexts. In this work, we use a DSML for the system domain adaptation and an ontology for the application domain adaptation.

The **main goal** of this thesis is to investigate two questions *through the example of task-oriented requirements engineering (TORE)*: (1) how a generic RE-method can be adapted to consider system domain-specifics through the use of a DSML; and (2) how a generic RE-method can be adapted to use application domain ontologies. The task-oriented RE conceptual framework TORE emphasizes stakeholders' tasks, and proposes a set of 18 decisions that have to be explicitly or implicitly made for the specification of requirements for an IS [PK04, ADEG09]. This framework uses four different levels of abstraction to refine decisions about the to-be system, from a high level of abstraction to more specialized lower-level requirement decisions. Thus, TORE provides a comprehensive guidance during the entire requirements process. However, it has not been studied so far, how TORE can be adapted to a system domain and DOs are used on different levels of system specification to improve requirements quality. Such knowledge is important for both, requirements engineers who are interested in improving requirements quality, and method designers, who aim at the enhancement or development of new RE-specification techniques. The results of this investigation are two method adaptations: first, the system domain-specific Decision specific TORE (DsTORE), and second, the application domain-specific task-oriented requirements engineering with ontologies (TOREOnto).

1.2 Research Goal and Design Problems

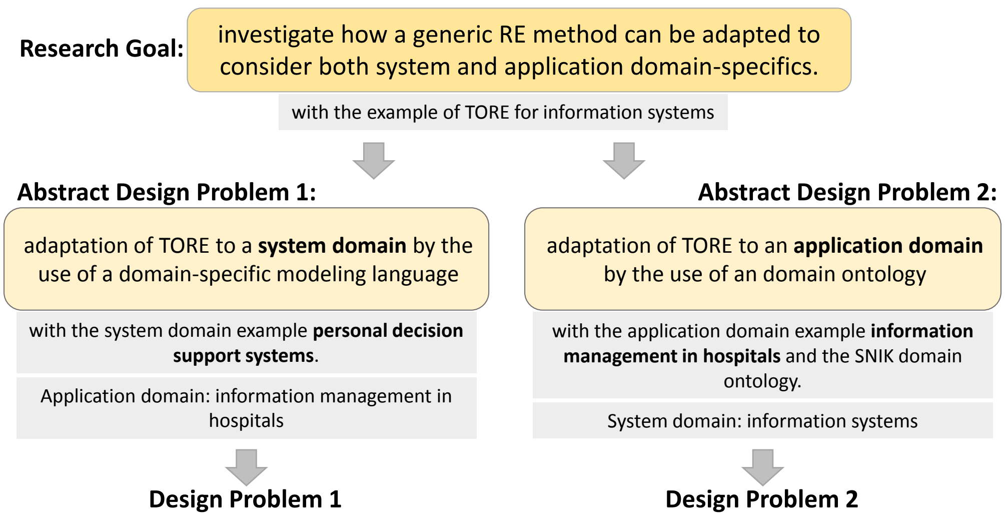

Research Goal:
investigate how a generic RE method can be adapted to consider both system and application domain-specifics.
with the example of TORE for information systems
Abstract Design Problem 1:
adaptation of TORE to a **system domain** by the use of a domain-specific modeling language
with the system domain example **personal decision support systems**.
Application domain: information management in hospitals
Design Problem 1
Abstract Design Problem 2:
adaptation of TORE to an **application domain** by the use of an domain ontology
with the application domain example **information management in hospitals** and the SNIK domain ontology.
System domain: information systems
Design Problem 2

Figure 1.2: The research goal and the two design problems with their examples

The main research goal of this thesis as explained in the previous section is visualized in Figure 1.2. The research goal is concretized by the application of TORE as an example of a generic RE method focusing on ISs. Both the system domain-specific and the application domain-specific adaptation are derived as *abstract design problems*. The abstract design problems 1 and 2 are answered by the application of TORE to a concrete system domain and application domain. During this application, the research experiences are collected with the main purpose of generalizing the findings and answering the question set in the main research goal for any RE method. The design problems are documented on the basis of Wieringa's template for design problems [Wie14], as introduced in subsection 1.4.1.

The example of the concrete system domain is the class of personal decision support systems (PDSSs), and the example of the application domain is IM in hospitals. The context of both examples is IM in hospitals, as has been provided by the semantic network of information management in hospitals (SNIK) research project, as will be introduced in detail in Section 2.5. The SNIK domain ontology describes the application domain of IM in hospitals. IM comprises all activities for the management of ISs in order to deliver information technology (IT) services. For several decisions in IM, the Chief Information Officer (CIO) who is responsible uses a DSS. The system domain of DSSs comprises ISs with various styles and forms with the intention of supporting decision-making.

The example of the abstract design problem 1 is the system domain of PDSSs in IM in healthcare. We can refine the abstract design problem 1 together with this example in order to define the following design problem 1. The research questions (RQs) to this design problem 1 are presented and explained in Section 3.2.

Design problem 1

Improve TORE **by** designing DsTORE **that satisfies** the use of a decision-specific domain-specific modeling language, **in order to** support requirements engineers in providing a good system domain-specific software requirements specification.

The example of the abstract design problem 2 is the application domain of IM in hospitals with the SNIK domain ontology, as will be introduced in more detail in subsection 2.5.2. We can refine the abstract design problem 2 together with this example in order to define the following design problem 2. The RQs to this design problem 2 are presented and explained in Section 8.1.

Design problem 2

Improve TORE **by** designing TOREOnto **that satisfies** the use of application domain ontologies, **in order to** support requirements engineers in improving requirements quality.

1.3 Contributions

This thesis will make the following contributions:

Contributions for design problem 1: The first contribution is an *as-is domain study* in order to understand in detail the IM department of a specific hospital and the organizational context of the CIO as the department's head. The second contribution is an exploratory case study investigating the extent to which TORE supports the RE specification of a PDSS. The third contribution is the decision specific adaptation DsTORE to support the specification of PDSSs. The fourth contribution is a case study in which DsTORE is evaluated.

Contributions for design problem 2: The fifth contribution is a literature review according to Kitchenham and Charters [KC07], that shows the state of the art regarding the use of domain ontologies in RE. As a result of this literature review, nine different usage scenarios of domain ontologies for improving requirements quality are synthesized as reusable *domain ontology usage patterns*. The sixth contribution is the application domain specific adaptation TOREOnto that supports the improvement of requirements quality with domain knowledge. The seventh contribution is a case study for evaluating TOREOnto.

1.4 Research Methodology

This chapter introduces in brief the research methodology that is used throughout this thesis. The first subsection 1.4.1 introduces *design science* as described by Roel J. Wieringa. Design Science is about the investigation, design and improvement of real-world artifacts in a given context. Furthermore, design science defines the primary structure of this thesis, which is separated into *problem investigation*, *treatment design*, and *treatment validation*. Part II and Part III follow the structure of the these design science activities. The second subsection 1.4.2 describes the Goal Question Metric (GQM) principle, which provides a systematic way of connecting research goals of an investigation with questions and the necessary metrics that define the scope of the raw information to obtain.

1.4.1 Design Science

Wieringa describes the idea of *Design Science Methodology* in the context of software engineering (SE) [Wie14, WM12], extending the earlier work of Hevner et al. [HMPR04]. Design Science is the *design* and *investigation* of artifacts in a given context. An **artifact** is anything created by someone for a practical purpose. Artifacts were designed to interact with a **problem context** in order to improve something in that given context. An **artifact's context** can be anything, such as the design, development, and use of ISs, where humans are part of the artifact's context.

Examples of artifacts and problem context [Wie14]

artifacts: algorithms, methods, notations, techniques, conceptual frameworks.
problem context: people, organizations, business processes, services, techniques, desires, goals.

A typical design science project consists of an object of study, such as the artifact, and two activities that are executed iteratively: (1) the **design activity**, which is intended to improve something for stakeholders by addressing a *design problem* that requires a change in the real world; and (2) the **investigative activity**, which is intended to empirically investigate an artifact in its given context. Design science projects follow one or more defined design problems. A **design problem** shall be formulated in accordance with Wieringa's template:

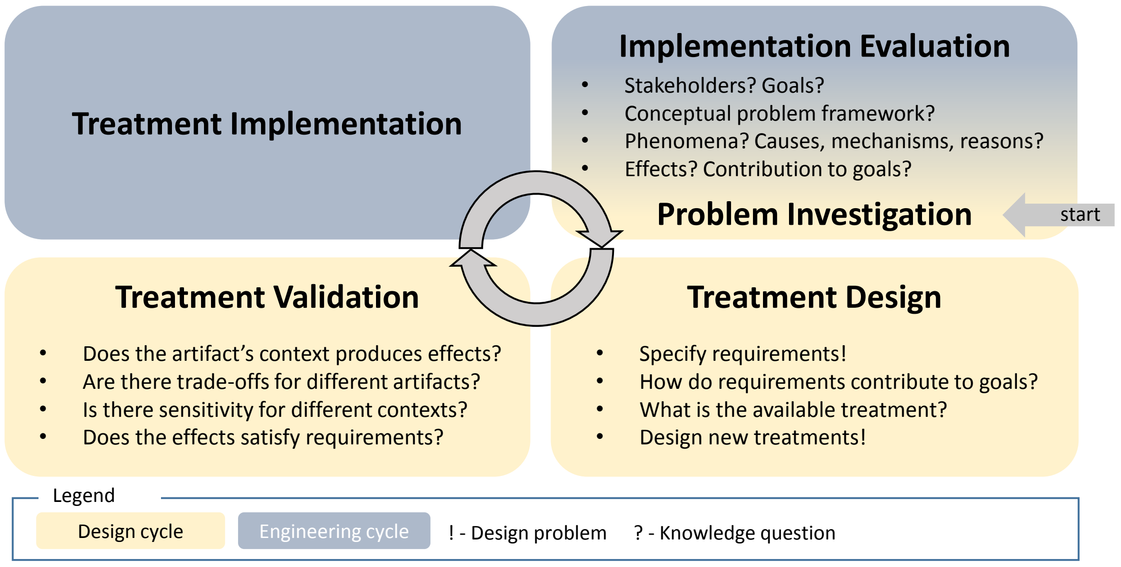

Treatment Implementation
Implementation Evaluation
Stakeholders? Goals?
Conceptual problem framework?
Phenomena? Causes, mechanisms, reasons?
Effects? Contribution to goals?
Problem Investigation
start
Treatment Validation
Does the artifact's context produces effects?
Are there trade-offs for different artifacts?
Is there sensitivity for different contexts?
Does the effects satisfy requirements?
Treatment Design
Specify requirements!
How do requirements contribute to goals?
What is the available treatment?
Design new treatments!
Legend
Design cycle
Engineering cycle
! - Design problem
? - Knowledge question

Figure 1.3: The engineering cycle according to Wieringa [Wie14]

Wieringa's design problem template

"**Improve** <a problem context> **by** <(re)designing an artifact> **that satisfies** <some requirement> **in order to** <help stakeholders to achieve some goals>" [Wie14].

The **treatment** is the interaction between the artifact and the problem context. The research which follows the principles of design science designs not just an artifact, but additionally an interaction between the artifact and the problem context, which is intended to treat the problem [Wie14]. Example: "A medicine (artifact) for the human body (problem context) is supposed to treat a disease (real-world problem)" [Wie14].

The **engineering cycle** is a rational problem-solving process and is structured as shown in Figure 1.3. It consists of the following five tasks: (1) The *problem investigation* questions what are the phenomena that must be improved and why. (2) In the *treatment design* task, a treated artifact that improves something is designed. (3) In the *treatment validation* task, this design is evaluated. (4) If and when the design is found to be appropriate, the treatment is applied to the original problem context in the task called *treatment implementation*. (5) Finally, the success of the treatment is evaluated in the task called *implementation evaluation*. When the treatment does not fulfill the defined design problems, a new iteration of the engineering cycle is necessary. The **design cycle** is a subset of the engineering cycle and consists of the first three tasks *problem investigation*, *treatment design*, and *treatment validation*. The execution of the design cycle tasks results in a validated treatment. After each design cycle, the improved design needs to be transferred during the engineering cycle to the real world.

Artifacts are designed and the design is documented as specification. In the context of design science, the term *specification* refers to the documentation of a designer's decisions about what to do. *Implementation* is defined as "the application of the treatment to the original problem context" [Wie14]. The *validation* of a treatment justifies its contribution to the expected design problem. The *evaluation* refers to the investigation of a treatment as applied in the real world. The evaluation requires the implementation of the artifact to be completed.

1.4.2 Goal Question Metric

Goal Question Metric (GQM) is a systematic way to create a measurement model on three abstraction levels, that were introduced by Basili et al. [BCR94]. The GQM approach is based on the assumption that an effective measurement requires defined goals, needs to be applied to the products, processes or resources in focus, and needs to be interpreted on the basis of the given context. This means that measurements need to be defined in top-down fashion, that means from goals down to obtained data. The result of the GQM is a measurement system targeting a particular set of issues and the interpretation of the measurement data.

The measurement model has three abstraction levels. On the conceptual level, a set of *goals* are defined that describe the aim of an investigation. A goal is defined for the objects of study, such as products, processes or resources. On the operational level, a set of *questions* are defined to characterize the assessment of the specific goal. These questions determine the selected quality of an object of study. A set of *metrics* are also defined on a quantitative level to answer each question in a measurable way. Figure 1.4 shows an example of a GQM. Assume that the *goal* of an investigation is to determine how DSSs support decision-making. This goal is refined with the help of two *questions*, (i) which decisions are supported; and (ii) by which features they are supported. The *metrics* used to answer question (i) are the level of structure of the decisions, their input and output and the type of decisions. The metric

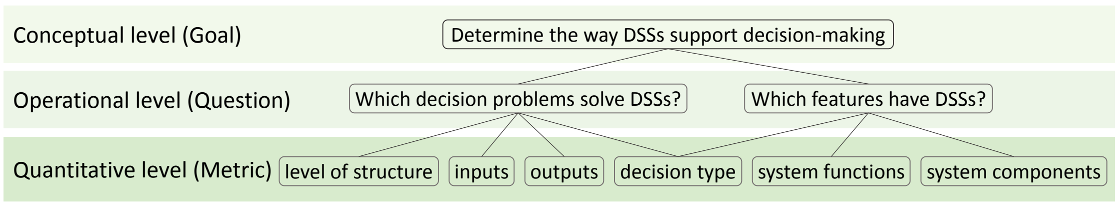

Conceptual level (Goal)
Determine the way DSSs support decision-making
Operational level (Question)
Which decision problems solve DSSs?
Which features have DSSs?
Quantitative level (Metric)
level of structure
inputs
outputs
decision type
system functions
system components

Figure 1.4: An example of GQM showing the three abstraction levels

decision type is also used to answer question (ii), together with the further metrics system functions and the system components used in an DSS.

1.5 Previous Publications

Parts of this thesis have been published as scientific publications. The following Table 1.1 provides an overview of these publications in chronological order and the related chapters.

Table 1.1: Publications that have previously emerged from this thesis

No.	Publication	Chapter
1.	Kücherer C, Jung M, Jahn F, Schaaf M, Tahar K, Paech B, Winter A: System Analysis of Information Management. In: INFORMATIK 2015. Bonn (Germany), Gesellschaft für Informatik; 2015 [KJJ+15].	4, 5
2.	Kücherer C, Liebe JD, Schaaf M, Thye J, Paech B, Winter A, Jahn F: Status Quo of Information Management in Hospitals - Results of an online survey, In: INFORMATIK 2016 [KLS+16]	4
3.	Kücherer, C, Paech, B: A Task-oriented Requirements Engineering Method for Personal Decision Support Systems - A Case Study. 19th International Conference on Enterprise Information Systems (ICEIS 2017), Porto (Portugal), April 26-29, 2017, SCITEPRESS 2017 [KP17]	5, 6, 7
4.	Kücherer C.: Use of Domain Ontologies to Improve Requirements Quality. In: Joint Proceedings of REFSQ-2017 Workshops, Doctoral Symposium, Research Method Track, and Poster Track co-located with the 23nd International Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2017). Essen (Germany), CEUR-WS.org; 2017 [Küc17]	9, 10
5.	Kücherer, C, Paech, B, Höffner, K: Usage Strategies of Domain Ontologies to Improve Requirements Quality - An extension of an existing Systematic Literature Review. **Status: submitted to Springer RE Journal**	9
6.	Kücherer, C, Paech, B: Task-oriented Requirements Engineering for Personal Decision Support Systems. Revised Selected Papers of ICEIS 2017, Springer Verlag **Status: accepted, to appear**	5, 6, 7

1.6 Thesis Outline

This thesis is structured into four parts and twelve chapters. An overview of the structure and the accompanying research questions and results is shown in Figure 1.5. The RQs are shown here for the purposes of providing an overview only, and they are explained in detail in the corresponding chapters. Part II and Part III are structured on the basis of design science.

Part I contains the preliminaries for this thesis. Chapter 2 presents the fundamentals of requirements engineering and requirements quality, task-oriented requirements engineering, ontologies, information management, and the semantic network of information management in hospitals, which is the case used throughout this thesis.

Part II provides a brief introduction to decisions and their support through ISs, along with a refinement of design problem 1 in Chapter 3. The domain study in Chapter 4 provides a pre-assessment for the problem investigation, which is then presented in Chapter 5. The decision specific adaptation DsTORE is presented as treatment design in Chapter 6. The treatment validation of DsTORE is presented as a case study in Chapter 7.

Part III provides an introduction, with the refinement of design problem 2, in Chapter 8, and a more detailed discussion of ontology types. The problem investigation in Chapter 9 presents the state of the art of domain ontologies usage for improving requirements quality, on the basis of a literature review and the description of principle domain ontology usage patterns. The TOREOnto adaptation for using domain knowledge as ontologies in TORE is presented in Chapter 10. The treatment validation of TOREOnto is presented in Chapter 11 on the basis of a retrospective evaluation of the obtained RE artifacts in Part III.

Part IV provides the answer to the main research goal in Chapter 12. Finally, this thesis is concluded by a summary and some ideas and directions for future work in Chapter 13.

Additional material, such as interview questions and questionnaires, is presented in Appendix A. An overview of TORE artifact types is presented in Appendix B. The metamodel of the semantic network of information management in hospitals ontology is shown in Appendix C. Further artifacts that complete the application of the DsTORE method are presented in Appendix D. The abbreviations used in this thesis and their long form are given in Appendix F. A glossary with explanations of important terms used throughout this thesis can be found in Appendix G.

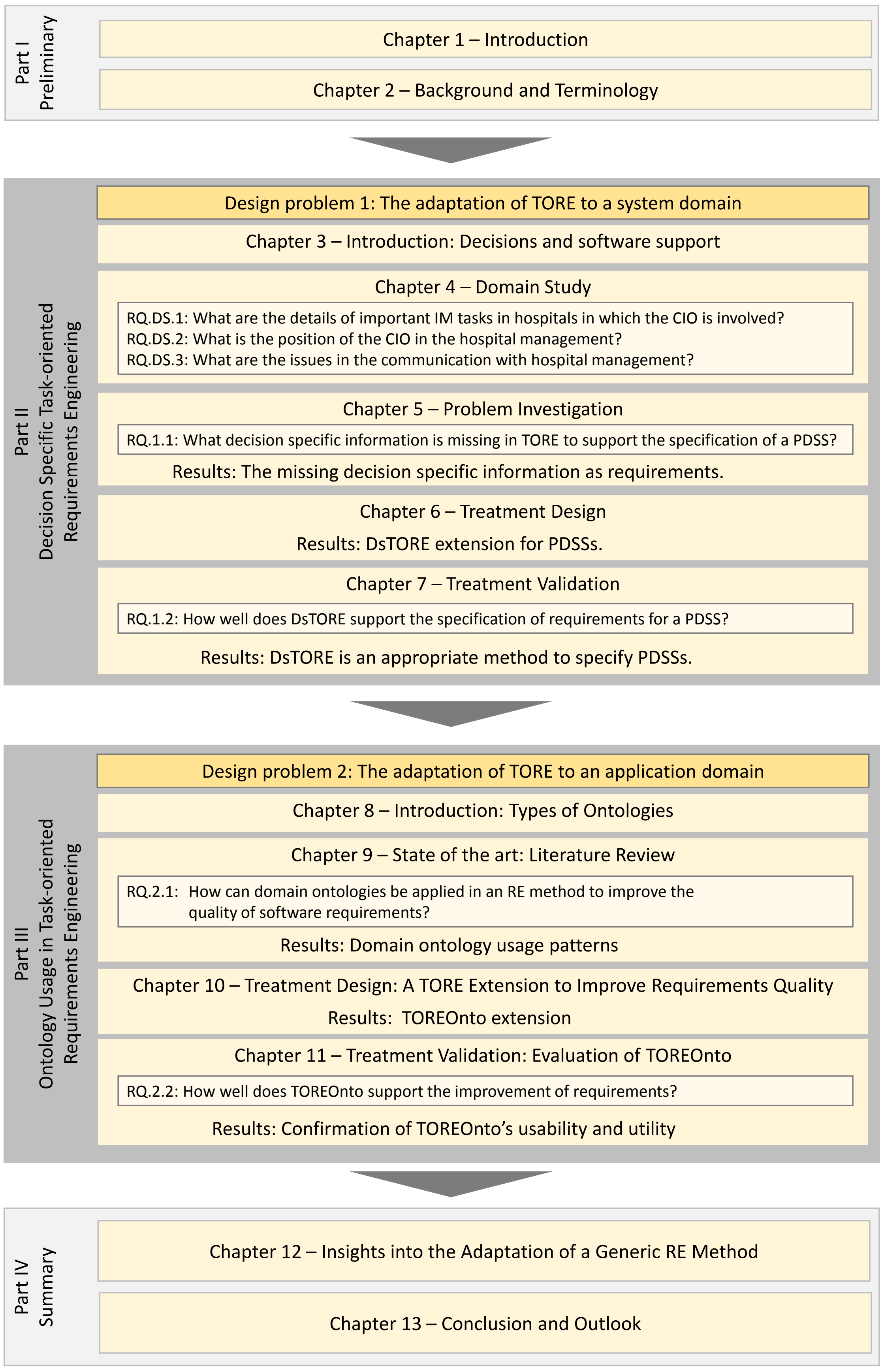

Part I
Preliminary
Chapter 1 – Introduction
Chapter 2 – Background and Terminology
Part II
Decision Specific Task-oriented
Requirements Engineering
Design problem 1: The adaptation of TORE to a system domain
Chapter 3 – Introduction: Decisions and software support
Chapter 4 – Domain Study
RQ.DS.1: What are the details of important IM tasks in hospitals in which the CIO is involved?
RQ.DS.2: What is the position of the CIO in the hospital management?
RQ.DS.3: What are the issues in the communication with hospital management?
Chapter 5 – Problem Investigation
RQ.1.1: What decision specific information is missing in TORE to support the specification of a PDSS?
Results: The missing decision specific information as requirements.
Chapter 6 – Treatment Design
Results: DsTORE extension for PDSSs.
Chapter 7 – Treatment Validation
RQ.1.2: How well does DsTORE support the specification of requirements for a PDSS?
Results: DsTORE is an appropriate method to specify PDSSs.
Part III
Ontology Usage in Task-oriented
Requirements Engineering
Design problem 2: The adaptation of TORE to an application domain
Chapter 8 – Introduction: Types of Ontologies
Chapter 9 – State of the art: Literature Review
RQ.2.1: How can domain ontologies be applied in an RE method to improve the quality of software requirements?
Results: Domain ontology usage patterns
Chapter 10 – Treatment Design: A TORE Extension to Improve Requirements Quality
Results: TOREOnto extension
Chapter 11 – Treatment Validation: Evaluation of TOREOnto
RQ.2.2: How well does TOREOnto support the improvement of requirements?
Results: Confirmation of TOREOnto's usability and utility
Part IV
Summary
Chapter 12 – Insights into the Adaptation of a Generic RE Method
Chapter 13 – Conclusion and Outlook

Figure 1.5: Structure and contribution of the thesis

2 Background and Terminology

This chapter provides definitions of the most important terms and concepts used throughout the thesis. Section 2.1 explains the foundations of requirements engineering and requirements quality. Section 2.2 gives a brief summary of the generic task-oriented requirements engineering method TORE in order to specify the requirements for information system (IS). Section 2.3 explains further details about ontologies and the related activities. Section 2.4 introduces the basics of information management, and in particular the application domain *IM in hospitals*. Finally, the research project semantic network of information management in hospitals (SNIK) is introduced in Section 2.5.

2.1 Requirements Engineering

This section covers the foundations for RE and the unified modeling language (UML) as a general-purpose modeling language.

2.1.1 Definitions

Information systems (IS) are software-based systems whose primary purpose is to manage and provide access to a database of information [Som09] and to fulfill commercial or organizational requirements. ISs comprise of hardware, system software (such as web servers, application servers or databases) and the application itself. ISs have an inherent system architecture which structures the system into different architectural layers and organizes its internal components.

Requirements Engineering (RE) is an interdisciplinary function with the goal of establishing and maintaining the requirements that need to be met by a system, a piece of software or a service [ISO11]. The outcome of RE is an agreed understanding between stakeholders that can be validated against real-world needs, and that provides the basis for the verification of system designs and the resulting solution [ISO11]. *Stakeholders* are either natural or legal persons, or organizations who are able to influence a certain project and impact on the project result [Poh16]. Stakeholders are always specific to a project and consist typically of at least users and acquirers, often complemented by regulatory authorities and laws [ISO11].

RE comprises of four main activities, namely elicitation, documentation, validation and negotiation, as well as the management of requirements [IRE15]. Requirements are motivated by the needs of stakeholders and describe the details of a software-based IS [ISO11]. More precisely, a requirement can be defined with regard to three aspects: i) it describes a condition or feature that is required for solving a problem or reaching a goal [PR15]; and ii) it involves a condition or feature that is fulfilled or provided by a system to complete a contract, regulatory act or any kind of specification. iii) it is documented according to i or ii. The specific role who is responsible for the activities of RE – and thus for the specification of requirements – is a requirements engineer. Requirements are documented in natural language or more formally as models. Natural language is the most important way to document and communicate requirements, ideally with an agreed and semantically defined terminology [IRE15]. The UML is a widely used modeling language for requirements with a semi-formally defined semantics [Obj15]. The complete set of requirements that makes up a system is called a software requirements specification (SRS). Although a software requirements specification (SRS) can be documented and structured in various ways, the ISO/IEC 29148 standard defines the structure of a SRS document [ISO11]. We use the term individual requirement (InR) as a synonym for the main term *requirement* whenever we need to distinguish between a single requirement and a SRS.

2.1.2 Unified Modeling Language

Natural language as a form of notation for requirements is universal, but can be imprecise since it has a low degree of formalism. Therefore, models are often used to specify requirements in a semi-formal or formal way. A great variety of modeling notations for requirements are available, such as the UML, the systems modeling language (SysML), the business process model and notation (BPMN), i*, goal models, scenarios and other such similar forms of notation. In this thesis the semi-formal modeling language UML is used for data models and activity diagrams, which are specified in the current version 2.5 by the Object Management Group (OMG) [Obj15]. UML has evolved hand in hand with object-oriented software development and thus contains object-oriented concepts.

A basic understanding of UML is essential for the UML-based TORE domain data model as introduced in Section 2.2, and the domain ontology usage patterns which are presented in Chapter 9. In the following, some relevant model elements are explained.

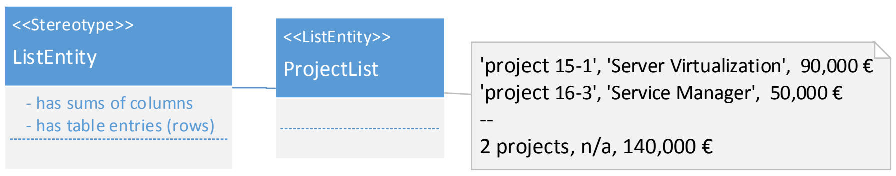
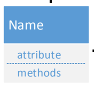

<<Stereotype>>
ListEntity
- has sums of columns
- has table entries (rows)
<<ListEntity>>
ProjectList
'project 15-1', 'Server Virtualization', 90,000 €
'project 16-3', 'Service Manager', 50,000 €
--
2 projects, n/a, 140,000 €

Figure 2.1: UML stereotype

Class models contain classes which represent any kind of entity that contains attributes and methods. Classes are notated by 

. For example, the class `BankAccount` has the attributes `overdrawLimit` and `currentBalance`, and the operation `setNewOverdrawLimit`. Associations between classes are either directed ($\longrightarrow$) or non-directed (——). The two relations of aggregation and composition both describe the relation between a whole and its parts. The aggregation (◇——) shows which parts (multiplicity 0..1) the whole consists of (multiplicity 1). The composition (◆——) is a special aggregation. It describes the relation of a whole to its parts, where the parts cannot exist without the whole. For example, a human cannot exist without a heart and a brain. *Stereotypes* are used in UML to extend model elements such as classes, relations or packages. Stereotypes define possible usage contexts of the model element and are indicated by the keyword «stereotype». The example shown in Figure 2.1 shows the definition and use of a stereotype «ListEntity». The stereotype «ListEntity»declares that all classes of this stereotype are expected to have the presence of multiple table rows and the respective sums of columns. The stereotype symbol indicates that the stereotyped class or classes provide the specialties of multiple table rows and the sums of columns.

2.1.3 Requirements Quality

InRs can be formulated as natural language texts or more formally as models. These requirements can be affected by quality issues, leading to wrongly functioning systems and additional effort during system development [EK08, Boe01, Poh16]. This is why InRs and SRSs should fulfill certain quality attributes, as defined by ISO/IEC 29148 [ISO11], such as being unambiguous, consistent, complete and correct.

The ISO standard 29148 [ISO11] defines requirements quality attributes (RQAs) that must be fulfilled for InRs and SRSs. The RQAs are summarized in Table 2.1. We also label whether they apply to InR or SRS. RQAs are varying in their importance. For example, ambiguity in requirements is less important than expected [PHK^{+}13, dBD10].

Table 2.1: A list of requirements quality attributes according to the ISO 29148 [ISO11] definition, InR=individual requirement, SRS=software requirements specification.

Quality attribute	Description and scope
Necessary	The requirement defines an essential capability or characteristic and is necessary for a product or process. (InR)
Impl. Free	The requirement states what is required in an implementation-independent way, not how the requirement should be met. (InR)
Unambiguous	The requirement can be interpreted in only one way, and is simple and easy to understand. (InR)
Consistent	The requirement is free of conflicts with other requirements. The SRS does not have contradictory or duplicated individual requirements. The same term is used for the same item in all requirements of the SRS. (InR, SRS)
Complete	The requirement is measurable and sufficiently describes the needed capability and characteristics. The SRS needs no further amplification as it contains everything pertinent to the definition of the specified system. (InR, SRS)
Singular	The requirement includes one and only one requirement. (InR)
Feasible	The requirement is technically achievable and fits within system constraints. (InR)
Traceable	The requirement is traceable to its source and to other system definition artifacts. (InR)
Verifiable	The requirement has the means to prove that the system satisfies the specified requirement. That means that the requirement is testable. (InR)
Affordable	The SRS can be satisfied by a solution that is feasible within defined constraints regarding cost, schedule, technical, legal, regulatory. (SRS)
Bounded	The SRS keeps the scope of the intended solution without increasing it beyond the user's needs. (SRS)
Correct	The stakeholders have confirmed that requirements are expressed correctly. Any possible conflicts have been resolved. (InR)

Task	Strategic information management
Purpose	Creation and use of IT strategy
~~Source~~	~~Interview with Mr. X, 28. June 2017~~
Responsible	Chief Information Officer

Subtask 1: **Creation of Strategic Information Management Plan**	
1a	Approval of IT strategy stakeholder Two stakeholders from the divisions of patient care, research and education, and information management are nominated for the information management board. The system sends a notification and lists the selected persons on a website.
1b	Perform workshop to define strategic IM-actions The IT strategy stakeholders participate in a workshop to define IM-actions that realize the hospital's strategic goals. The system allows the documentation of IM-actions and shows the results on a website for the management.
1c	Documentation of IM-actions The prioritized IM-actions are documented in the IT strategy. The system provides the IM plan template and supports its collaborative creation.
1d	Presentation of the Strategic Information Management Plan The CIO presents the created strategic IM plan to the hospital's board of directors.
Subtask 2: **Use of IT strategy**	
~~2a~~	~~Re-prioritization of projects~~ ~~The CIO adapts the prioritization of planned projects according to their contribution to IM-actions, as defined in the strategic IM plan.~~
2b	Prospective and Retrospective justification of the department's finances The CIO gives a report about the costs incurred to realize the defined IM-actions.
2c	Evaluation of the Strategic Information Management Plan's achievement At the end of the financial year, the compliance with the strategic IM plan is examined.

Figure 2.2: Example of a user task requirement with defects

As an example to demonstrate some requirements' defects, we use the task description presented in Figure 2.2. Although user tasks and the context of IM will be introduced in more detail in Section 2.2 and Section 2.4, for now it is important to understand the context of the example. The task in Figure 2.2 describes the activities, called subtasks, that Mr or Mrs X in his or her role as CIO executes to create and use a strategic IM plan. A strategic IM plan is a document which describes the measures used to strategically align the IM department in a hospital, let us say over a five year period. The IM department manages the hospital ISs to deliver IT services. The tasks specifies that the CIO approves the IT strategy stakeholders (subtask 1a), in order to perform a workshop in which IT actions are defined and prioritized (subtask 1b). The IT actions are used for the creation of the strategic IM plan (subtask 1c). Finally, this plan is presented to the hospital's board of directors for approval (subtask 1d). The strategic IM plan is used in multiple ways, such as the re-prioritization of projects which contribute to strategic goals (subtask 2a), the justification of expenditures (subtask 2b), and the evaluation of the strategic IM plan's achievement (subtask 2c).

Based on this example, we explain the RQAs InR.correct, InR.unambiguous, InR.consistent, InR.traceable and SRS.complete. The underlined and crossed-out text in the example indicates the requirements quality defects.

1. InR.traceable: The source of the requirement was an interview with Mr. X, who is missing in the header of Figure 2.2 as indicated by the crossed-out text. As a result, it is not clear from where the information captured in this template originates.
2. InR.unambiguous: Subtask 1a does not specify whether the IM board contains two stakeholders at all, or whether two stakeholders of each division form the IM board.
3. InR.consistent: A domain expert would see that the term *IT strategy* in subtask 1c is used falsely instead of the document *strategic IM plan*. Accordingly, the name of *subtask 2* must be named *use of strategic information management plan*.
4. SRS.complete: If subtask 2a was missing in the specification, the SRS would be incomplete and the system to be developed would lack features related to this subtask.
5. InR.correct: The subtask 2c has not been validated with the stakeholder, who would have discovered the following mistake: the compliance with the strategic IM plan is examined *before* the financial year ends.

2.2 Task-oriented Requirements Engineering

Task-orientation focuses on the RE process and helps to deliver software that satisfies user needs [ADEG09, PK04]. The main aspect of task-oriented RE relates to the tasks that a user needs to execute with the support of a system. The task-oriented framework TORE is based on 18 Decision Points (DPs) which are grouped into four levels of abstraction, as a conceptual model as is shown in Figure 2.3. The Decision Points (DPs) delineate important aspects of a system's design which need to be decided on during the requirements specification. The notion of DPs provides a conceptual model for RE, which is independent of particular domains, processes and artifact types. There is no strict order of deciding DPs. The activities related to the gathering of relevant information for a DP can be executed in parallel, whenever it is appropriate and possible in the overall RE process. Note also that decisions in DPs are distinct from decisions modeled as requirements for DSSs.

The relation between the terms *artifact*, *artifact type* and *notation* is shown in Figure 2.4. For each DP, TORE suggests possible *artifact types* for documentation, such as models,

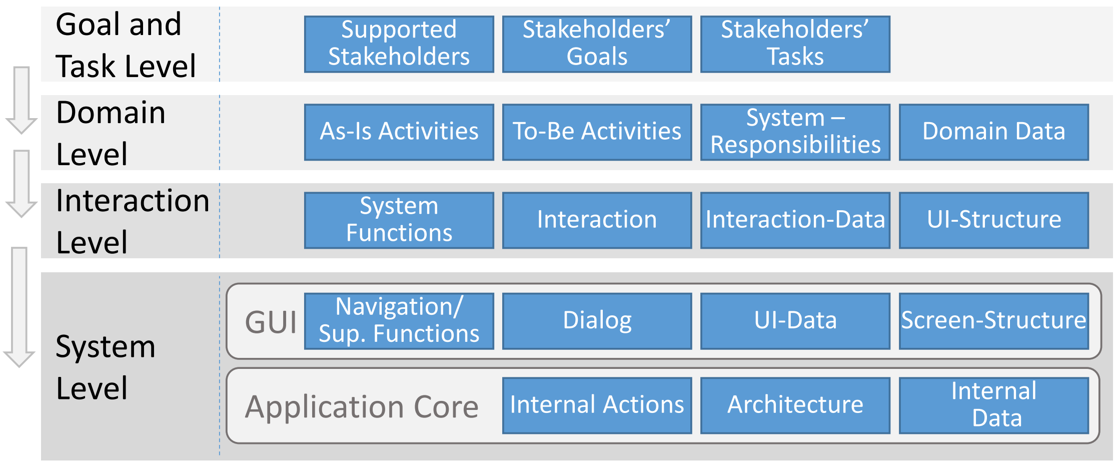


Figure 2.3: Decision Points of TORE according to [PK04] and [ADEG09]

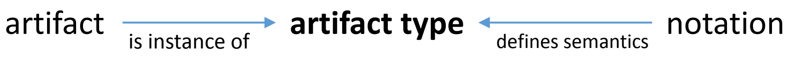


Figure 2.4: Artifact type, artifact and notation.

templates, and tables[1]. In the following section, we highlight the artifacts that are relevant to this thesis. More information on TORE specific artifact types or well-known artifact types can be found in [PK04, ADEG09]. Each artifact type follows a *notation* that comprises the syntax and semantics of the artifact type. The *artifact* is a specific instance of an artifact type, which describes a concrete requirement.

Example of an artifact, artifact type and notation

The task & support template by Lauesen is an *artifact type*. The *notation* and thus the semantics of the templates are defined by Lauesen in [Lau05]. The filled task & support template describing the task project management of hospital x is the *artifact*.

In this thesis, several artifact types are often used, as shown in Table 2.2. Artifact types are denoted in the following section as **bold face** and DPs are denoted in *italics*. A complete overview of TORE artifact types is shown for the sake of completeness in Appendix B. The next subsections provide a detailed description of TORE's DPs and the artifacts used to document the respective decisions, which are structured according to the four levels of abstraction.

[1]Note that the term *notation* used in [ADEG09] and [PK04] mostly covers both an *artifact type* and its *notation*. The rationale for distinguish between both of them is to make a clear distinction between structure (template), instance (concrete task) and semantics (allowed syntax and semantic definitions). Although the decision of a DP is documented with an artifact, the artifact is distinct from the DP.

Table 2.2: TORE Decision Points and supported artifact types relevant to this thesis [PK04]. Rows marked in grey are introduced in Chapter 6.

Level	Decision Point	Artifact Type
Goal & task	Stakeholders' Tasks	Task description
Domain	As-is activities, To-be Activities	Subtask template
Domain	Domain Data	UML class diagram, ER diagram
Domain	Categorization of Subtasks	Task description
Domain	To-be Decision-specific Subtask	Decision-specific subtask template
Interaction	UI-Structure	Workspace model
Interaction	System Functions	Function description
GUI	UI-Data	UI Prototype, Virtual window
GUI	Screen-Structure	UI Prototype, Virtual window

2.2.1 Task Level

On this level, the supported roles, tasks and goals of the to-be IS are decided. TORE's *goal & task level* contains three DPs. Stakeholders are persons or roles who will be supported by the developed IS (users) or who will influence the resulting system such as steering committees, financiers, or operations. It is important to have the exact set of stakeholders. In the DP *Supported Stakeholders*, these roles are determined and documented by the artifact types **stakeholder maps**, and by user descriptions such as **persona** [C^+04] or **role descriptions** [ADEG09]. The DP *Stakeholders' Goals* captures the primary goals that stakeholders want to achieve with the system. In TORE, both organizational and user goals are supported. There are several more frameworks to document stakeholders' goals, such as **goal graphs** as used in Goal Oriented Requirements Engineering (GORE) [EB13, SKS07] and i* [Yu97, YLM07], simple AND/OR **goal refinement trees**, or **natural language**. The DP *Stakeholders' Tasks* refines the DP *Stakeholders' Goals* by the intended supported user tasks[2]. User tasks are performed by users to reach a certain goal and consist of one or many activities or business processes. The to-be system supports the stakeholders' tasks that these users perform as part of their work in a specific role. Tasks can be divided into subtasks that are manageable and self-contained units of work. User tasks in TORE can be documented in the form of various notations. The literature proposes to use flexible and extendable *task descriptions* [Pae00, PK04]. In this thesis, we use Lauesen's **task & support** notation for the documentation of user tasks, as they allow us to document tasks and their details.

[2]The terms *user tasks* and *Stakeholders' Tasks* are used synonymously.

T1.2	**Check in**
Start	A guest arrives.
End	The guest has got a room(s) and a key. Accounting started.
Frequency	Total: Around 0.5 check-ins per room per night. Per user: 60...
Difficulty	A bus with 50 guest arrives
Subtasks:	**Example solution:**
1. Find room Problem: Guest want neighbor rooms; price bargain. 1a. Guest has booked in advance. 1b. No suitable room.	System shows free rooms on floor maps System shows bargain prices, time and capacity dependent.
2. Record guest data. 2a. Guest recorded at booking. 2b. Regular guest.	(Simple data entry, see data model) (Search with many criteria, e.g. name, booking number, phone)
3. Record that guest is checked in.	
4. Deliver key. Problem: Guest forgets to return key; guest wants two keys.	System prints electronic keys. New key for each customer.

Figure 2.5: Example of a user task described with Lauesen's task & support template [Lau05]

Lauesen's Task & Support: User task descriptions are functional requirements on the domain level [Lau03]. Lauesen presents several ways to document user tasks [Lau05, Lau03]. One way is an annotated task list that describes the user tasks along with existing problems, present ways and possible future solutions. A second is the task & support template [Lau05, Lau02], as shown in Figure 2.5. In task & support, an *example solution* describes explicitly a possible future solution that takes existing problems into account for each subtask.

A user task is identified by *ID* and a *name*. *Start* and *End* document what initiates a task and when it ends. The *frequency* indicates how often this task is carried out. Possible *difficulties* show situations when this task is hard, stressful or when it requires a high level of accuracy. Refined subtasks are also contained in the task description template. The subtask description resides on the domain level and illustrates both the as-is situation (now) and the to-be situation (future). The subtask is described with imperatival language by the pattern <verb> <object>, e.g. find(v) room(o). Existing or past problems are both described as *problem*. Variants of a subtask are indicated by adding a characters (a,b,c, ...) to the subtask ID. Lauesen proposes several other optional task headers, such as*triggers* for an external signal or an event, *goal or purpose* that describes the business purpose of the task, *user* that characterizes the responsible person who carries out the task, *Precondition* as the situation that must be valid before a task starts, and *post-condition* that is reached, when the task has finished.

ID and name	Subtask 2 - Record guest data
Task Description	The data of the arrived guest(s) is recorded in the system. If the guest has already registered in the past, the data is validated only.
Role	Receptionist and Guest
Trigger	Guest arrived
Performance	Maximum of 70 guests per day due to the limited number of beds
Risks	Data loss because of unintentional overwrite of data

Figure 2.6: Task Description according to [PK04]

2.2.2 Domain Level

On this level, the details of the user tasks, in particular their activities, domain data and the future supported activities, are specified. TOREs *domain level* contains the following four DPs: The user tasks of the *task level* are refined in the DP *As-is Activities* into subtasks. Subtasks describe the activities, especially how each task is currently being performed in detail. Appropriate artifact types for specification are **UML activity diagrams**, **business process notations**, Lauesen's **task & support** template, and **task descriptions**. An example of a task description is given in Figure 2.6. A subtask is identified by Any kind of identifier (ID) and/or *name*. A detailed textual description is given in the *Description*. If necessary, the supporting *Role* can be referred to. A *Trigger* starts the activity. Necessary requirements for the *Performance* of the activity can be captured, as well as possible *Risks* that are associated with the activity. The DP *to-be activities* describe the details of the future stakeholder tasks. The same artifact types of the DP *As-is Activities* apply. In the DP *System-responsibilities*, the to-be activities are defined, which are executed by humans, by systems or by both (that means, users who interact with the system). In some cases, only a subset of to-be activities are provided by a system. Possible artifact types for this DP are **UML activity diagram** and **activity description**. The entities, their attributes, and their relations relevant to the given problem and domain are determined by the DP *Domain Data*. The domain data model (DDM) is a model which describes the domain data as either **ER diagram** or an **UML class diagram**.

2.2.3 Interaction Level

On the *interaction level*, the specification of the human-computer interaction is made. This defines how the user interacts with the system, which system functions and interaction data are required to support the users, and how the user interface (UI) needs to be structured.

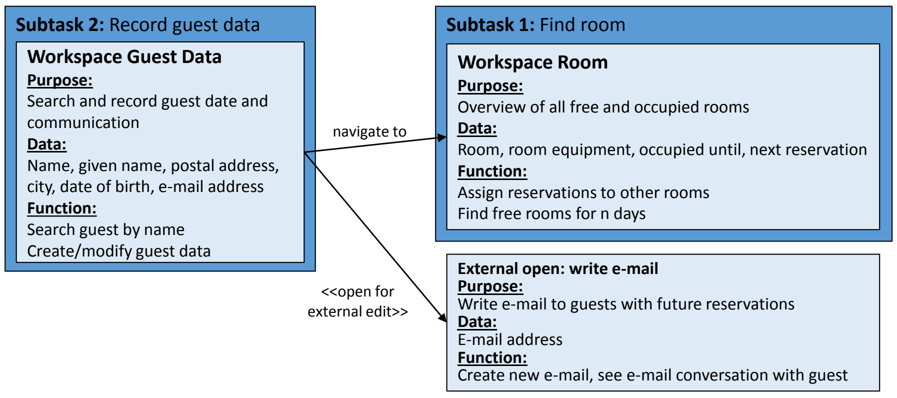
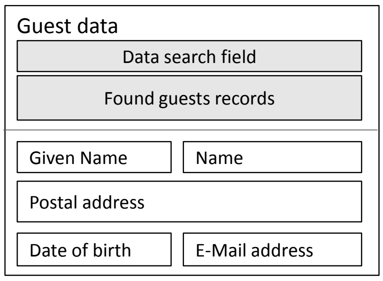

Subtask 2: Record guest data
Workspace Guest Data
Purpose:
Search and record guest date and communication
Data:
Name, given name, postal address, city, date of birth, e-mail address
Function:
Search guest by name
Create/modify guest data
navigate to
<<open for external edit>>
Subtask 1: Find room
Workspace Room
Purpose:
Overview of all free and occupied rooms
Data:
Room, room equipment, occupied until, next reservation
Function:
Assign reservations to other rooms
Find free rooms for n days
External open: write e-mail
Purpose:
Write e-mail to guests with future reservations
Data:
E-mail address
Function:
Create new e-mail, see e-mail conversation with guest

Figure 2.7: An example of a simple workspace model

The *interaction level* contains four DPs. The DP *System Functions* determines which system functions the system will provide and to what extent. System functions are executable operations that consume, manipulate or produce data. Details of the system functions are visible behavior, input, output, and so on. Typical visualization functions such as the display of a data set are no system functions. The artifact type of this DP is a **free text function description**, or a **system function description** based on a template, as shown in Figure 2.8a. The DP *Interaction* describes the interaction between an actor and system activities, based on a **use case** template. The use case template used in the context of TORE emphasizes the flow of events, their exceptions, and it also shows for each **system responsibility** how the user navigates in the UI structure. The DP *Interaction-Data* determines the data necessary for the user/system interaction, which is finally displayed in the system's UI. The interaction data represents the input data and output data of system functions. One possible artifact type is the **interaction data model (IDM)**, which follows a **UML class model** notation. In the case of high complexity, it is reasonable to show an excerpt of the IDM with the scope of a set of tasks or subtasks. The DP *UI structure* determines details about the combination of workspaces, data, and system functions. Workspaces group data together with system functions, specifically for a set of stakeholder tasks, and define the navigation to other workspaces. The data and system functions contained in one workspace are finally close together in the UI. The DP *UI Structure* is documented with a **workspace model**. An example of such a model is shown in Figure 2.7. The workspace *record guest data* performs the purpose of searching through and recording guest data. The required data to complete the subtask are the guest's personal details, such as name and e-mail address. The required system functions are search for a guest's data and create or modify the guests data. The presented workspace model also shows that users possibly navigate to the *find room* workspace or an external application for sending e-mails. The data in the workspace model must be consistent with the IDM.

Name	Search guest
Input data	at least a part of: name, given name, postal address, date of birth or e-mail address.
Output data	none, one or many guest data record(s).
Description	The systems searches for parts of the input data in all given fields of the guest record.
Exceptions	The guest database is unavailable.
Rules	none
Quality require.	fast (less than 2 seconds for the answer time)
Preconditions	Guest records must be in database.
Post conditions	The identified guest data is returned.

(a) An example of system function



(b) Example of virtual window

Figure 2.8: Example of an artifact: system function and virtual window

A further refinement of **workspaces** are **virtual windows (VWs)**. VWs are used in this thesis also for the purpose of prototyping. Figure 2.8b shows an example of a VW for the previously presented subtask *check in* in Figure 2.6 and its associated workspace in Figure 2.7. Virtual windows (VWs) have been proposed by Lauesen [Lau05] to illustrate the need for information early in the RE process. VWs contain all data that is needed at once to fulfill a user task. Data can be grouped into related blocks of information and show realistic sample data and extreme data. VWs can support one or many user's tasks. For the sake of simplicity, VWs omit showing system functions or interaction elements such as menus or buttons [Lau05]. VWs are a precursor of UI prototypes and are used to link user tasks, data models, and user interface design [LH01].

2.2.4 System Level

Decisions about the system's internals and the graphical user interface (GUI) are made on this level. The *System Level* is divided into the two sub-levels *GUI* and *Application Core*. The first refines in particular the *interaction level* with platform- and implementation-specific GUI details. The latter is concerned with the definition of various technical aspects of the system.

2.2.4.1 GUI

The three DPs *Screen Structure*, *navigation/support functions*, and *UI-data* are closely related to refine the UI-structure into a platform-specific screen structure. The platform typically predefines the UI design elements, for example the fact that desktop systems have other UI controls from mobile platforms. As the main artifact type for these DPs, **UI prototypes** are suited to define the screen structure, the navigation functions and the UI-data. Depending on complexity and needs, **UI prototypes** can be paper-based, sample mock-ups or interactive prototypes without functionality. The DP *dialog* refines the user and system interaction by the definition of screen sequences or some parts of it. **State diagrams** can be used to describe screen dependencies and sequences in combination with the user interactions, and are expressed as signals.

2.2.4.2 Application Core

The DP *Application Core* contains the three DPs *Internal Actions*, *Internal Data*, and *Architecture*. In object-oriented implementations, the internal actions are equivalent to methods of a class, and the internal data are equivalent to attributes of realizing classes. The DPs *Internal Actions* and *Internal Data* include the design decisions for the implementation. Such decisions cover the software structure, internal data models, design guidelines and the software design patterns to be used and/or developed. At this level, the structure and behavior of components are defined and documented in several models. These are a **functional model**, UML structure models such as a **class diagram**, and UML behavior models such as **collaboration diagram**, **activity diagram** and **state chart diagram**. The DP *Architecture* captures all decisions relevant to the architecture. These are physical constraints given by the customer and in particular are the defined non-functional requirements. Nonfunctional requirements determine the quality aspects of a system, such as security, availability, usability, and have a high impact on the system's architectural design. Although nonfunctional requirements are enormously relevant to any system, they are not in the focus of this work. More information on nonfunctional requirements can be found in [Som09] and [Int11].

2.3 Ontologies

Ontologies were originally used in Greek philosophy and in the writings of Aristotle, who attempted to classify and understand the nature of being and the structures of reality. In computer science, *computational ontologies* have been used from the 1990s as an integral part of knowledge-based systems [GOS09]. Ontologies conceptualize real-world knowledge as formal concepts, their relations to each other, and their rules [HP09]. In this work we follow the definition of a *computational ontology* from Studer et al.: "An ontology is a formal, explicit specification of a shared conceptualization" [SBF98].

Guarino et al. [GOS09] summarize the meaning of this definition as follows. The term *conceptualization* signifies an abstract and simplified view in an area of interest of the real world, for example the application domain. A conceptualization contains objects, concepts, other entities, and their relations to each other. *Formal* means that the expressions of conceptualization (the concepts) are given in a formal language. The expression *explicit specification* refers to the intentional use of axioms that define the semantics of the vocabulary in a formal language, for example the relation *cooperates-with* between two persons is symmetrical, non-reflexive and intransitive. In other words, an ontology can be seen as an abstract model that is used to share information about explicitly defined and generally understood concepts and constraints of a specific knowledge domain.

Types of Ontologies: There are different types of ontologies that exist [SBF98]. Domain ontologies are an example of such a type. Different classification schemes help to distinguish between the different types of ontologies that exist, based on the ontology's content. We discuss the different ontology types and classification schemes in Section 8.2 on page 94.

Ontology Languages: A great variety of formal ontology languages are used to describe ontologies [DVB+15, VRM16]. The resource description framework (RDF) is a data model used to represent binary relations between identifiable resources such as concepts, relations or properties [Pan09]. In RDF, each concept is represented by a triple relation of *subject*, *predicate*, *object*, which describes the relations (predicate) between real-world subjects and objects, for example `(SRS, consistsOf, InR)`. Dermeval et al. [DVB+15] shows that the Web Ontology Language (OWL), as defined by the World Wide Web consortium (W3C) [AvH09], is the most prominent ontology-description language used in context of RE. OWL extends RDF and provides a set of standard relations with a predefined semantics. In RDF, classes are used to store concepts. Instances of classes are called individuals. Ontology classes typically form what is known as a metamodel in SE. We focus on RDF/OWL ontologies.

Querying and Reasoning: An ontology can be *queried* by a formal language such as SPARQL [HBS09] in order to access its concepts, relations, and individuals. With the help of *automated reasoning*, logical inference is supported based on rules (the axioms) that are contained in an ontology.

Ontology Engineering: The term *ontology engineering* [GPFLC04] refers to the methodologies, tools and languages, and all activities that enable the development and maintenance of ontologies. The activity *ontology learning* means to create an ontology from literature or other sources, in particular to create or extend the ontology's concepts, relations, and axioms [CMSV09]. The activity *ontology population* means to extract terms from a source, for

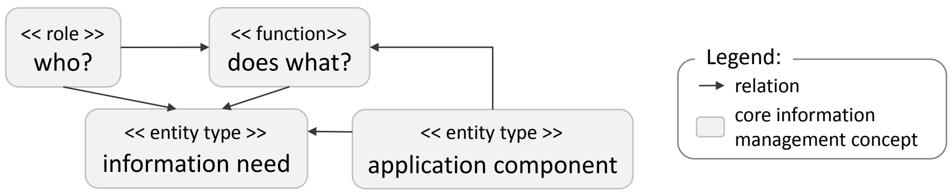


Figure 2.9: The main IM concepts *function*, *role* and *entity type* [WHA+11]

example requirements, and to add them as individuals to the classes of the ontology [MLP08]. The modeling of OWL/RDF ontologies can be effected with Protégé [3], which also offers various graph-based visualization plug-ins.

2.4 Information Management

The management of ISs is called information management (IM), comprising all activities to successfully delivery IT services both today and in the future. There are different general definitions of IM available [HRS14, Krc15, WHA+11]. Since the case for this thesis is available through the SNIK-project (see Section 2.5), the application domain is *IM in hospitals*. Winter et al. [WHA+11] focus on *health information systems*, and this is therefore used as the IM definition for this thesis.

The term *management* refers to the management tasks that are necessary to define the goals, structures and behaviors of an organization. The management of ISs in an organization is called information management (IM). IM comprises the management tasks related to the management of information processing in an organization, and in particular in this context, to hospitals. The use of IM enables the hospital to undertake and oversee a systematic information processing that contributes to its strategic goals. The tasks of IM are *planning*, *directing*, and *monitoring*. Planning focuses on the arrangement of ISs and their architecture according to organizational goals. Directing focuses on the implementation and establishment of ISs and their operation. Monitoring focuses on the verification of ISs development and its operation. The domain of e-health and hospitals is very sensitive with specific requirements, since the ISs that are used directly support the medical treatment of patients. IS-failure, unavailability, errors or data-leakage has the potential to cause direct harm to humans.

Winter et al. divide IM into strategic information management, tactical information management and operational information management in focusing on the management of hospital information system (HIS). HISs are specialized ISs that deal with processing data,

[3] https://protege.stanford.edu/

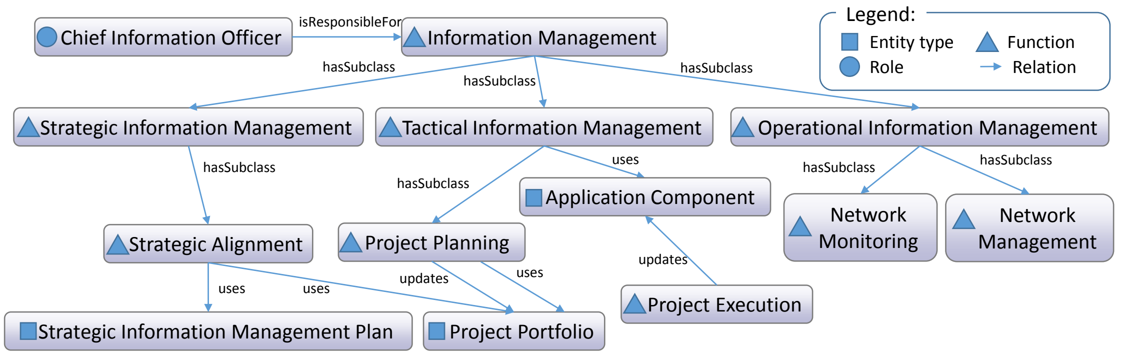


Figure 2.10: Example of IM functions, roles and entity types [WHA+11] (source: www.snik.eu/graph)

information, and knowledge in healthcare environments. *Strategic information management* establishes strategies and principles for the IM division that results from a strategic information management plan. *Tactical information management* follows the needs of *strategic information management* and implements solutions or changes to the IM aspects of organizations, typically through projects. The provisioning of IT services is the primary task of *operational information management*. Both the IT service management and the IT governance are closely related to IM. IT service management focuses on operational management in order to enable the provisioning of IT services, while IT governance focuses on the need for strategic alignment if the technical and organizational infrastructure are to deliver IT services. The IM concepts are structured into *functions*, *entity types* and *roles*, as shown in Figure 2.9. IM functions are the tasks that a certain role executes or is responsible for. IM functions use or modify entity types. The term *IM function* describes the same phenomena as in the case of TORE's tasks and subtasks. As the focus of this thesis is RE, we use the terms tasks and subtasks after the introduction of IM, except for making references to the SNIK ontology. Application components are the building blocks of HISs, that support IM functions within organizations in general and especially in organizations related to the healthcare sector. All IM-activities consist of IM functions that use or update information and are supported by application components in organizations.

Figure 2.10 shows an excerpt of conceptualized knowledge of the IM in hospitals, which is available in the SNIK ontology (see subsection 2.5.2). In the following section, this excerpt is explained to provide a detailed impression of the IM domain. The Chief Information Officer as head of the IM department is responsible for the function **IM**, which, as we have seen, divides into the three dimensions of **strategic information management**, **tactical information management**, and **operational information management**. The function **strategic alignment** is a subclass of **strategic IM**, and thus inherits properties from its

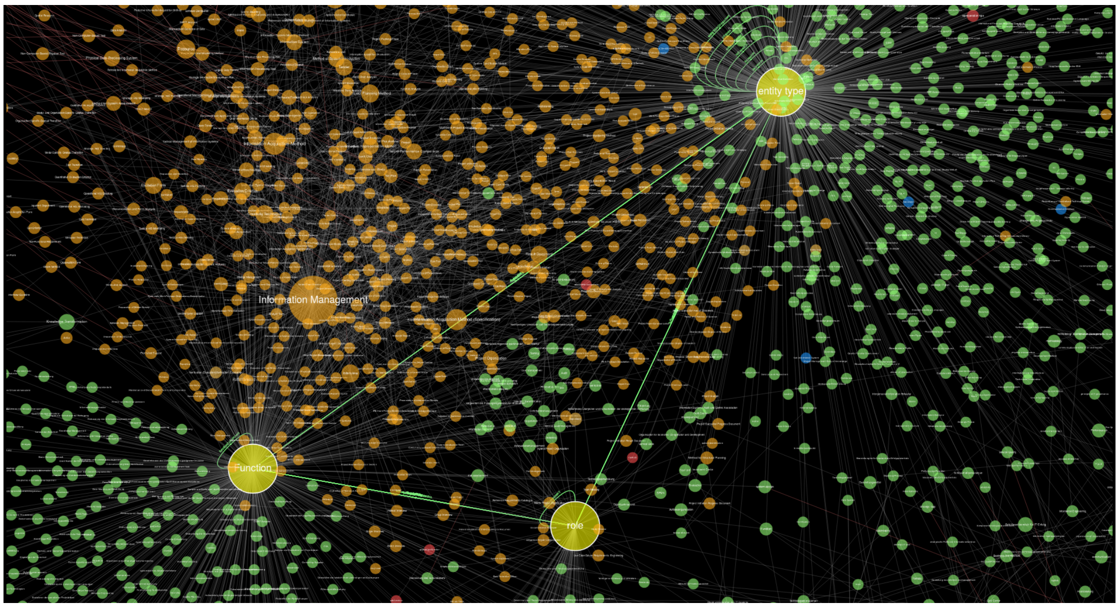


Figure 2.11: An excerpt of the SNIK ontology (source: www.snik.eu/graph)

parent, in this case the fact that the *role* CIO is responsible for subclasses of strategic IM functions. The purpose of the function **strategic alignment** is to align the division's goals with the overall strategic goals of the hospital. Strategic goals are documented in the *entity type* **strategic IM plan**, and are realized through **projects** that are documented in the *entity type* **project portfolio**. The function **project execution** belongs to the **tactical IM** and modifies the *entity type* **application components** – that is, the HIS components – of the hospital. The *function* **project planning** uses and updates the *entity type* **project portfolio**. The functions **project planning** and **project execution** are subclasses of the **tactical IM**. The functions of **operational IM** are **network monitoring** and **network management**.

2.5 Project Semantic Network of Information Management in Hospitals (SNIK)

This section gives a brief introduction to the project semantic network of information management in hospitals (SNIK)[4]. The objective of SNIK is the creation of a semantic network of IM in hospitals. This network contains IM terms and their relations, and is beneficial for education purposes and for the use during the development of IM-specific ISs [JSPW14].

[4]Funded by the DFG (German Research Foundation) under the grant no. 1605/7-1 and 1387/8-1. The project's website can be found at www.snik.eu.

2.5.1 Project Goals

In the SNIK project three goals are pursued: (1) The creation of a theoretically and empirically evidenced ontology that contains and relates the concepts of IM in hospitals, as based on specific literature within this domain [JSPW14, SJT+15]. Ontologies conceptualize real-world knowledge and the relations between such knowledge elements. (2) The development of lectures that use the SNIK ontology to convey IM knowledge to students [JSK+16]; and (3) The development of a software tool for CIOs to navigate through the data of IM, as based on the SNIK ontology.

2.5.2 SNIK Ontology

The SNIK ontology contains a semantic description of the IM-specific to the e-health domain. The contained IM concepts and relations are structured according to the main IM concepts, as introduced in Section 2.4 and Figure 2.9, and is defined by a metamodel. This metamodel defines the allowed class types *role*, *function*, *entity type* and the allowed relations between them. The metamodel follows the main IM functions. For reasons of completeness, the SNIK metamodel is shown in Appendix C. More details of the SNIK ontology can be found in [JSPW14, SJT+15, HJK+17].

The SNIK ontology provides a knowledge base for the development of an IM-specific tool to support the decision-making of CIOs. As of December 2017, the ontology contains 2,000 classes, 4,500 relations and 15,000 annotations, and is still evolving. Annotations describe details of the entities such as description, their associated type within the metamodel, different meta data attributes such as the page number or chapter of the entity's origin, and subclass relationships. Figure 2.11 presents an overview of the SNIK ontology as an example. A browser-based visualization of the SNIK ontology[5] allows one to search for and navigate in the SNIK ontology [SJT+16]. Concepts can be expanded to see their source and detailed description.

2.5.3 CIO Navigator

The project team develops, besides the ontology, a dashboard for the CIO, which is called CIO navigator for the specific hospital x (CIONx). The CIO has the goal of successfully providing IT services for the hospital staff, and therefore applies the principles of strategic

[5]The visualization is available here: www.snik.eu/graph.

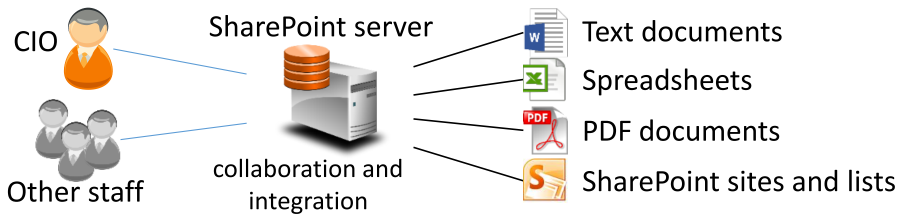


Figure 2.12: Integration aspect of CION

information management, tactical information management and operational information management. The current situation in IM departments of hospitals is that the CIO needs to collect distributed and scattered information in several systems and documents for individual decisions, as is illustrated in Figure 2.12. The goal of the CIONx tool is to support the CIO in such decision-making processes. CIONx shall enable the navigation in an information base of IM data with a decision-specific data presentation made up of different data sources. One idea of the CIO is to use CIONx as a tool to provide a transparent view of the operative and tactical data of the IM department for IT-staff and visitors. Within SNIK, the project team wants to use synergies between the knowledge gathered in the SNIK ontology and the tool development for the specific IM department.

The SNIK project provides the case for the studies of both domain-specific RE method adaptations. The studies of the system domain-specific adaptations are presented in Chapter 4, Chapter 5, and Chapter 7. The evaluation study of the application domain-specific adaptation is presented in Chapter 10.

Part II

Decision-specific Task-oriented Requirements Engineering

3 Introduction

This chapter introduces the foundations and basic terminology with which to understand *decisions* and their software support by DSSs, and explains the motivation behind the choice of a personal DSS as an example of the system domain-specific adaptation of TORE. In Section 3.1 an overview of RE methods for DSS are provided. In Section 3.2 the refinement into more specific RQs of the design problem 1 is explained. In Section 3.3 the details of the decision and decision-making are introduced in brief. Finally, the software support for decision-making by DSSs is explained in Section 3.4.

3.1 Problem Context

As was explained as a motivation in subsection 2.5.3 on page 32, the CIO as executive for the IM requires software support for decision-making in IM. Today, data warehouses (DWs) are a standard technology for decision support in companies. They are a specific form of DSSs which represent a subgroup of ISs. As discussed in a recent overview by Hosack et al. [HHPC12], there is a large variety of DSSs and an extensive history of research on DSSs. At the same time, this research is ongoing and incorporates new trends such as social media and mobile computing [Gao13]. Similarly, there is a growing field of research which focuses on the development methodologies for DSS. Saxena [Sax91] is one example of a very early development methodology which emphasizes the understanding of decision tasks and prototyping. In a more recent review, Gachet and Haettenschwiler investigate early and widely acknowledged development processes of DSSs [GH06], focusing on either decisions, on system engineering or on both. Gachet and Haettenschwiler emphasize the importance of an evolutionary approach which pays attention to both organizational and technical issues. In the same vein, Arnott [Arn10] has developed a business intelligence development approach which starts from a fundamental understanding of senior executives' behavior. We are interested in *personal decision support systems (PDSSs)*, as introduced in detail in subsection 3.4.3, which are small-scale systems that support decision-making of one or a small group of executives with an exactly defined set of supported decisions [Arn08]. While the early development processes

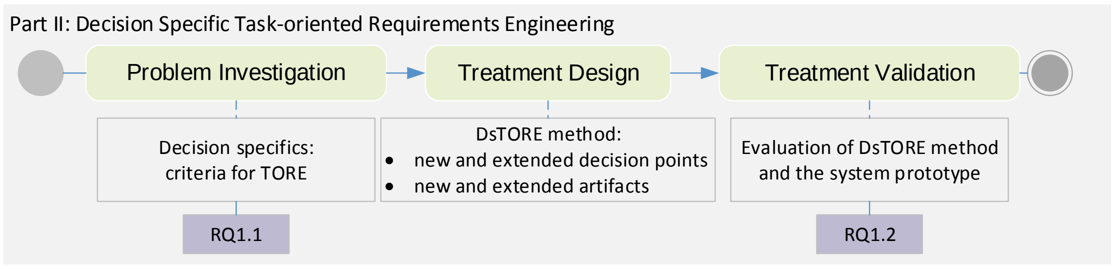

Part II: Decision Specific Task-oriented Requirements Engineering
Problem Investigation
Treatment Design
Treatment Validation
Decision specifics: criteria for TORE
DsTORE method:
• new and extended decision points
• new and extended artifacts
Evaluation of DsTORE method and the system prototype
RQ1.1
RQ1.2

Figure 3.1: Overview of design science approach and RQs for Part II

for the specification of DSSs provide a general framework, they do not describe a detailed RE method. Requirements engineers need to specify the requirements for PDSSs with an appropriate RE methodology. An overview of more recent RE methods for the specification of DSSs, focusing on either decisions, on system engineering or on both [GH06, SG09, GRR16], is given as related work in Section 6.1 on page 65.

3.2 Research Questions

The design problem 1 as introduced on page 7 is: *Improve TORE by designing DsTORE that satisfies the use of a decision-specific domain-specific modeling language, in order to support requirements engineers in providing a good system domain-specific software requirements specification*. We refine design problem 1 into RQ.1.1 and RQ.1.2 as will be discussed below.

RQ.1.1: What decision-specific information is missing in TORE to support the specification of a PDSS?

RQ.1.2: How well does DsTORE support the specification of requirements for a PDSS?

The design problem 1 demands a TORE extension in the form of a DSML that considers system domain specifics. The chosen example for the design problem 1 is the system domain of PDSS in the application domain *IM in hospitals*. In order to define a solution to design problem 1, it is essential to understand current weaknesses in a PDSS requirements specification with TORE. In order to provide such an extension, we follow the design science methodology (on which see Section 1.4 on page 8). First, it is essential to understand what are the phenomena which must be improved and why. Thus, we investigate the system domain-specifics that are not explicitly considered in TORE. Therefore we raise RQ.1.1 that queries the missing decision-specific information in the SRS that is created on the basis of TORE.

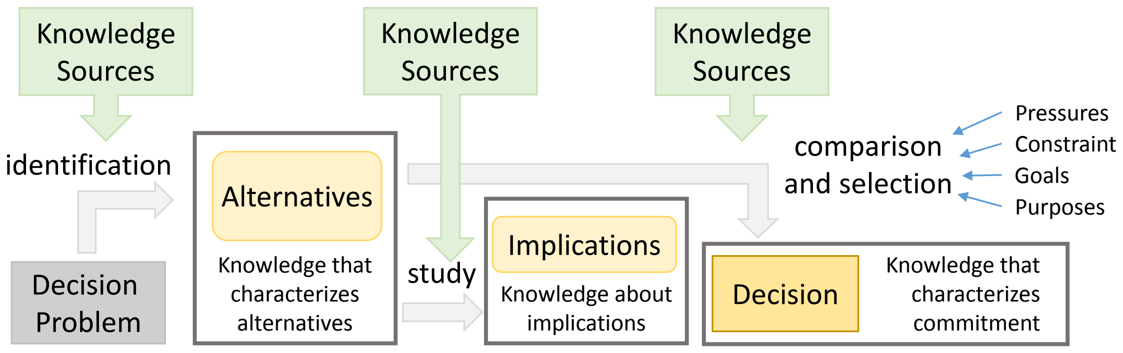


Figure 3.2: A knowledge-based conception of decision-making [Hol08a]

In the treatment design, we adapt TORE with the goal of explicitly supporting the system domain-specific information. The result is the decision-specific extension DsTORE, which provides the DSML to create SRSs for PDSSs. According to the design science methodology, the treatment design needs to be evaluated in order to investigate the extent to which the design treats the problem. For this reason, we raise RQ.1.2 that validates the DsTORE design. Figure 3.1 shows the design science cycle used to answer RQ.1.1 and RQ.1.2 in Part II of this thesis.

The answer to design problem 1 is given as the system domain-specific DsTORE adaptation. The experiences gathered during the system domain-specific method extension are used in Part IV to answer the research goal of this thesis.

3.3 Decisions and Decision-Making

Holsapple [Hol08a] defines a decision as a choice that is made between multiple alternative courses of action, and which in turn leads to a desired objective. This is illustrated in Figure 3.2. Decision-making is a non-random selection of one out of many alternatives. For a given decision problem, appropriate alternatives need to be identified. Alternatives have implications that need to be studied with regard to each alternative. Finally, the alternatives are compared with each other. The selection of the favorite alternative is influenced by pressures, constraints, goals and purposes and becomes the decision. For the activities *identification*, *study*, *comparison and selection*, knowledge sources are used. The investigation and documentation of alternatives, implications and the decision creates new knowledge. The process of decision-making is named in Holsapple as the *decision-making episode*, but is defined for the purposed of this work as simply *decision-making*.

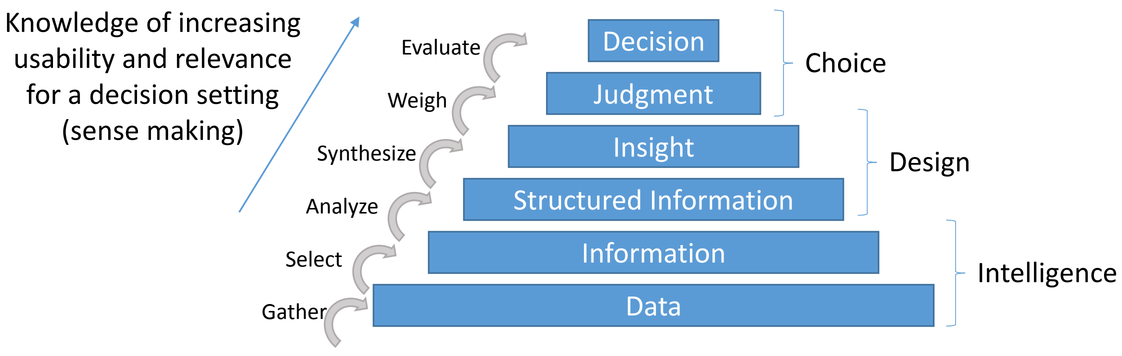


Figure 3.3: States of Knowledge [Hol08a]

The following example illustrates the decision-making conception in Figure 3.2. **Decision problem:** A logbook software for vehicle services is available in a new version 5.0. The current version 4.0 does not cause any troubling issues, but there is no reliable information about the end of life cycle for the version 4.0. **Alternatives:** (a) Keep the current version 4.0; (b) upgrade to the new version 5.0; (c) wait for the improved version 5.1 which contains bug fixes. **Implications:** Choosing to remain on the current version creates a technical debt that might cause trouble in the future, when the support for version 4.0 ends. Upgrade to the version 5.0 might cause failures. Waiting for version 5.1 has no implications. **Decision:** As the operations staff has some available resources, the CIO decides for alternative (b).

The traditional process of decision-making can be extended by the knowledge-based view [Hol08a, HW01]. The basic assumption of the knowledge-based view is that a decision *is* knowledge and that new knowledge emerges whenever a decision is made. Decisions are based on knowledge manipulation, which comprises the gathering, assembling, transforming, and constructing of information. Holsapple describes the knowledge of different states, as is illustrated in Figure 3.3. *Data* is gathered from various sources. The selection of data generates *information*. When information is analyzed, it leads to *structured information*, which can then be synthesized to gain *insights*. Insights are weighed to obtain a *judgment*. If a *judgment* is evaluated, it becomes a *decision*.

Holsapple further distinguishes three types of decision problems [Hol08a]. *Structured decision problems* have defined criteria in order to make the decision. They have known alternatives and implications, and require specific types of knowledge. By contrast, *unstructured decision problems* are by their nature unexpected, have an unstable context and unknown alternatives, possess unknown criteria, and arise out of situations in which either no knowledge or an incomplete level of knowledge is available. *Semi-structured decision problems* are in between structured and unstructured decision problems.

Decision knowledge is stored in a *data source*, and accessed via interfaces or through file access. A data source is described with reference to the data source format, location, and content. For example, a project list used by an executive can be described with reference to a spreadsheet (format), a uniform resource locator (URL) (location), and a list of projects (content). A text document contains unstructured data, whereas the structured data of a spreadsheet follows a formal definition of data types. Thus, data sources of structured data are typically databases, while semi-structured data are kept in spreadsheets, and unstructured data in text-documents. The different types of data sources are summarized as follows:

Types of Data Sources

Databases: containing structured information.
Spreadsheets: containing semi-structured information.
Text documents: containing unstructured information.

3.4 Decision Support Systems

This section introduces decision support systems (DSSs) in greater detail. The definition of DSS is given in subsection 3.4.1. A taxonomy of different types of DSSs in the scholarly literature is given in subsection 3.4.2. Details about PDSS as the chosen example of the system domain-adaptation are given in subsection 3.4.3. An overview of typical DSS-features is given in subsection 3.4.4. Finally, the CIONx tool is classified using the presented DSS types.

3.4.1 Definition of Decision Support Systems

Charles B. Stabell has investigated what a DSS is in the following way [Sta87]. He emphasizes that DSSs are used in situations where users — in particular, managers — need to make decisions about complex and badly structured problems. Holsapple defines a DSS as "a computer-based system that represents and processes knowledge in ways that allow decision-making to be more productive, agile, innovative, and/or reputable." [Hol08a]. From a more general view, Holsapple defines a DSS as a digital storehouse of knowledge. In general, DSSs support all three types of decision problems. In order to do this, DSSs cover a broad range of different system types and incorporate a variety of different techniques, such as decision models or artificial intelligence [Hol08b, PA05].

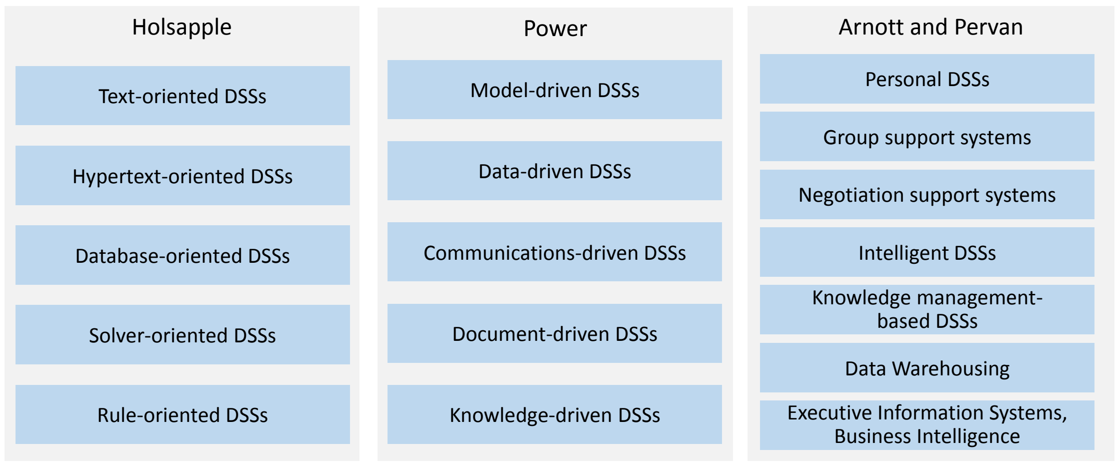


Figure 3.4: DSS taxonomies of Holsapple, Power, and Arnott and Pervan

3.4.2 Taxonomy of Decision Support Systems

The literature provides three taxonomies by which to classify DSSs, as are shown in Figure 3.4. The taxonomies of Holsapple [Hol08b], Power [Pow08b] and Arnott and Pervan [AP08] all differ from one another in their categorization, although they have some overlapping aspects.

Holsapple [Hol08b] focuses on both the data consumption of the DSSs ((hyper-)text-oriented, database-oriented), and the technology used for decision-making (solver-oriented, rule-oriented). Powers [Pow08a, Pow02] uses the DSSs's main focus for categorization, such as the decision model, the data orientation, a communication focus or a document focus. In this thesis, the taxonomy of Arnott and Pervan [AP08, PA05] is used, since it is the most fine-grained one that considers both the users and the type of data used that is used for decision-making. Arnott and Pervan propose the following disjoint sub-fields of DSSs:

1. **Personal decision support systems (PDSSs)** have evolved from Management Information Systems (MIS) and are individual small-scale systems for a single manager, supporting one decision task.

2. **Group Support Systems** combines information technology, communication technology and DSSs to support shared decision-making among a group of managers.

3. **Negotiation Support Systems** are based on concepts of game theory and focus on negotiation between opposing parties.

4. **Intelligent Decision Support Systems** utilize artificial intelligence (AI) to support decision-making.

5. **Knowledge Management-Based DSSs** enable decision support by providing knowledge management (storage, retrieval, transfer and application) for individuals, groups, and organizations.

6. **Data Warehousing** (DW) are systems that provide both data infrastructure and raw data for decision support.

7. **Enterprise Reporting and Analysis Systems** include executive information systems (EIS) and business intelligence (BI) systems. EIS are data-oriented DSSs that provide reporting about an organization to management of all levels. Drill-down functionality and multidimensional view to data, which have later evolved to online analytical processing (OLAP), are typical system functions of EIS.

3.4.3 Personal Decision Support Systems

Table 3.1: Operational and decisional systems (The left three columns are made in accordance with [GRR16] and [SG09], except the rows *structure of data* and *decisions*)

Criteria	**Operational**	**Decisional**	**PDSS**
Objective	business operation	business analysis	specific business decisions
Main functions	daily operations (OLTP)	DSS (OLAP)	decision-specific UI data presentation, reporting
Decisions	n/a	unstructured, semi-structured, structured	semi-structured, structured
Usage	repetitive, predefined	innovative, unexpected	repetitive, predefined
Design orientation	functionality	subject	subject
Kind of users	clerk	executives	executives
Number of users	thousands	hundreds	one to ten
Accessed tuples	hundreds	thousands	hundreds
Data sources	isolated	integrated	heterogeneous, isolated
Structure of data	structured	structured	unstructured and structured
Granularity	atomic	summarized	atomic and summarized
Time coverage	current	historical	current and historical
Access (work units)	simple transactions	complex queries	simple queries
Size	MB/GB	GB/TB/Petabytes	MB

The specifics of personal decision support systems are further detailed by Arnott [Arn08]. He summarizes that PDSSs are developed for either a single or a small group of managers with the goal of improving the process and outcome of decision-making. The scope of a PDSS is the support of either one or a small number of decision tasks, where often semi-structured data based on spreadsheets are used. The fact that PDSSs are small-scale systems refers to the supported small number of users and features and requirements. Salinesi et al. [SG09] provide a comparison of operational ISs and what are called decisional ISs, which is adapted in [GRR16]. We use this comparison to distinguish PDSSs from DSSs in general, and add a third column to Table 3.1. The *objective* of PDSSs is to support decision-making within a defined context. Its *main function* is to provide specific and predefined information for structured and semi-structured decisions in an overview UI or as a report for executives. In a similar way to DW, the primary *users* are executives, who access heterogeneous and isolated *data sources* at runtime. In contrast to DW, this usage is repetitive and predefined, and the amount of accessed data is not extensive. In summary, the data of PDSSs are complex, while their usage is simple.

3.4.4 Features of Decision Support Systems

An overview of typical features of DSSs helps to shape the requirements of TORE's DP system functions. There is not lot of scientific literature which is either about DSS-features, nor specifically about PDSSs. Power [Pow08b, Pow07] describes features of data-driven DSSs that cover — besides other things — data filtering and retrieval, predefined data displays, and excel integration. Holsapple mentions several DSS features which reside on different levels of abstraction. Although he does not discuss all these features in greater detail, they provide a way of recognizing the essential functionality of DSSs: (1) recognize occasions that warrant decisions; (2) acquire the additional knowledge from external sources; (3) select the knowledge; (4) generate the new knowledge; (5) assimilate the knowledge into the storehouse; (6) present the knowledge in desired formats; (7) coordinate the knowledge flows among decision participants; and (8) measure the decision process for future improvement. In addition, Turban defines several key characteristics of DSSs that are used to support semi-structured and unstructured decision problems [TSD10]: (1) support managers at all levels, individuals and groups; (2) support interdependent or sequential decisions; (3) support intelligence design; (4) support a variety of decision processes and styles; (5) be adaptable and flexible, and provide an interactive ease of use; (6) become used to increase effectiveness and efficiency; (7) ensure that humans control the process; (8) allow to be developed by end users; (9) provide decision modeling and analysis; (10) provide data access; (11) provide stand-alone integration ; and (12) be web-based. This presentation of features enables a first assumption of necessary features for the consideration in the RE specification.

3.4.5 Classification of the CIO-Navigator

The purpose and goal of CIONx is presented in subsection 2.5.3. We conclude the main goal of CIONx to be the support of the CIO in structured IM decision problems. The knowledge sources used for the decisions provide three types of knowledge states, namely *data*, *information* and *structured information* from any type of data sources. Data sources are *structured*, *semi-structured* and *unstructured*.

According to the characteristics presented in Table 3.1, CIONx is a PDSS for the following reasons: The *objective* of CIONx is to support the CIO (as executive) in specific IM-related business decisions. The *decision problems* are structured and semi-structured. The *usage* is repetitive with predefined decisions. The accessed *data tuples* are small, and they are contained in heterogeneous unstructured and structured *data sources*, for which see Figure 2.12. The *granularity* of data is both atomic and summarized, covering both *current* and *historical data*. Section 4.4 on page 56 provides details of the specific tasks of the CIO, which are supported by the implemented CIONx.

4 Domain Study

This chapter describes the domain study, which helps to understand the application domain IM in hospitals in greater detail. The domain study comprises two parts: (i) First, a system analysis of the IM department, which will be presented in Section 4.2. In this system analysis, the tasks of the IM department with CIO involvement and their support by ISs are investigated. The system analysis shows the first insights into the tasks strategic information management, project management, and change management, all of which are relevant in Part II and Part III of this thesis. (ii) Second, a survey of 134 CIOs, which will be presented in Section 4.3. This survey reveals details of the CIO role and its embedding in the hospital organization, and helps to understand the responsibilities, communication needs and details about the work of the CIOs. Both the system analysis and the survey are summarized in Section 4.5.

4.1 Research Questions

The goal of this domain study is to provide first insights into details of the IM of two hospitals. In order to understand this application domain in greater detail, we need to investigate the IM concepts *function*, *role* and *entity type*, as introduced in Section 2.4 on page 29. First, we investigate the *tasks* of the IM department which the CIO is involved in, and not only those which are limited to one specific hospital. Thus, we raise RQ.DS.1. Second, the *role* of the CIO has relations to other management roles of the hospital, such as the hospital's board of directors and the chief executive officer [WHA+11]. A detailed understanding of the CIO's position in the management hierarchy shows the CIO's status and his/her decision power. Thus, we raise RQ.DS.2 to reveal the CIO's position in the hospital management. Third, management tasks involve communication, and communication involves information, which are represented by *entity types*. The investigation of the communication between the CIO and the hospital management in general provides insights into the way the CIO works, his/her responsibilities and communication needs. Thus, we query in RQ.DS.3 the issues in the communication between the CIO and the hospital management. In summary, these RQs contribute to the example of design problem 1:

RQ.DS.1: What are the details of important IM tasks in hospitals which the CIO is involved in?

RQ.DS.2: What is the position of the CIO in the hospital management?

RQ.DS.3: What are the issues relating to the communication with hospital management?

4.2 System Analysis of Information Management in a Hospital

The system analysis was performed in 2015 and provides an overview of tasks in the IM departments of two hospitals, in order to give an answer to RQ.DS.1. CIONx is specific to one of these hospitals, and thus focuses only on this hospital to the following extent. The CIO is responsible or involved in most of these tasks. The system analysis that is presented in this thesis is only an excerpt of a larger system analysis that was carried out. Other aspects of the system analysis included the construction of a three layer graph based model ($3LGM^2$) model, the relation of the investigated tasks to Information Technology Infrastructure Library (ITIL) processes, and interoperability issues. These aspects lie beyond the scope of this thesis and are presented in detail in [Jun15, KJJ+15]. This section is structured in the following way. In subsection 4.2.1 the details of the methodological background which was used to structure the system analysis are presented. Lastly, in subsection 4.2.2 the results for RQ.DS.1 are presented.

4.2.1 Method

The system analysis method described by Ammenwerth et al. [AHKGW15] is used in the domain study in order to follow a structured analysis process. The goal of such system analyses is to investigate and document the as-is state of an IS that is specific to the healthcare environment. Ammenwerth's system analysis comprises four phases, as is shown in Figure 4.1, and allows multiple iterations until all information has been gathered with the defined quality. The system analysis starts with the formulation of a specific information need. This information need is refined in the activity *planning*. The as-is information is gathered in the activity *information acquisition*, then transformed to models in the activity *processing and modeling*, and finally verified in the activity *verification*. The result of the system analysis is an analysis report which fits the initially formulated information need. The results of the

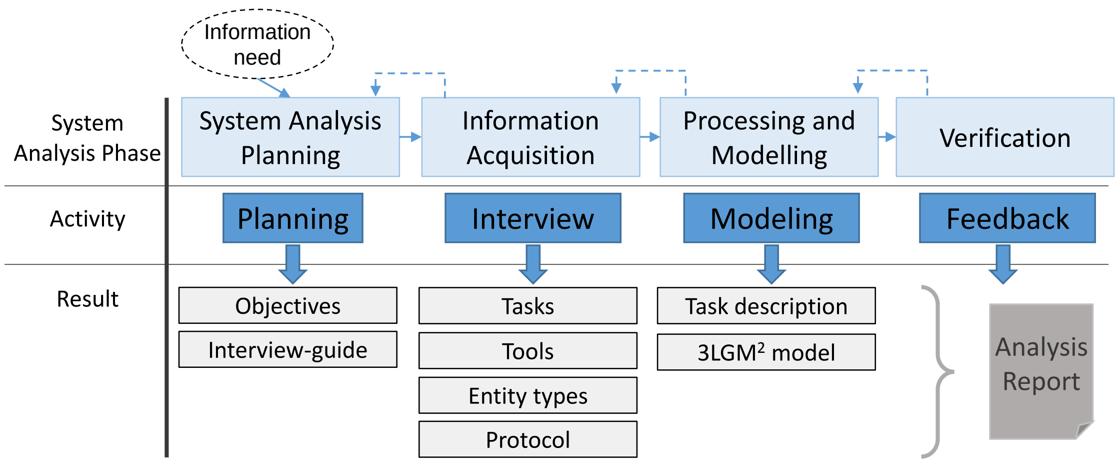


Figure 4.1: The system analysis according to [AHKGW15] and the results of each phase

system analysis are well suited to be documented as $3LGM^2$. This section describes our instantiation of this method for the specific hospital X.

4.2.1.1 Phase 1: System Analysis Planning

In this planning phase, the information need is first refined into the objective, and then the detailed planning of the system analysis including the scheduling is accomplished. The objective (O1) of this analysis is: to elicit the as-is-state of the tasks, their related entity types and the supportive tools within the IM of hospital X. Ammenwerth et al. distinguish between different system analysis types. Motivated by the objective, we use a *task analysis* to discover and analyze the IM tasks. Additionally, we use a *process and activity analysis* to identify input and output entity types of tasks. By means of an *equipment analysis*, tools that support the execution of tasks are identified.

4.2.1.2 Phase 2: Information Acquisition

The methods of information acquisition are not limited by Ammentwerth et al., but the oral questioning including interviews, the written survey, the observation and the data or document analysis are all mentioned as examples. We use *interviews* as the information acquisition technique for the following reasons [AHKGW15]: (1) interviews promise to deliver good results for the objectives; (2) they represent a reasonable time investment; and (3) they allow preparation and flexibility during execution. Observations are not appropriate due to the time limitation of staffs and the sensitivity of the IM division. Data analysis is also not

Table 4.1: Interviewees in hospital X

Interviewee	Position/Role	Type
Person 1	CIO	face to face
Person 2	Project Manager / IT Strategy	face to face
Person 3	Assistant IM	face to face
Person 4	Application Manager	face to face
Person 5	Service Manager IM	face to face
Person 6	System Manager IM	phone
Person 7	Assistant IM	phone

Table 4.2: Questions of the interviews

Goal	Question	Metric
O1	Which tasks are within your responsibility?	m1: tasks
	Who else is in charge of this task?	m2: contact persons
O1	Which input and output information is processed?	m3: entity types
	Which interfaces and communication type are used?	m4: interfaces
O1	Which applications are used to support the tasks?	m5: tools

appropriate, since there is no explicit data available about tasks, entity types and the tools used. Written surveys would work well to gather an exact description of tasks, but they normally address larger user-groups.

In a first conversation with the CIO, we identified seven persons who are responsible for IM-tasks and selected them as interviewees. Interviews of several roles allow a comprehensive view of the IM's tasks, entity types and tools. The roles include the CIO, several department managers, a project manager and assistants, as are shown above in Table 4.1. We conducted one interview per person who was associated with one single role, and we used the same interview-guide for all interviews. Five interviews were face to face within a four-week timeframe and two were been held over the telephone.

The questions presented above in Table 4.2 were used in the interview. This table also shows the relations of the questions to the objectives and the envisioned results. In the interview that had a maximum duration of 1.5 hours, the interviewer led the interview with the interviewee and up to two additional persons recorded the answers. The interviewer was also taking notes to write a protocol.

4.2.1.3 Phase 3: Processing and Modeling

Immediately after the interview, the protocols of the minute-taker were merged with the interviewer's protocol into one protocol. The task-related information that was obtained was then transferred and assembled into a detailed task description for each identified task, as is presented in Figure 4.2 by way of example for the task change management. An overview of all IM tasks has also been created from these task descriptions, and is categorized into strategic, tactical and operational IM, as is presented in Figure 4.3. The system analysis results in a comprehensive task description and a hospital-specific 3LGM^2 model of IM [Jun15].

ID: A-002 **Date:** 30.03.2015	**Task:** Change Management **Version:** 1.2
Objective	Collection of changes
Possibility of intervention	edit request, estimation of change
Trigger	user's request
Priority	low / ~~high~~ / ~~medium~~
Initial situation	completed request form, eligible change rationale
Information In	change request form
Information Out	change report, change appraisal, changed component
Resources	staff of operational IM, change advisory board, division manager, possibly department manager
Application Systems	Microsoft Office (Word, Outlook), Microsoft SharePoint
Interfaces	Exchange server, SharePoint server

Figure 4.2: Task template according to [PK04], describing the task change management

4.2.1.4 Phase 4: Verification

The condensed protocol was then sent by e-mail for review to the interviewee. The protocol did, however, not contain the task description and the 3LGM^2 model. Four of seven interviewees gave feedback to the protocol with corrective comments. The final task description and updated 3LGM^2 models were presented to the CIO.

4.2.2 Results

We have identified 13 tasks that are supported by 47 different tools.

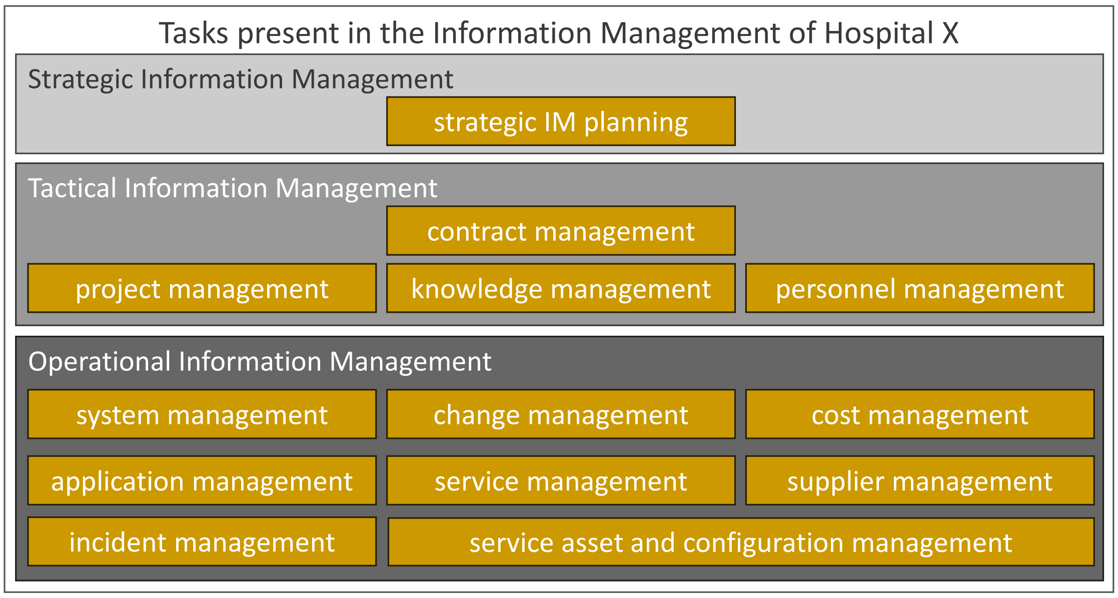


Figure 4.3: The identified tasks associated with strategic, tactical and operational IM

4.2.2.1 Identified Tasks in Information Management

An overview of the tasks which are categorized into strategic information management, tactical information management and operational information management is shown in Figure 4.3.

The only task of strategic information management is strategic IM planning. The tactical information management comprises four tasks: contract management, project management, knowledge management, and personal management. The operational information management comprises eight tasks: system management, change management, cost management, application management, service management, supplier management, incident management, and service asset and configuration management. We described all of these tasks, together with their subtasks, in a detailed task description. As an example, we show the results of the task change management in the following subsection.

4.2.2.2 Identified Subtasks and Entity Types of Change Management

Table 4.3 shows the identified subtasks of change management together with the consumed and modified entity types of each task. Whenever a user needs a new or changed functionality, s/he creates a `change request` using a web-based form in subtask T-003: record information about change. The details of a change request are stored in an entity type `change request`. The modification of `change requests` are performed by users in subtask

T-004: modify information about change. After the `change request` is received by the IM department, members of the change advisory board check its necessity in subtask T-005: process the change request. When a change is found to be necessary, a `change report` with the comment about the requested change is created. The same entity type `change report` is produced in subtask T-006: maintain systems, which specifies general changes during the monthly service maintenance. A message about the assessed `change request` is sent by the change advisory board to the users in subtask T-008: notify user about change. The `change request` form is now saved for division-wide access in subtask T-007: store a change request form. If the `change request` is accepted by the change advisory board in the form of a `change appraisal`, the change is implemented within the subtask T-009: implement change. The entity type `change appraisal` contains the appraisal of all department managers who state whether this change request should be realized. As soon as the change is available in the operational environment, its impact is assessed in the subtask T-010: evaluate change.

Table 4.3: Subtasks and entity types of task change management

ID	Subtask	Entity types consumed	Entity types modified
T-003	Record information about change	–	change request
T-004	Modify information about change	change request	change request
T-005	Process the change request	change request	change report
T-006	Maintain systems	–	change report
T-007	Store a change request form	change report	–
T-008	Notify user about change	change report	change appraisal
T-009	Implement change	change appraisal	change request
T-010	Evaluate change	change request	–

4.2.2.3 Identified Tools of Change Management Process

The identified subtasks of change management are supported by the collaboration platform Microsoft SharePoint Server, the office tools Microsoft Word and E-Mail communication conducted through Microsoft Outlook. The mapping between task and tool is summarized in Table 4.4.

Table 4.4: Tools supporting subtasks of change management

Tool	Subtask
Microsoft SharePoint	Record information about change (T-003), Modify information about change (T-004), Maintain systems (T-006), Store a change request form (T-007)
Microsoft Word	Process the change request (T-005)
Microsoft Outlook	Store a change request form (T-007), Notify user about change (T-008)

4.3 Survey of Information Management in German Hospitals

IM departments in hospitals provide IT services which support both patient care and administrative hospital functions. The complexity and the impacts of IT failures are continuously increasing. Therefore, professional IM departments are necessary, even though their importance is commonly underestimated [MSM08, LKH08]. Little is known about the organization and variety of IM departments in German hospitals. Therefore, within the SNIK project, a survey was undertaken to characterize their capabilities. Two parts of this survey influence this thesis: (1) the CIO's position in the hospital management hierarchy; and (2) communication of the CIO with the hospital management. The survey and its results shown here were published in [KLS+16] and were developed in cooperation with the SNIK project team, of which the author of this thesis was a member.

4.3.1 Methods

The evaluation is based on an online survey of 134 CIOs, who completed their questionnaire in its entirety. The survey questions were developed according to application domain-specific literature, the SNIK ontology, and interviews with domain experts, and were then improved by a pretest. The study is designed as a cross-sectional study and comprises 59 questions[1]. The questionnaire also contains questions for other goals, which are however not focused in this survey excerpt. In the survey N=1284 CIOs were invited by e-mail to participate in the online survey. The data from the online survey was collected between February 12, 2016 and the beginning of April 2016. The survey resulted in 176 analyzable questionnaires that were at least half-completed, resulting in a response rate of 13.7 %. We used the questionnaires that were at least half-completed for the analysis of the questions where $n > 134$. The number of answers that were considered is indicated as *n* and the standard deviation as σ for each evaluated question. The question number in the questionnaire is indicated by Q.

[1]The online survey questions are available at http://www.snik.eu/de/Ergebnisse/fragebogen2016.pdf.

4.3.2 Results

The CIO's Position in the Hospital Management Hierarchy: The survey questions Q18, Q3, Q6[2] are used to answer RQ.DS.2, which focuses on the position of the CIO in the hospital management hierarchy. We define this position depending on the CIO's inclusion in the hospital management, the job description, and the number of subordinated hospitals. Slightly more than half of the CIOs (54.9 %, n=95) are responsible for a single hospital (Q3, total n=173), whereas all others are responsible for more than one hospital (45.1 %, n=78). Those CIOs who are responsible for more than one hospital take care of 3.97 hospitals on average ($\sigma = 6.394$, n=78). All participants were requested to name their job description (Q6 with free text, total n=170), which show a great variety. We derived the CIO's position in the management hierarchy of the hospital from their given job description. The keywords *leading*, *leader*, *manager*, and *head of* each indicate an executive role and are present in 82.9 % of the job descriptions. Interestingly, only in 3.5 % of the job descriptions is the term CIO mentioned explicitly. A large majority of 94.4 % (n=152) of the CIOs are not members of the management (Q18, total n=161).

In summary, we can answer RQ.DS.2 by saying that in most cases the CIO is not a member of the hospital management, but in the majority of cases the job description reveals an executive status of the CIO. Slightly more than half of the CIOs are responsible for one single hospital, and in all other cases they are responsible for up to four hospitals.

Communication Issues with Hospital Management: The survey questions Q40, Q49, Q55 answer RQ.DS.3 and investigate the kind of information that the CIO uses in his or her communication with the hospital management. From the used information, and in particular from the used IM entity types, we have come to understand the communication issues. Figure 4.4 shows an overview of the used strategic, tactical, and operational information and their respective frequencies. In order to make the survey completely relevant to this thesis, we consider *occasional* and *frequently used* as one group. Regarding strategic information management, the most commonly used information are finances, security concept, critical IT services, hospital processes, IT-relevant hospital targets and IT-strategy. We argue that the latter two categories of information directly relate to the user task strategic IM planning, as is shown in Figure 4.3. Regarding tactical information management, the most commonly used categories of information are project status and project portfolio. Regarding operational information management, the most commonly used categories of information are IT costs and capacity plan (n=137).

[2]The questions Q3 and Q6 offer free text fields.

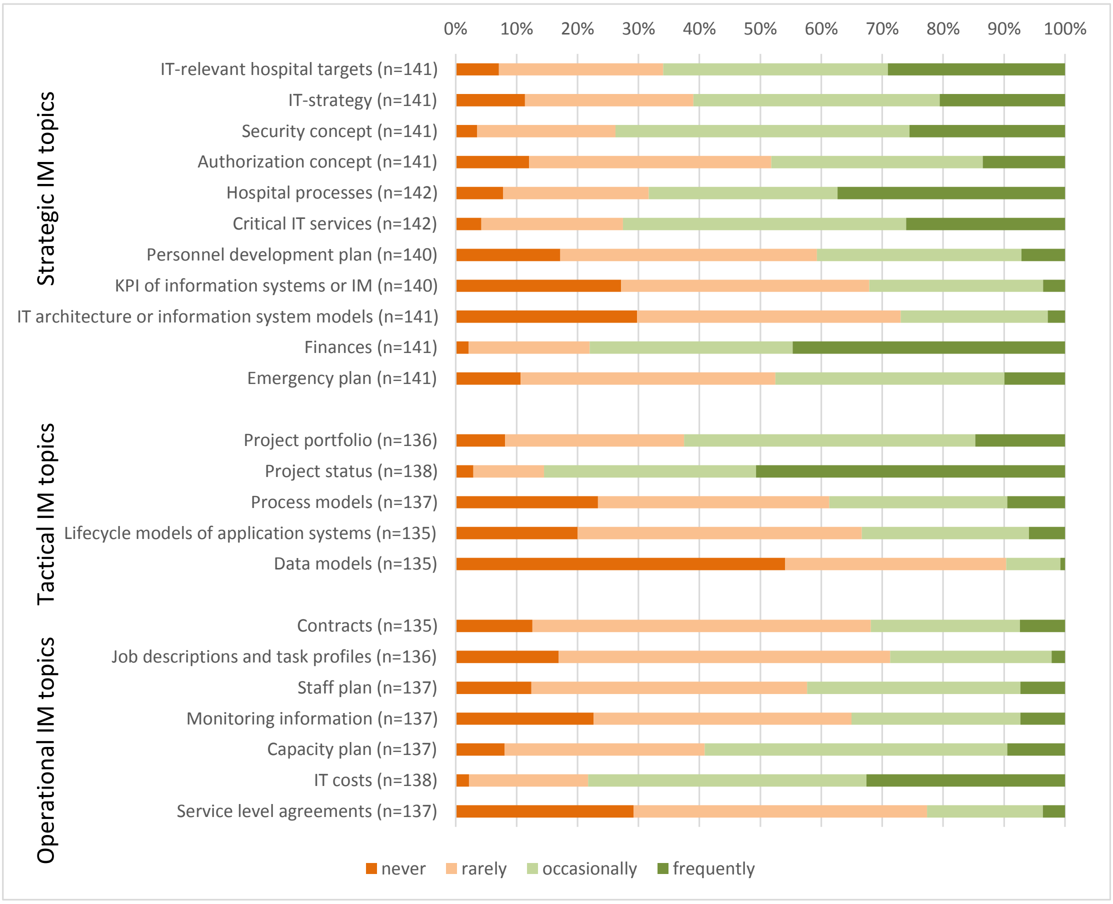


Figure 4.4: Used strategic, tactical, and operational information for CIO communication

In summary we can answer RQ.DS.3 by stating that the major issues in communication are related to projects, finance, security, critical IT and hospital services, and IT-relevant hospital targets.

4.4 Selection of Tasks for Further Investigation

From the tasks presented in subsection 4.2.2, the CIO has chosen in the course of a project meeting four tasks for further investigation. The primary purpose of the IM department is to provide IT services and not to exceed the allocated budget. Thus, the following tasks are particularly important for the CIO: (i) project management, (ii) change management, (iii) service management, and (iv) cost management. In consultation with the CIO, we prioritized the tasks depending on their importance and selected the first two tasks project management and change management for further investigation in Chapter 5 of this thesis.

The task project management is prioritized on the grounds that project management is very important for the CIO in order to direct the IM department, and financial aspects are included in this task. The task change management is selected since changes impact upon IT service operations and potentially cause harm to successful service delivery. The selected tasks cover tactical information management and operational information management, but leave out any task relating to strategic information management. Therefore, we have added the task strategic IM planning to the case study Chapter 7. In summary, we investigate a task from each of the three IM differentiations, namely strategic information management, tactical information management, and operational information management.

The survey's results regarding important information in the hospital management communication (RQ.DS.3) confirm the importance of the two selected tasks, namely strategic IM planning, and project management. The system analysis results (RQ.DS.1) confirm the relevance of the three tasks selected by way of examples, not only for the selected hospital X but also for other hospitals. In focusing on these tasks, we have chosen an important area for the RE specification of the CIONx tool, which is not limited to one hospital, and which has a high relevance to the CIO.

4.5 Chapter Summary

System Analysis: In this chapter, we performed a system analysis according to Ammenwerth et al. in order to analyze and assess the tasks, entity types and tools used within the IM in one hospital. The system analysis provides an overview of 13 tasks in IM in which the CIO is directly involved, answering RQ.DS.1. As one example, the task change management was described in greater detail, covering the used entity types and tools. **Survey:** The survey shows that although the CIO is mostly not a member of the hospital management, the job description reveals an executive status of the CIO (RQ.DS.2). Important categories of information in the communication between the CIO and the hospital management (RQ.DS.3) are IT-relevant hospital targets, IT strategy, project status and project portfolio, and finances.

5 Problem Investigation

This chapter presents the problem investigation as a case study to investigate the extent of TORE's support for the RE specification of PDSSs. Section 5.1 describes the design and data collection of the case study. Section 5.2 describes the results. Section 5.3 presents the threats to its validity. Finally, this chapter is summarized in Section 5.4.

5.1 Study Design

The research objective of this case study is to identify the missing decision-specific information in TORE in order to support the specification of a PDSS. Thus, we want to answer RQ.1.1: *What decision-specific information is missing in TORE to support the specification of a PDSS?* In particular, by *decision-specific information*, we mean such information that is necessary in order to make a decision. By *missing in TORE*, we mean such requirements which are as of now inadequately represented in TORE's DPs.

We investigated the RE process for the specification of CIONx, that was explained in Section 2.5 on the basis of TORE (see Section 2.2). Therefore the activities shown in Table 5.1 were carried out. The supported stakeholder (DP Supported Stakeholder) is the CIO of the particular hospital X. Activity 1 refers to the task analysis described in the domain study, that was presented in Chapter 4. The knowledge gained in activity 1 about the IM tasks provides the basis for activity 2. In activity 2, we performed a semi-structured interview with the CIO to comprehend the CIO's tasks and the goals of CIONx. This interview followed the questions that are presented in Section A.1. We investigated the tasks $\mathrm{project\ management}$, $\mathrm{service\ management}$, $\mathrm{change\ management}$, and $\mathrm{cost\ management}$, that constituted 4 out of the 13 identified tasks from the domain study in Section 4.2. The resulting artifact was a task description according to Lauesen's Task & Support, which provides a detailed description of these four tasks. In consultation with the CIO, we prioritized these tasks on the basis of their importance and selected the tasks *project management*, and *change management* for all further investigation. The task description and the meeting's minutes were reviewed by the

Table 5.1: Overview of the case study's activities, duration, and effort to create or modify the artifacts (indicated by ↳). Duration in [h:mm]

ID	Duration	Activity	**Decision Point** / Description
1	-	Task analysis	Stakeholders' Tasks, Domain Data (Domain study)
2	2:15	Interview	Supported Stakeholders, Stakeholders' Goals, Stakeholders' Tasks, As-is Activities, Domain Data
↳	16:00	Engineering	Artifacts: Task & Support task description, goal list
3	-	Document analysis	based on screen-shots
4	-	Document analysis	operative files

CIO. During a document analysis in activity 3 based on screen-shots, we analyzed several spreadsheets used by the CIO for decision-making to understand the decision-specifics. These screen shots contained the majority of data that is necessary for the investigated task project management and change management. In an additional document analysis in activity 4 based on operative files, we analyzed further decision-specific documents in the form of spreadsheets (see Figure 5.2) and text-documents. The document analyses helped us to understand the decision-specific data in particular. We did not visit the other DPs, as the insights from these activities already provided a good picture of the decision-specific requirements.

5.2 Results

The problem investigation shows that TORE's DPs can capture a large amount of the relevant requirements gained during the interview. However, some details of the important information about the CIO's decisions cannot be captured explicitly with the existing artifact types and DPs in TORE, as is explained in the following. These five decision-specific pieces of missing information are presented below as criteria c_1-c_5. We use these criteria to shape the treatment design, as will be presented in Chapter 6.

5.2.1 Distinction between Decisions and Subtasks

$\mathbf{c_1}$: *Distinction between decisions and conventional subtasks*. We found subtasks with and without decision character. Some subtasks contain a decision problem as a recurrent choice

ID: Subtask	PM 3a: **P**rioritize projects of `project list`
Actor	CIO
Contribution	Consider important projects in budget planning.
Cause	Next fiscal year starts in two months.
Description	Decision about order of new projects for budget planning before a project can start. The assigned priority depends on its contribution to...
Pre condition	All project proposals for next fiscal year are available. The CIO knows the goals and necessity of projects.
Info-In	Urgency, proposed priority of proposer (project proposal), project list, controlling-list, remaining budget (difference calculation).
Post condition	Changed priorities in project list. Budget is transferred to new project.
Info-Out	Priority of new project. Re-prioritization of existing projects in project list.

Figure 5.1: Subtask project prioritization in standard template

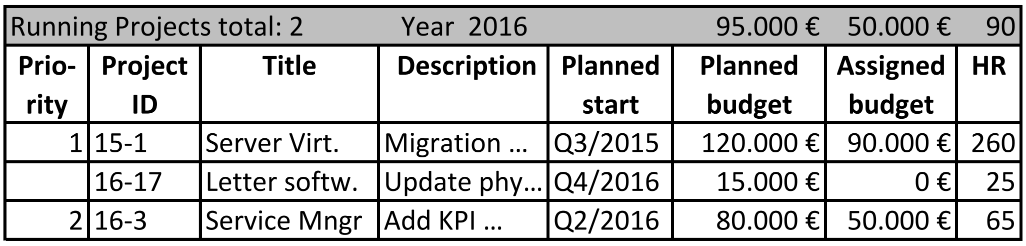
between several alternatives, each of which entails different implications. Such subtasks are particularly relevant for the PDSS requirements specification.

5.2.2 Decision-specific Data

c$_2$: *Detailed description of the data necessary to make the decision.* Based on the information from the pre-assessment and interview, we created the subtask description prioritization of projects as a subtask of the task project management, as is shown in Figure 5.1. The documentation with a standard subtask template does not show explicitly the data required for a decision. The domain data model shows the entities, but not the relation between the entities and decisions. By way of an example, Figure 5.1 shows two deficits. It cannot be explicitly documented (a) how the remaining budget is calculated, that is which entity-attribute pairs are used for the calculation, and (b) which information is exactly necessary from the `project list` and the `controlling list`.

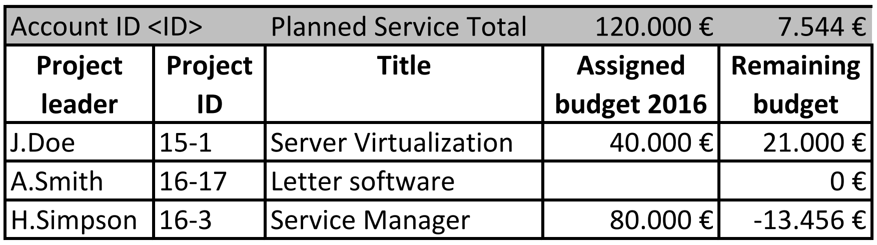
5.2.3 Decision-specific Rules

c$_3$: *Decision-specific rules to choose from alternatives.* Decisions always contain alternatives (see Section 3.3 on page 39). In some cases, alternatives can be chosen on the basis of predefined rules. In order to understand decisions and their alternatives, it is essential to emphasize rules, if there are any. For example, the CIO assigns a high priority to new projects when they contribute to service protection, a medium-level priority when the project concerns



Running Projects total: 2			Year 2016		95.000 €	50.000 €	90
Prio- rity	**Project ID**	**Title**	**Description**	**Planned start**	**Planned budget**	**Assigned budget**	**HR**
1	15-1	Server Virt.	Migration ...	Q3/2015	120.000 €	90.000 €	260
	16-17	Letter softw.	Update phy...	Q4/2016	15.000 €	0 €	25
2	16-3	Service Mngr	Add KPI ...	Q2/2016	80.000 €	50.000 €	65

(a) Part of `project list`



Account ID <ID>		Planned Service Total	120.000 €	7.544 €
Project leader	**Project ID**	**Title**	**Assigned budget 2016**	**Remaining budget**
J.Doe	15-1	Server Virtualization	40.000 €	21.000 €
A.Smith	16-17	Letter software		0 €
H.Simpson	16-3	Service Manager	80.000 €	-13.456 €

(b) Part of `controlling list`

Figure 5.2: Decision-specific spreadsheets

service operation, and a low priority to all other projects. This rule is not made explicit in the standard template for task descriptions, such as in Figure 5.1.

5.2.4 Decision-specific Patterns in Data

c$_4$: *Support of decision-specific patterns in data for tables and summary fields.* The document analysis has shown two recurring patterns in the data that is required by the CIO for decision-making. First, there are combinations of entity/attribute pairs from more than one single entity. Second, computed data such as sums or any other kind of condensed data is evident. For instance, the CIO prioritizes new projects which might change the priority of existing projects. Currently, s/he requires two documents to decide about the priority of new projects. First, the spreadsheet `project list`, illustrated in Figure 5.2a contains all ongoing, completed, and planned projects. Each project is specified with a title and priority, a planned start date, the estimated budget and Human Resources (HR) effort, and an approved budget. In the top row, there is the number of projects and the sum of planned and assigned budgets for projects running in the year 2016. Second, the `controlling list` which contains an overview of all projects' budget statuses as is shown in Figure 5.2b, on the basis of each project's expenses. This list contains the project's leader, ID and title, as well as the assigned budget in 2016 and the remaining budget. Again, there are single rows and summary fields in the header, such as the total assigned budget in 2016 and the total remaining budget (`unused_budget`) of all projects. It is important to describe and model explicitly these decision-specific data patterns (beside the UI prototypes), in order to be able to aggregate all data in one model and check for dependencies and consistencies. The domain data model does not support the modeling of data combinations and aggregations from different entities.

5.2.5 Data sources

c_5: *Relation between content, format, and location for data sources.* Executives work with spreadsheets containing aggregated data (for example, Figure 5.2) and some text documents. Their content and format are both known to frequent users. Spreadsheets and text documents represent heterogeneous decision-specific data sources. In order to relate the patterns in data (c_4) with their origin, we need to document and understand these data sources. These heterogeneity is specific to PDSSs (see subsection 3.4.3). Thus, a detailed description is an important input for system design decisions. Such a description should contain the content in terms of entities, the format and the location. Locations of data sources are identified and specified in IS typically with Uniform Resource Identifiers (URIs)[1]. A URI points to an abstract or physical resource, such as a file or a website. It contains besides an address the access method, such as Hypertext Transfer Protocol (HTTP) or file access.

5.3 Threats to validity

The threats to validity are structured according to Runeson et al. [RH09]. **Construct validity** considers whether the study measures what it claims. The investigated tasks contain decision-making tasks of the CIO. The documentation of the requirements was performed with unmodified requirements artifacts, which are typically used in TORE. The results might change with other investigated decision tasks. We mitigated this possibility of change by investigating the four distinct tasks, project management, change management, service management, and cost management on two different levels of IM. These tasks contain several different decision problems. **Internal validity** considers causal relations of investigated factors, that is to say effects of unknown factors that influence an investigated factor. We see the threat that the requirements engineer applied some of the artifacts for the first time, and thus had less experience with the RE artifacts. We mitigated this by two activities: (1) the continuous review of the RE artifacts between the requirements engineer and the supervisor of this thesis; and (2) the verification with current scholarly literature of the intended use and detailed structuring of the RE artifacts and templates used. **External Validity** describes the

[1]This is described by IETF RFC 3986 `https://tools.ietf.org/html/rfc3986`.

generalizability of the findings and the transfer of study results beyond the study. The case study is based on a single case only with four tasks, and it involved only one person. The results change with other types of decisions, for example unstructured decisions, or decisions with unknown alternatives and a high level of uncertainty. **Reliability** considers the influence of the specific researcher and indicates threats to validity for a repetition of the study. The main threat to reliability is that the author of this thesis in his role as requirements engineer had a large influence on the development and process of the case study, in particular during the interviews and the data analysis. We mitigated this through continuous discussion with the supervisor of this thesis.

5.4 Chapter Summary

The domain study presented in Chapter 4 provides the basis for this case study and helped in our selection of the investigated tasks. The problem investigation that was presented in this chapter as a case study consisted of one interview, the creation of TORE artifacts for the DPs *Supported Stakeholders*, *Stakeholders' Goals*, *Stakeholders' Tasks*, *As-is Activities*, *Domain Data*, and two document analyses. In order to support the particular system domain of PDSSs, TORE needs explicit support for the following decision-specific information, which is the answer to RQ.1.1: c_1: Explicit distinction of decisions from conventional subtasks; c_2: Detailed description of data necessary for decision-making; c_3: Decision-specific rules to choose from decision alternatives; c_4: Support of decision-specific patterns in data for tables and summary fields; and c_5: Relation between content, format, and location for data sources.

6 Treatment Design - A Decision-specific Extension of TORE

This chapter introduces the design of the DsTORE method as a decision-specific (Ds) extension of the task-oriented RE method TORE. A brief overview of existing RE methods for DSS is given in Section 6.1 with the intention of differentiating TORE from other DSS RE methods. The result of the treatment design is the DsTORE-method extension that satisfies the requirements given by the answer to RQ.1.1.

6.1 Related Work

This section provides a brief overview of RE methods for DSS in order to show how TORE relates to other approaches. Garcia et al. [GRR16] present a comprehensive mapping study of 27 articles related to RE for DSS. Although the title of their study suggests DSSs, the search terms and the identified literature both focus on DW. The authors conclude that there is a gap in the process of creating the design of a DSS in a methodical manner. In particular, the business needs do not receive sufficient attention during the RE process. They propose to elicit business needs from stakeholders by asking them the question "What do you do and why?". TORE exactly addresses this kind of question. A description of the activities of stakeholders (i.e. a description of what they do) is captured in TORE's DPs *Stakeholders' Tasks* and *As-is* and *To-be Activities*. Business needs and the rationale why stakeholders perform some tasks are considered in DP *Stakeholders' Goals*. Another overview by Salinesi and Gam [SG09], which is not covered by Garcia et al., provides a comprehensive discussion of 15 RE approaches for DSSs, all of which are DW-specific. They found three families of methodological processes: (1) Data-driven approaches, which are focused on the structure of data sources to design multi-dimensional schemas. However, these approaches do not cover decision specifics. (2) Requirement-driven approaches, which focus on decision-makers' requirements. The weakness of these approaches is that the availability of data sources is not treated sufficiently, with the result that user requirements may not be realizable; and

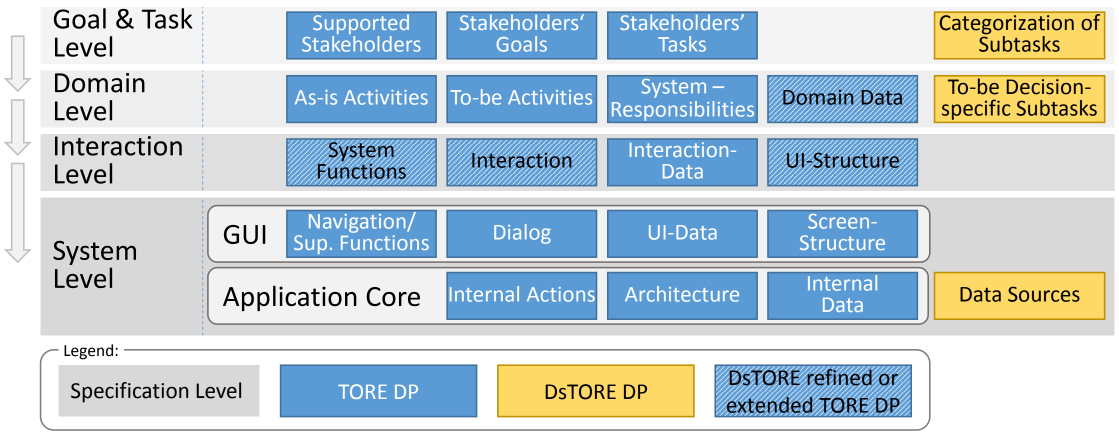


Figure 6.1: Decision Points of DsTORE

(3) Mixed-driven approaches, which are a combination of (1) and (2). TORE supports both aspects with data-related DPs together with the focus on tasks and goals in the DPs *Stakeholders' Goals* and *Stakeholders' Tasks*. Data-related DPs are in particular *Domain Data*, *Interaction-Data*, and *UI-Data*. In addition to the three types of methods, TORE supports the consideration of UI details. Both overviews confirm the need to understand the needs of businesses and stakeholders. Since the approaches of both overview papers focus on DWs, they support in particular the specification of DW data schema, data marts for DWs, and DW-specific data transformation processes, which are not important for PDSSs. The presented RE approaches do not provide a detailed task-oriented set of guidelines for the development of PDSS.

There are several goal-oriented RE approaches for DSSs. Again they all focus on DWs. These approaches share the importance of the business and decision focus, but also place emphasis on decision specifics for DW, such as key performance indicators [PJR^{+}14] or business strategy and indicators [BTM12, TB15]. Giorgini, Rizzi and Garzetti propose with GRAnD [GRG08] a goal-oriented method which connects a decisional model with a source schema. While this method provides a detailed analysis of the goals and decisions, it does not provide guidance for the UI design and early prototyping with the user.

In summary, we did not find a RE method for DSSs in general and PDSSs in particular, which provide guidance for a seamless transition from the business level to UI details. A guidance that covers all aspects discussed above is provided by TORE in a flexible manner.

6.2 Overview

This section presents DsTORE as system domain-specific TORE adaptation. The design of DsTORE addresses the decision-specific criteria described in Section 5.2. In this way, DsTORE extends TORE to provide the artifact types and notations that make up the DSML for PDSSs. Thus, the emphasis is to provide a more precise RE. We first introduce new decision-specific DPs with related artifact types, as are depicted on the right side of Figure 6.1. In some cases, we adapt DPs, as are indicated as dashed lines inside the DPs in Figure 6.1. DsTORE uses the same artifact types as TORE. There are two extended artifact types, namely the subtask description and the domain data model. Additionally, there is a new data source model.

6.3 Goal & Task Level

This TORE level contains the new DP *Categorization of Subtasks* to distinguish decisions from conventional subtasks.

6.3.1 Categorization of Subtasks

The new decision-specific DP *Categorization of Subtasks* fulfills criterion c_1 and enforces the identification of subtasks which *are* decisions. A subtask can be either a *decision* or a *conventional subtask*. Two strategies can be used to categorize subtasks into decisional tasks and conventional subtasks: (1) the presence of alternatives to choose from; and (2) the verb or substantive of the subtask, indicating a decision.

For the IM application domain, we use a non-exhaustive categorization model encompassing seven categories. In the categorization model we distinguish between objects or situations (x) and subjects (y). In other application domains, the categorization scheme needs to be populated differently. Decisions are subtasks that are related to the **Approval** of x, **Evaluation** of x or y, and **Prioritization** of x. Conventional subtasks are related to the Monitoring of x, Documentation of x, Communication of x to y, and Support of x or y. Table 6.1 shows the identified subtasks of project management and their categorization.

Table 6.1 presents all subtasks of the task project management for which the CIO is responsible. The column *category* contains the subtask categorization, as has been explained above. The complete details of all project management subtasks are given in Table D.1,

Table 6.1: The subtasks of task project management

ID	Subtask	Category
1	Support the creation of project definition from change request	Support
2	Add project to `project waiting list`	Documentation
3 Project prioritization		
3a	Prioritize projects on the `project waiting list`	Prioritization
3b	Present `project waiting list` as half-year plan at board of directors	Communication
3c	Re-prioritize the `project waiting list` with the board of directors	Prioritization
3d	Promote and justify project approvals	Communication
4	Nominate a project leader for a project	Approval
5 Project monitoring		
5a	Monitor project execution	Monitoring
5b	Evaluate the project status, end date, target date and project progress	Evaluation
5c	Escalate project internally if necessary	Communication
5d	Escalate project externally if necessary	Communication
5e	Execute half-year review for projects	Monitoring
5f	Evaluate the project's half-year review	Evaluation
5g	Communicate the project's evaluation results to the board of directors	Communication
5h	Evaluate the project's budget	Evaluation
6	Communicate and present the project's results to the board of directors	Communication
7	Monitor the transition phase	Monitoring

which are described with Lauesen's Task & Support template. Figure 6.2 presents an excerpt of this task description as an example. We propose to categorize the subtasks either in a table according to Table 6.1 or as an additional column in Lauesen's Task & Support notation, as shown in Figure 6.2.

A brief explanation of the subtasks in Table 6.1, with a focus on possibly existing decisions, is given in the following section. Some new projects arise from change requests. As each project requires a written project definition, the CIO supports the creation of this document when necessary in subtask 1, which is a *support* task. Requests for new projects additionally arrive at the CIO additionally from sources such as meetings, e-mail and from the board of directors. When such new projects arrive, the CIO adds them to the `project waiting list` in subtask 2, which is a *documentation* task. Once the projects have been added to the `project waiting list`, they must be prioritized and budgeted, since there are more projects than there are human and financial resources. This is performed in subtask 3a, which is a prioritization task. When the prioritization is completed and there is a clear view of the planned projects for the next fiscal year, the CIO presents the `project waiting list` to the hospital's board of directors. This is done in subtask 3b, which is a communication task

ID	Subtask	Subtask Category	As-is Solution	Example Solution
3a	Prioritize projects on the project waiting list	Prioritization	The approval of a project is represented by the assignment of a priority. The project list is a sub-set of the project waiting list. Before a priority is assigned, each project must have a budget and project definition. The project's costs are planned with regard to the team plan and available budget. If a change request has a large scope, it will be realized as a project. When the change request is urgent, the project will be started straightaway without having a waiting phase. As a consequence of this, the project needs an emergency budget.	Using the new DSS, all information for the project's planning are displayed. The data support the definition of the order of execution for all projects. A summary shows the `project list` and `project waiting list` with sorting and filtering. The support of resource-management within the projects would be helpful. The integration of MS-Project is also interesting.

Figure 6.2: The subtask project management 3a according to Lauesens Task & Support

as illustrated in Figure 6.2. In some cases, the hospital's board of directors wants to boost the priority of some projects. Thus, a re-prioritization of projects is performed in subtask 3c, which is a prioritization task. Once this `project waiting list` is finalized and the projects are about to be initiated, the CIO discusses, maintains, justifies and negotiates project approvals. This is documented in subtask 3d, which is a communication task. As soon as the project is listed in the `project waiting list`, it requires a responsible project manager. The CIO nominates an appropriate person with the necessary skills and qualifications for the given project context in subtask 4. As the CIO needs to choose one project manager from a group of staff, this is an approval task. Every two weeks the CIO monitors the progress of the project with the project manager. Since the main focus of this monitoring in subtask 5a is to determine the current project status, this is a monitoring task and not decision task. The current status and the project deadlines are evaluated in subtask 5b. Delayed projects or other obstacles to the project's progression require actions that are decided by the CIO. Since this is a choice of several alternative actions, this constitutes a decision. In subtasks 5c and 5d, the CIO escalates internally or externally to remove possible obstacles, which is a communication task. Within a half-year period, the CIO reconsiders the budget of all projects in the monitoring subtask 5e. In subtask 5f, the CIO evaluates the results of the half-year review. S/he decides about project splits and about the possibility of increasing a project's budget. Thus, this is an evaluation task. In subtask 5g, the CIO reports a summary of the project status and possible budget increase to the hospital's board of directors, which is a communication task. In subtask 5h, the CIO evaluates the budget for investments and the services of the division. Since the evaluation can result in either an escalation of the project or budget rebooking, this constitutes a decision. In subtask 6, the CIO communicates the project results to the project stakeholders or the hospital's board of directors. There is no decision in the communication. In subtask 7, the CIO monitors the transition phase of a finished project to service operations. Also here, there is no decision.

6.4 Domain Level

The TORE domain level contains one new DP *To-be Decision-specific Subtasks*, in order to explicitly consider decisions and document them with an extended subtask template. The DP *Domain Data* is extended to support the patterns of decision-specifics in domain data.

ID: Subtask	PM 3a: Prioritize projects of *project list*		
Category	Prioritization		
Actor	CIO	**Supporting Actors:** project leader	
Contribution	Consider important projects in budget planning.		
Cause	Next fiscal year starts in two months		
Description	Decision about order of important projects...		
Pre condition	All project proposals for next fiscal year are available. The CIO knows goals and the necessity of the projects.		
Combined Data	**Entity**	**Attribute**	
	Project List Table Entry	project.priority	
	Project List Table Entry	project.start	
	...	...	
	Project List Table Entry	budget.planned	
Computed Data	**Entity**	**Attribute**	**Computation**
	Controlling List Table Entry	budget.remaining	assigned budget - sum(project expenses)
	Controlling List	<derived attribute> unused_budget	sum(budget.remaining, all projects)
Post condition	Defined order, assigned priority of new projects. Budget is transferred to new project.		
Info out	**Entity**	**Attribute**	
	Project List Table Entry	project.priority	
Rules	Priority definition by project contribution: (high) service protection; (medium) service operation; (low) nice to have.		

Figure 6.3: Decision subtask project prioritization

6.4.1 To-Be Decision-specific Subtasks

Decision-related subtasks are specified in the DP *To-be decision-specific subtasks*, and are documented with the decision-specific subtask template. The *decision-specific subtask template* extends TORE's subtask template to define the semantics for decision artifacts and fulfills c_2 and c_3. It covers the details of a decision in five structured fields, which are marked in yellow in Figure 6.3. The field *category* contains the category of the subtask, as is classified in the DP *Categorization of Subtasks*. *Combined data* expresses necessary decision- and/or stakeholder-specific data with entity/attribute pairs. Data aggregations, for example time or spatial data, which are often used for decision-making, are documented by *computed data* and are indicated by a forward slash, '/', for the derived attributes in the DDM, as is shown in Figure 6.4. Details of the computations, such as add and subtract, are documented in the column `computation`. The result of the decision is documented in the field *Info-out*, again with the entity/attribute pairs. *Rules* describe which alternative will be chosen by the decision-maker and under which circumstances. The notation can either be textual or formal, as is appropriate.

6.4.2 Decision-specific Domain Data

Decisions require specific domain data which is defined in the extended DP *Domain Data* and which uses additional stereotypes. The entities and attributes need to be consistent with those of the decision-specific subtasks. The decision-specific spreadsheets, as presented in the problem investigation (see Chapter 5), are modeled in the domain data in Figure 6.4 with UML stereotypes. Lists, such as the `project list` (see Figure 5.2a), are modeled by the stereotype «`ListEntity`». The «`ListEntity`» is related to the listed entities and to a «`View`» entity which captures combined attributes. As an example of this, the `project list` (see Figure 5.2a) is modeled in the domain data model in Figure 6.4. Similarly, the domain data model contains the project-specific controlling list `Controlling List Project<ID>`. This list holds the computed attribute `overall cost` which is calculated by adding up the individual expenses of the project. The list rows are modeled by the class «`View`» `Controlling List Table Entry`, which possesses the attributes *supplier* and *expense*. The complete domain data model for the entire task project management is shown in Figure D.6 on page 218.

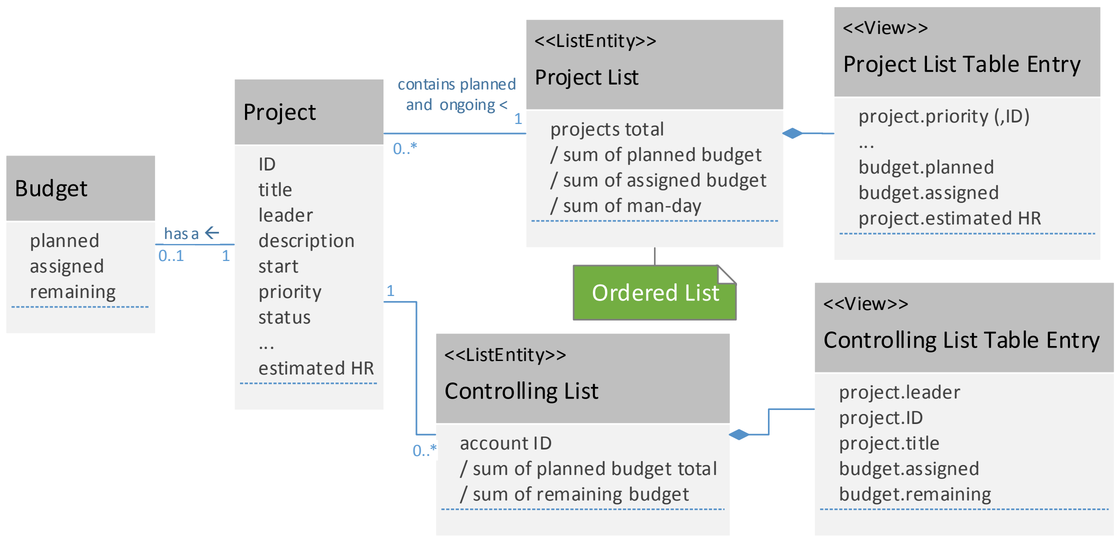


Figure 6.4: Part of domain data model for task project management

6.5 Interaction Level

On the interaction level of TORE, three DPs have decision-specific aspects, namely the DPs *System functions*, *Interaction* and *UI Structure*.

6.5.1 System Functions

In the DP *system functions*, TORE focuses on the to-be system functions. DSSs have a limited range of typical system functions, as are described in subsection 3.4.4. Therefore, we add criterion c_6: the RE specification should consider typical PDSS-specific system functions explicitly. Examples of PDSSs's system functions include the creation of reports, the user's navigation to data sources, and document export. We adapt the DP *System Functions* to use a decision-specific predefined set of system functions as a guidance to support completeness and to avoid gold plating. This set can be adapted according to the different types of DSS.

6.5.2 Interaction and UI-Structure

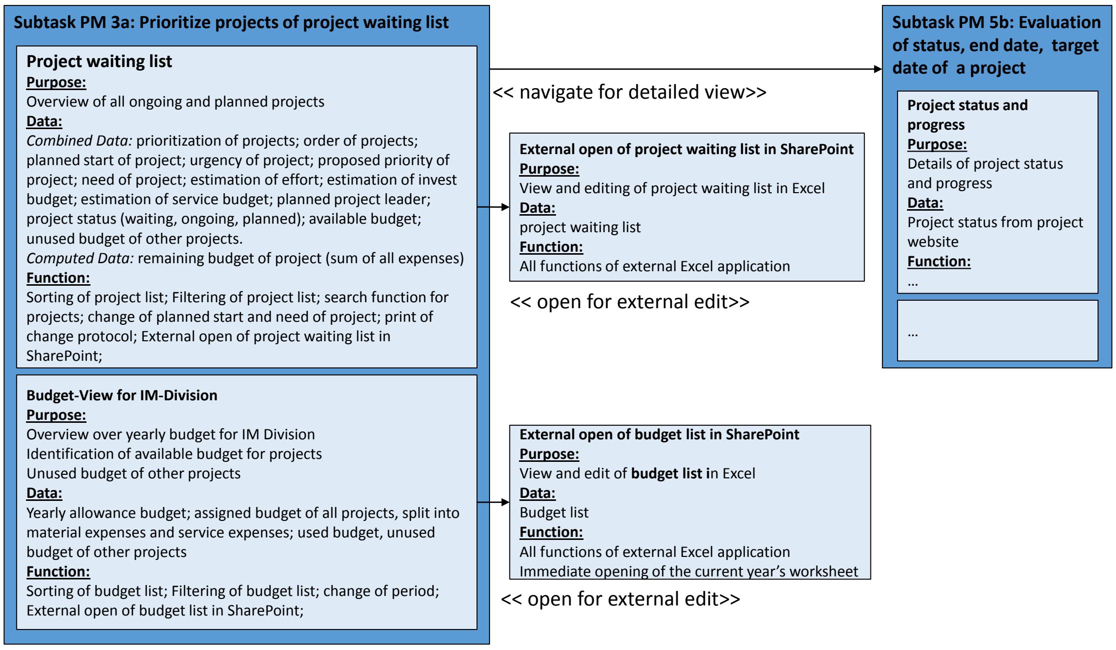

Subtask PM 3a: Prioritize projects of project waiting list
Project waiting list
Purpose:
Overview of all ongoing and planned projects
Data:
Combined Data: prioritization of projects; order of projects; planned start of project; urgency of project; proposed priority of project; need of project; estimation of effort; estimation of invest budget; estimation of service budget; planned project leader; project status (waiting, ongoing, planned); available budget; unused budget of other projects.
Computed Data: remaining budget of project (sum of all expenses)
Function:
Sorting of project list; Filtering of project list; search function for projects; change of planned start and need of project; print of change protocol; External open of project waiting list in SharePoint;
Budget-View for IM-Division
Purpose:
Overview over yearly budget for IM Division
Identification of available budget for projects
Unused budget of other projects
Data:
Yearly allowance budget; assigned budget of all projects, split into material expenses and service expenses; used budget, unused budget of other projects
Function:
Sorting of budget list; Filtering of budget list; change of period; External open of budget list in SharePoint;
<< navigate for detailed view>>
External open of project waiting list in SharePoint
Purpose:
View and editing of project waiting list in Excel
Data:
project waiting list
Function:
All functions of external Excel application
<< open for external edit>>
External open of budget list in SharePoint
Purpose:
View and edit of budget list in Excel
Data:
Budget list
Function:
All functions of external Excel application
Immediate opening of the current year's worksheet
<< open for external edit>>
Subtask PM 5b: Evaluation of status, end date, target date of a project
Project status and progress
Purpose:
Details of project status and progress
Data:
Project status from project website
Function:
...
...

Figure 6.5: The workspace model of subtask project management 3a

We also adapt the DP *Interaction* which is typically described by use cases. Since PDSSs provide a dashboard-like UI with one screen for each decision, a detailed description of

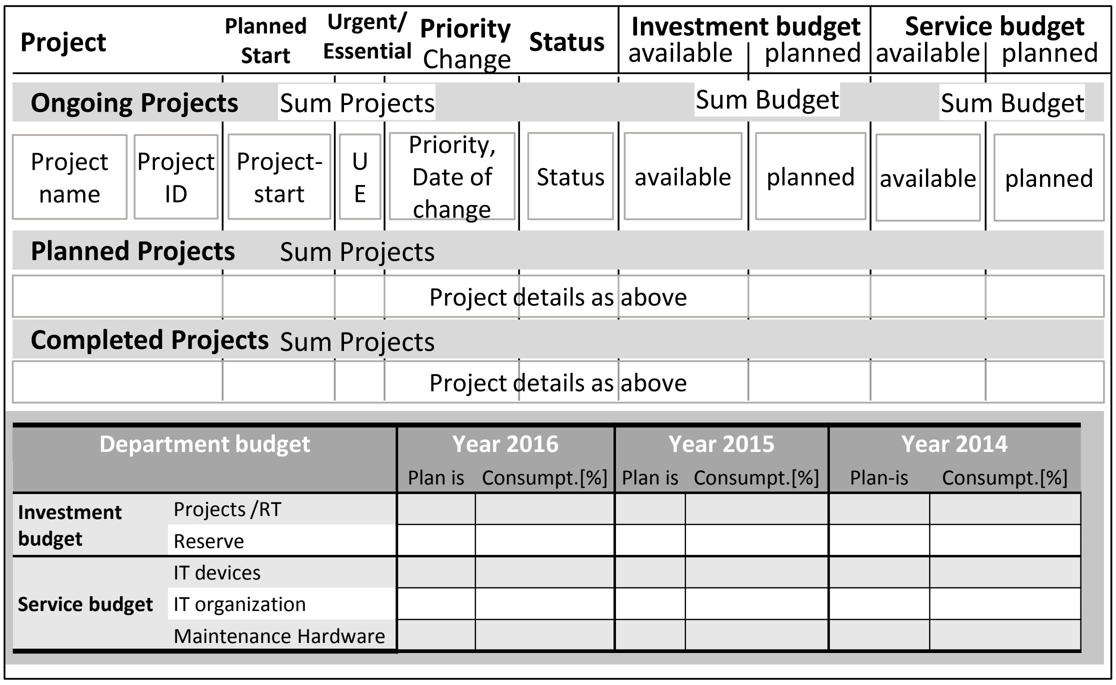


Figure 6.6: The virtual window for project management

interaction on this level is not necessary. The workspace model suffices to capture the navigation between the different workspaces. Figure 6.5 shows as an example the workspace model for the decision task project management 3a.

6.6 System Level

The system level contains one additional DP *Data Sources*. The DP *UI-Data* and *screen structure* is documented with a Virtual Window. Figure 6.6 shows the VW of the subtask prioritize projects of project list, as introduced in Figure 6.3. It also shows the arrangement of the decision-specific subtask's data.

6.6.1 Data sources

On the system level, there is the new DP *Data Sources*. In TORE, data sources are left to software design. As was argued for c_5, the consideration of data sources in DsTORE is important. A data source description can be captured in a simple table with the rows *entities*, *format*, and *location*, as shown in Figure 6.7. All entities in this description are related and consistent with entities in the DP *Domain Data*. The example of the data model shows that the strategic IM plan is a (text-)document located on a specific Uniform Resource Locator (URL) on the SharePoint server.

Entity	Format	Location
project list	excel/csv	http://filesrv1/2016/pwl.xlsx
controlling list project	excel/csv	http://intra.sharepointSrv/2017/controllingList.xlsx

Figure 6.7: The data source model task project management

6.7 Chapter Summary

The generic TORE method can be extended to adequately support details of the RE for PDSSs, which represent a subgroup of information systems. The presented extensions and modification constitute the treatment design to design problem 1. In summary, the TORE extensions can be characterized in the following way. Decision-specific subtasks are distinguished on the goal and task level, with the new DP *Categorization of Subtasks*. Such-decision specific subtasks are then documented in the DP *To-be decision-specific subtasks* with an extended subtask template on the domain level in detail. This subtask template contains both combined and computed information for decision-making, and decision-specific rules to capture decision-specific information. Decision-specific data of spreadsheets contain rows of single information and the related summary fields. Such information can be modeled in the extended DP *Domain Data* using the stereotypes «`ListEntity`» and «`View`». The entities of the domain data directly relate to the decision-specific subtask description. On the interaction level of TORE, there are three modified DPs. The typical features of DSSs, as discussed in subsection 3.4.4, need to be explicitly considered in the DP *System Functions*. The DP *Interaction* is typically described by use cases. A detailed description of the interaction for PDSSs with dashboard-like UIs on this level is not necessary. Thus, the workspace model suffices to capture the navigation between the different workspaces. On the system level, the new DP *Data Sources* emphasize the data sources which deliver the information for decision-making, as is documented by a simple data source model.

7 Treatment Validation - Evaluation of DsTORE

This chapter presents the treatment validation used to evaluate DsTORE in a two-phase case study in order to answer RQ.1.2. Section 7.1 describes the design and data collection of the case study. Section 7.2 describes the results for the RE phase. Section 7.3 presents the results for the system prototype. Section 7.4 presents a discussion of results of the case study, where we also present the lessons that we have learned. In Section 7.5 the threats to validity are discussed. In Section 7.6 the general experiences from the method adaptation are presented. Finally, the chapter is summarized in Section 7.7.

7.1 Study Design

The research objective is to identify the extent to which DsTORE supports the RE of PDSSs. The *object of study* is the application of the DsTORE method. Therefore, we raise and answer RQ.1.2: *How well does the DsTORE-adaptation support the specification of requirements for a PDSS*? The case study is designed in two phases: In **phase one** we re-visit the tasks project management and change management, which have been investigated in the problem investigation, as was presented in Section 5.2, but now using DsTORE artifact types for the specification. Missing information was additionally elicited. In **phase two** we investigate the CIO task IT-Strategy. In the following section, both phases are described in detail.

We used four semi-structured interviews (activities 1-4) to elicit the requirements with DsTORE, as is shown in Table 7.1. The interviews were based on the questions presented in the Appendix in Section A.2. The visited DPs are the following: *Categorization of Subtasks*, *As-is and To-be Activities*, *To-be Decision-specific Subtasks*, *UI-Structure*, *System Functions*, *Interaction-Data*, and *Screen Structure*. Table 7.2 shows which artifacts have been created after the process of elicitation that was carried out during the activities 1-4 and 7-8. For each artifact the time needed for its creation is given, including an internal review and a review with the stakeholder. The task description of the CIO tasks project management

Table 7.1: Overview of the treatment validation activities, duration, and effort required to create or modify the artifacts (indicated by ↳). (T=Telephone), Duration in [h:mm]

ID	Dur.	Activity	**Decision Point** / Description
			Tasks of Treatment Validation, Phase 1: project management, change management
1	1:20	Interview	Categorization of Subtasks, To-be Decision Specific Subtask, UI-Structure
↳	56:00	Engineering	Artifacts for activity 1: task description, decision-specific subtask, workspace model, virtual window
2	1:20	Interview	System Functions, Interaction-Data
↳	15:00	Engineering	Artifacts for activity 2: function description, domain data model
3	T1:30	Interview	System Functions, Interaction-Data, UI Data
↳	23:00	Engineering	Artifacts for activity 3: workspace model, domain data model, virtual window
4	1:45	Interview	To-be Activities, Domain Data, Screen-Structure
↳	12:00	Engineering	Artifacts for activity 4: UI prototype, workspace model, virtual window
5	–	Development	Development of system prototype (~12 months, 40 hours per month)
6	1:00	Presentation	System prototype in software
			Tasks of Treatment Validation Phase 2. Task: IT strategy
7	1:30	Interview	Stakeholders' Tasks, Categorization of Subtasks, To-be Decision-specific Subtasks, Domain Data, Interaction-Data
↳	3:00	Engineering	Artifacts for activity 7: task description, decision-specific subtask, domain data model
8	1:30	Interview	To-be Decision-specific Subtasks, Interaction-Data
↳	4:00	Engineering	Artifacts for activity 8: decision-specific subtask, domain data model
9	T0:45	Evaluation	Evaluation of interviews, artifacts, system prototype
10	1:00	Evaluation	Interview with CIO for evaluation of system prototype based on questionnaire (see Section A.2)
11	–	Evaluation	Questionnaire for the resultant activities triggered by RE (see Section A.5)

and change management were used from the problem investigation. All other artifacts were newly created. Two DPs were not relevant. First, the DP *Navigation/Supporting Functions*, since no supporting functions were necessary. Second, the DP *Dialog*, since the dashboard does not support complex dialogues. All system DPs were refined during the implementation of the system prototype in activity 5, in which two students implemented a system prototype in .NET which integrates into SharePoint. The system prototype realized the support for the CIO for the tasks project management and change management. Due to resource constraints, the system prototype was not extended to support the task IT-Strategy. This system prototype was presented to the CIO in activity 6. In **phase two** we investigated the CIO task IT-Strategy with the activities 7-8. The DPs *Stakeholders' Tasks*, *Categorization of Subtasks*, *To-be Decision-specific Subtasks*, Decision-specific *Domain Data*, and *Interaction-Data* were visited in activity 7. In activity 8 the requirements of DPs *To-be Decision-specific Subtasks*, and *Interaction-Data* were refined.

Table 7.2: Overview of artifacts created (ID = Activity ID referring to Table 7.1)

ID	Decision Point	Artifact	Duration
1	Categorization of Subtask Decision-specific Subtask UI-Structure Screen-Structure	Task description Decision-specific Subtask description UI structure diagram Virtual Window	56 h
2	System Functions Interaction-Data	Function description Domain Data Model	15 h
3	System Functions Interaction-Data UI-Data	UI structure diagram Domain Data Model Virtual Window	23 h
4	Interaction UI-Structure Screen-Structure	UI Prototype UI structure diagram Virtual Window	12 h
7	Stakeholders' Tasks Categorization of Subtask Decision-specific Subtask Domain Data Interaction-Data	Task description Task description Decision-specific Subtask description Domain Data Model Domain Data Model	3 h
8	Decision-specific Subtask Interaction-Data	Decision-specific Subtask description Domain Data Model	4 h

We performed the data analysis for both phases after all interviews. In order to refine RQ.1.2, we use a GQM approach [BCR94]. We study the following effects of the application of DsTORE (see [Wie14] for similar distinctions). The **efficiency** is studied in terms of the time taken for the creation of artifacts and the interviews. Therefore, we raise RQ.1.2.1

Table 7.3: Metrics of data collection for the case study. Column question refers to Section A.3

Goal	**Metric**	**Description of Metrics**	**Question**
efficiency	m1	Time taken for the creation of artifacts	–
	m2	Number & duration of interview sessions in total	–
	m3	Stakeholder's feedback to the interviews	1 – 9, subsection A.3.1
usability	m4	Stakeholder's feedback to the artifacts	1 – 38, subsection A.3.3
	m5	Requirements engineer's feedback	–
utility	m6	Stakeholder's acceptance of the system prototype	see Section 7.3

and use the metrics m1 and m2 to obtain the answer. The **usability** is studied in terms of the ease of use of the artifacts, and the process for the stakeholder (m3 and m4) and the requirements engineer (m5). Thus, we raise RQ.1.2.2. The **utility** is studied in terms of the value of the outcome for the stakeholder with RQ.1.2.3 by using m6. The data for metrics m3, m4 and m6 was collected in a semi-structured interview as activity 9, based on the questionnaire in the Appendix Section A.3. Table 7.3 summarizes the metrics to answer RQ.1.2.1 – RQ.1.2.3.

RQ.1.2.1: What effort is required in order to execute the method?

RQ.1.2.2: What is the perceived ease of use of DsTORE by the requirements engineer and the stakeholder?

RQ.1.2.3: What is the value of the outcome for the stakeholder?

7.2 Results for the RE Phase

The creation of artifacts (m1) took 106 hours in phase one and 7 hours in phase two, on the basis of the duration of activities shown in Table 7.2. Four interviews with a total duration of approximately 7 hours in total were conducted (m2) in phase one, and two interviews with a total duration of 3 hours were conducted in phase two, as is shown in Table 7.1. In summary, the answer to RQ.1.2.1 is given by Table 7.2 and Table 7.1. The CIO liked the interviews of both phases and rated the semi-structured execution of interviews as good (m3), since it gave him room to discuss ideas. He suggested two improvements: 1) the provisioning of an RE-process description with a graphic sequence upfront; and 2) the consultation of other departments' experts in some situations. The CIO sometimes found it difficult to provide all

Table 7.4: CIO's rating of the artifacts. (● difficult, ○ easy, ○○ very easy, ++ very important, + important, - less important)

Artifact type	**simplicity**		**importance**
	understanding	comment	understanding
Task description	○	○	+
Categorization of Subtasks	●	●	+
Decision-specific task description	●	○○	+
Data sources	○○	●	++
Workspace model	●	●	+
Virtual windows	○○	○	++
UI prototype	○○	○○	++

relevant decision information. It would also have helped if he had had a prepared description of decisions before the RE phase. The meeting's minutes and the correction iteration was good and helpful for the CIO. The interview preparation and the review were difficult to carry out due to time restrictions. Especially for phase two, the CIO rated the identification of decisions on the whole as good. For the task IT strategy, the CIO rated the description of decision-related information as difficult. He did not consider the definition of the IT strategy as a decision, although the interview had identified five decisions. The CIO's feedback to the artifacts (m4) is summarized in Table 7.4.

The CIO's perception of artifacts did not change between phase one and phase two. All decision-specific information was contained in the artifacts. The CIO perceived the *task description* as important, easy to understand and easy to comment on. However, he considered the *Categorization of Subtasks* difficult to understand and difficult to comment on. Once the categorization was completed, the CIO understood and agreed on its importance. Initially, the CIO did not see the importance of the *Decision-specific Task Description*, and hence he rated the understanding as difficult but easy to comment on. After the first review of this description, he rated them as important. He perceived the description of complex processes (such as decisions) as challenging. The CIO judged the description of *data sources* to be very easy to understand, but difficult to comment on (and fill in), since he was not aware of detailed knowledge about URLs. The CIO perceived the *workspace model* as a complex and overloaded representation and rated it as difficult to understand and comment on. He recommends to hide workspace details and to focus on the navigation between them. The *UI Prototype* gives the most detailed description of the system-to-be and allows the CIO to check whether the UI contains the necessary information. It was rated as very easy to understand and comment on, and as a very important artifact.

The requirements engineer rated (m5) the detailed understanding of the artifact types and their relations to each other as very important for structuring the interviews. The requirements engineer needs to understand the RE process in order to choose an appropriate sequence of DPs to visit and artifacts to create. During the RE phase, s/he must keep all artifacts consistent, which is challenging. The DsTORE artifacts enable a detailed understanding of the decisions. The specification was extremely helpful to guide the developers during the implementation of the system prototype, in particular the knowledge of the required data and their origin. In summary, we can answer RQ.1.2.2 to the effect that some of the used RE artifacts are difficult to understand and comment on, all of them are important, and all of them contain the relevant decision-specific information.

During the system prototype evaluation, which was presented in Section 7.3, the CIO mentioned that the RE phase triggered some changes within his IM division that related to the investigated tasks. We wanted to understand these changes in greater detail and therefore asked the CIO which activities and task changes he started after the RE phases of problem investigation and treatment design. The questionnaire we used is presented in Section A.5. The CIO started the following activities, which also impact upon decisional data.

1) The project planning was consolidated. Previously separated lists with ongoing and waiting projects with the scope of one year were joined together in one continuous project list. Filter criteria helps the CIO to work with this list.

2) The project websites containing the project status were moved to a dedicated project server, which affects the data source model.

3) The financial controlling list will be migrated to the SharePoint server.

4) He plans to evaluate and introduce a contract management tool.

5) He plans to integrate the change management documentation from SharePoint into the IT-service management tool SCSM.

Regarding his tasks, there are no changes compared to the interviews. However, in the future it could be necessary to change workflows, a change which will affect subtasks. The decision-specific information for subtasks has not been changed so far.

Table 7.5: Metrics of system prototype evaluation. The column *question* refers to Section A.4.

Goal	Metric	Description of Metrics	Question
Acceptance	p.m1	Stakeholder's feedback on positive system prototype properties	C.1
	p.m2	Stakeholder's feedback on negative system prototype properties	C.1
	p.m3	Stakeholder's rating of liking	C.5
	p.m4	Missing properties in general	C.2
	p.m5	Identified errors and problems	C.2
	p.m6	Requested improvements by the stakeholder	C.8
Usability	p.m7	Stakeholder's rating of usability	C.3
	p.m8	Stakeholder's feedback to positive usability properties	C.3
	p.m9	Missing properties regarding usability	C.3
Utility	p.m10	Stakeholder's rating of utility	C.4
	p.m11	Likelihood of future use	C.7
	p.m12	Obstacles to future use	C.7
	p.m13	Degree of compliance with given requirements	C.6

7.3 Results of the System Prototype Evaluation

We used an additional GQM scheme for the evaluation of the system prototype (m6 in Table 7.3) regarding acceptance, usability and utility, as is shown in Table 7.5. A screen shot of the CIONx system prototype UI for the task project management is shown in Figure 7.1. As the hospital's default language is German, the system prototype follows this restriction.

The CIO mentioned five positive properties (p.m1) of the system prototype: (a) it was easy for the CIO to understand the tool; (b) CIONx is the central place to gain an overview of data in an aggregated form; (c) CIONx meets the idea of a dashboard; (d) is modularized and extendable; and (e) quick filtering is helpful. On the other hand, the CIO mentioned the following three (p.m2) as negative properties of the system prototype: (a) the system prototype was extremely slow; (b) the results were only partially comprehensible on the basis of the limited test data; and (c) the process of drilling down into the data needs further consideration. The CIO stated (p.m3) that he likes the system prototype *well*. As missing properties (p.m4), the CIO enumerated the following issues: (a) The change request overview needs to contain the last comment of the change history. (b) There is no advantage in displaying the department's budget along with the change details. Show it on a separate page. (c) It is sufficient to show material expenses and service budget for change requests. (d) Show the planned and available budget for material expenses and the service budget. (e) The department's budget should be at the top of the page *project overview*. Some

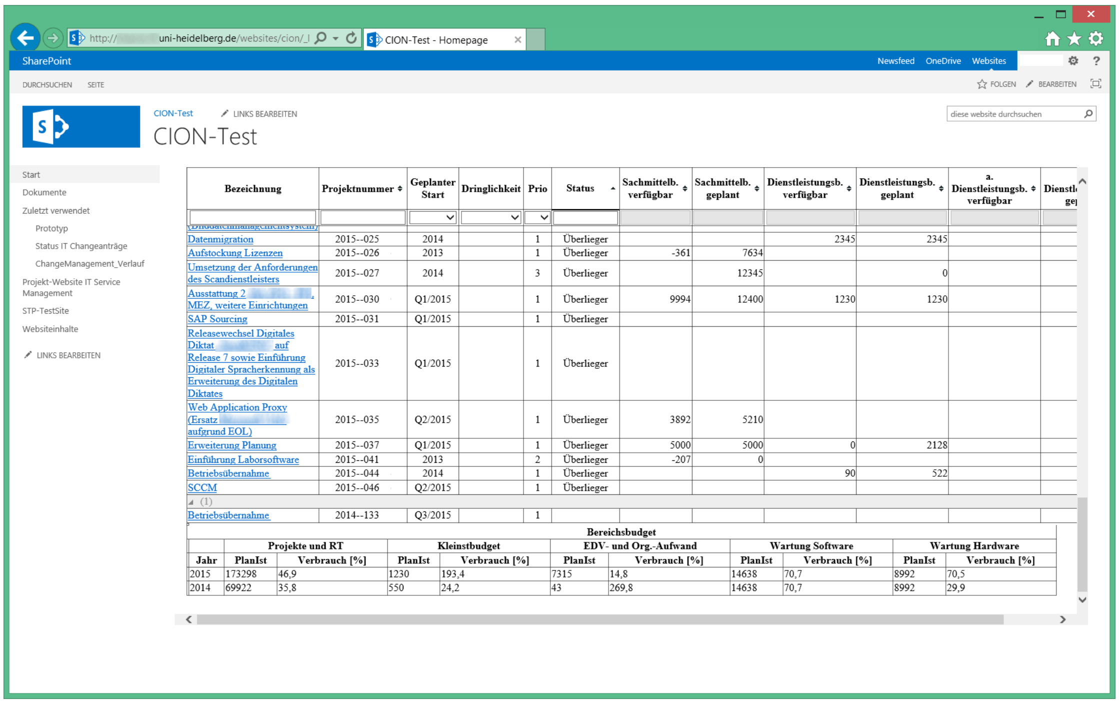
Table 7.6: The obstacles of the future system prototype usage

ID	Obstacle
o1	the availability of all data necessary for decisions
o2	the hospital's ability to continue the development of the software
o3	the existence of defined responsibilities
o4	a proof of operational safety

minor issues in p.m4 included missing buttons, change of labels, and open pages in new tab. The CIO identified two errors related to navigation (p.m5): (a) the return from the project detail page to the overview results in a redirection to the start page; and the (b) horizontal scrolling is unusual. The CIO requested one improvement (p.m6) in this regard, that is, to separate budget onto a different page (this has already been mentioned as missing property, cf. p.m4)). The usability of the system prototype (p.m7) was regarded as *good* by the CIO. The mentioned positive usability property (p.m8) was the ease of navigation between different CIO tasks. The CIO stated that he is not missing any properties related to usability (p.m9), but he assumed that the observed problems might be related to test data. For future development it seems important to the CIO whether the focus of CIONx is to gain an overview or an in-depth view. The CIO rated the utility of the system prototype as (p.m10) *useful* and stated that he *maybe* will use the system prototype in the future (p.m11). Overall, four possible obstacles (p.m12), which are identified by o_n in Table 7.6, hinder the CIO to use the system prototype productively in the future.

The CIO stated that his expectations formed by the requirements specification (p.m13) were *well* fulfilled. Further comments of the CIO were: (a) The RE led to some changes in the division. In particular during the conversations, he gained some ideas for the restructuring of data. (b) The invested time was beneficial. (c) The lessons learned constitute the major value of the project. (d) For the CIO it is still unclear to what extent the dashboard will integrate data from the system environment. Possibly there is more to do in this area.

In summary, we can answer RQ.1.2.3 that the prototype is useful for the stakeholder, has a good usability and fulfills the overall expectations on the basis of the listed requirements. Overall the answer to RQ.1.2 is that DsTORE supports the specification of requirements for a PDSS adequately. All decision-specific information was contained in the artifacts. DsTORE allows an early and continuous stakeholder feedback, and it emphasizes the decision-specific data and decision-specific UI prototyping. Some artifacts were initially difficult to understand but were rated as important. The CIO rated the system prototype as on the whole useful and good, since the CIO's expectations based on requirements were fulfilled.



CION-Test

Bezeichnung	Projektnummer	Geplanter Start	Dringlichkeit	Prio	Status	Sachmittelb. verfügbar	Sachmittelb. geplant	Dienstleistungsb. verfügbar	Dienstleistungsb. geplant	a. Dienstleistungsb. verfügbar
Datenmigration	2015--025	2014		1	Überlieger			2345	2345	
Aufstockung Lizenzen	2015--026	2013		1	Überlieger	-361	7634			
Umsetzung der Anforderungen des Scandienstleisters	2015--027	2014		3	Überlieger		12345		0	
Ausstattung 2 [redacted], MEZ, weitere Einrichtungen	2015--030	Q1/2015		1	Überlieger	9994	12400	1230	1230	
SAP Sourcing	2015--031	Q1/2015		1	Überlieger					
Releasewechsel Digitales Diktat [redacted] auf Release 7 sowie Einführung Digitaler Spracherkennung als Erweiterung des Digitalen Diktates	2015--033	Q1/2015		1	Überlieger					
Web Application Proxy (Ersatz [redacted] aufgrund EOL)	2015--035	Q2/2015		1	Überlieger	3892	5210			
Erweiterung Planung	2015--037	Q1/2015		1	Überlieger	5000	5000	0	2128	
Einführung Laborsoftware	2015--041	2013		2	Überlieger	-207	0			
Betriebsübernahme	2015--044	2014		1	Überlieger			90	522	
SCCM	2015--046	Q2/2015		1	Überlieger					
(1)										
Betriebsübernahme	2014--133	Q3/2015		1						

Bereichsbudget

	Projekte und RT		Kleinstbudget		EDV- und Org.-Aufwand		Wartung Software		Wartung Hardware	
Jahr	PlanIst	Verbrauch [%]	PlanIst	Verbrauch [%]	PlanIst	Verbrauch [%]	PlanIst	Verbrauch [%]	PlanIst	Verbrauch [%]
2015	173298	46,9	1230	193,4	7315	14,8	14638	70,7	8992	70,5
2014	69922	35,8	550	24,2	43	269,8	14638	70,7	8992	29,9

Figure 7.1: The developed software prototype in a browser

7.4 Discussion

In this section, we discuss the results of the DsTORE evaluation and possible improvements to DsTORE, as well as the lessons learned during the method adaptation.

7.4.1 Discussion of DsTORE

As PDSS share some properties with IS (see Table 3.1 on page 43), TORE provides the basis for the RE specification of a PDSS. The *problem investigation* as presented in Chapter 5 has shown the missing decision-specific information that shaped the design of DsTORE. Small adaptations in DsTORE help both to understand decisions and to gain a detailed specification of all PDSS aspects. The *treatment validation* shows that DsTORE is accepted by the stakeholder and the requirements engineer, although there is potential for improvement.

The effort of 106 hours to create the artifacts is very high, but this effort decreased significantly in phase two on the basis of the prior experience. Four interviews lasting 7 hours in total and two interviews lasting 3 hours in total is a reasonable effort. Based on the CIO's feedback, the semi-structured interviews are adequate to elicit requirements and allow for a creative discussion. The results show that the CIO views atomic decisions as a part of larger business- and/or management-processes, and therefore as important to him. The CIO rated only the artifact workspace model as less important for him. We used the workspace model to structure the virtual windows. In the future, the workspace model might be used only by the requirements engineer, while the structure is discussed with the stakeholder directly in the virtual windows. The CIO's rating emphasizes the importance of the data sources and the UI. The CIO's suggestions do not concern the DPs, but rather the execution of the interviews, which can easily be improved. For example, decisions can be identified on the task level with the CIO without specifying details, and can then be detailed with external experts. The latter can also supply the data source details. The categorization of subtasks can be done initially by the requirements engineer and then presented to the CIO for review. The artifacts' consistency can be improved by a yet to be defined tool support. Both the CIO and requirements engineer, felt that a RE-process description was missing which covered activities and artifacts. This can easily be provided in the future.

7.4.2 Discussion of the System Prototype

In this subsection, the results of the system prototype evaluation are discussed in brief. The evaluation of the system prototype completes the answer to RQ.1.2, as the SRS created with DsTORE was used as the basis for the development of the system prototype. The fact that only small changes are necessary to CIONx completes and confirms the evaluation results.

The evaluation of the CIONx system prototype shows its usability and usefulness for the CIO, who on the whole finds the system prototype *well*. In particular his expectations of the system prototype formed by the requirements specification were *well* fulfilled. The evaluation also shows some deficiencies of the system prototype. The deficiencies related to navigation, in particular the negative and missing properties, can be changed in the system prototype with minimal effort. Some deficiencies are directly related to the test system and its data, and are nonexistent in a productive system environment. The reasons why the CIO will *perhaps* use the system prototype (see metric p.m12 of Table 7.5) can be mitigated by the following selective measures. Missing data (see obstacle o1 in Table 7.6) can obviously neither be provided by CIONx nor can it be used with CIONx. It is up to the CIO or a responsible project manager to adapt the department's organizational structures to capture unavailable but relevant data for future use. We saw that most of the data is available as scattered or unstructured data, which increases effort, but principally allows a use in CIONx. The obstacles o2 and o3 (on which see Table 7.6) regarding the hospital's ability to further develop the software and undefined responsibilities can be mitigated by a skilled staff member who is responsible for the software and continuous development. This person needs an initial training and the ability to use a provided system documentation. A simple system structure, together with an easy process to build and deploy the system, will ease its further development. The proof of operational safety (on which see the obstacle o4 in Table 7.6) is the most difficult one, since it targets possible side-effects to the existing IM services. As CIONx is a prototype, there is certainly additional effort necessary to turn it into a product, and then to prove the operational safety. We minimized possible side effects through taking two measures. First, the system prototype is realized as a SharePoint extension, avoiding any new infrastructure. Second, the prototype accesses the data sources in a read-only mode to avoid any file locks.

7.4.3 Lessons Learned

Successful experiences The following sequence is helpful for creating some of the artifacts of DsTORE, in particular: (1) task and subtask, (2) detailed decision description, (3) domain data model, (4) workspace model, (5) virtual windows, and (6) UI Prototype. A template for the naming of subtasks should be used, for example `<verb | nominalized verbs>` `<object>`. During the process of eliciting the to-be decision-specific subtask description, it is important to continuously ask the stakeholder which data exactly is necessary for the decision. It is important to distinguish explicitly between domain entities and attributes, since this is not done directly by the stakeholder. It was helpful during the requirements specification to map attributes of rules onto the domain data model. It is worth the effort to assemble the results of the decision description into virtual windows as early as possible.

Difficulties Synonyms for similar verbs should be avoided. The specification of rules is often awkward, especially when they are complex, or have not been formulated previously.

Improvements triggered by RE The RE activities in which the CIO was involved triggered some follow-up activities in the IM department. During the interviews, we queried and captured details of decisions and their related data. As a result of these interviews, the CIO realized some existing weaknesses and possible improvements. This is an important aspect of RE. For activities such as system assessments and data governance which emphasize deficiencies in data or system structure explicitly are often neglected. The RE for a system like CIONx incorporated a system analysis and the interviews which triggered possible improvements.

7.5 Threats to Validity

The threats to validity are structured according to Runeson et al. [RH09]. **Construct validity** considers whether the study really measures what it claims. The artifact types of DsTORE were not yet fixed during phase 1. As all DsTORE DPs as presented in Chapter 6 were applied in the treatment validation, we believe that the DsTORE artifact types that have been temporarily used and discarded do not distort the final results.

Internal validity considers causal relations between the investigated factors, that is to say effects of unknown factors that influence one or more of the investigated factors. Training obviously affected the effort and the opinion of the stakeholders. The effort decreased between phase one and phase two, owing to training effects of the requirements engineer and the CIO. We asked the CIO about his opinion in activity 9, after he had gained the experience of undergoing both phases. **External Validity** describes the generalizability of the findings and the transfer of study results to beyond the study. The case study is based on a single case only and involved only one person. DsTORE is specific to PDSSs and the transferability to other DSS types is unclear. **Reliability** considers the influence of the specific researcher and indicates threats to validity for a repetition of the study. The main threat to reliability is that the author of this thesis in his role as requirements engineer had a large influence on the development and process of the case study, in particular during the interviews and the data analysis. We mitigated this through continuous discussion with the supervisor of this thesis.

7.6 General Experiences with Method Adaptation

Overall, the adaptation of TORE to system domain specifics is possible. TORE's structure into the DP is helpful, as it provides the basic scheme for method adaptation. During the method adaptation, new DPs can be added, and at the same time existing DPs can be extended or modified. Thus, it is important to consider through careful deliberation whether to add a new or otherwise to modify an existing DP. The decision about adding or modifying a DP needs to reflect the criteria of the novelty of an aspect. Adding a new DP is necessary when the system type in focus requires a previously unconsidered aspect. An example of this is the categorization of subtasks, which is in general not necessary for ISs. To extend or modify an existing DP should be considered if similar aspects are already incorporated into TORE, but if they nevertheless need to be narrowed or broadened to fit the system type in focus. The aspect of the support for decisional data in the data model is a new *detail* of a data model, and thus is not worth a new DP. In particular new aspects that require a more detailed documentation only, such as rules in decision subtasks, do not justify a new DP.

We have used the design science approach and its engineering cycle. This was beneficial for the following reasons. First, it helped us to structure the adaptation of TORE into problem identification, specific adaptations, and their evaluation. Second, it forced us to create explicit justifications for the adaptations, as the design demands the requirements to improve. Third, it forced us to collect explicit feedback from the stakeholders involved in the method execution. It is important to discuss the adaptations with experts who were not involved in the method execution. These experts were several persons, differentiated in two groups. First, they are domain experts of IM in hospitals, who are involved in the SNIK research project and in teaching. Second, they are postgraduate doctoral students. The reason why this is important is that they ask for the rationale and novelty of the artifact and method adaptations. Some misunderstandings of the experts were hints that pointed to unsound adaptations or the need for artifact modifications.

During the *treatment design* we had the experience to develop several further artifact types, which were finally either discarded or simplified. Examples of such intermediate used artifact types are the following: (1) An artifact which we called *information folder*, and which contains similarly to a briefcase a dossier of all relevant decision information; (2) an explicit relation of subtasks, decision and reports; and (3) an artifact called aggregation model, which documents and details reports and their necessary data aggregations. We recognized that the *information folder* is well modeled by VWs and workspace models. Thus, there is no need for a new artifact type. We have discarded the modeling of the explicit relation between decision, reports and subtask, as we realized that decisions *are* subtasks. The *aggregation model* was finally integrated into the extended DDM, as it does not justify being regarded as a separate model, a move that would lead to additional complexity.

7.7 Chapter Summary

In this chapter, the treatment design DsTORE was evaluated in a two-phase *case study* based on the specification and development of CIONx to show its basic feasibility. Four semi-structured interviews have been used to elicit the requirements with DsTORE for CIONx. The CIONx is a dashboard application for decision makers in the information management application domain, to navigate in current data for predefined and reoccurring decisions. The specification comprises decisions within the tasks project management, change management and IT strategy. DsTORE supports the identified six decision specifics (c_1-c_6) for a PDSS method, as was described in Chapter 5 and Chapter 6. In summary, this evaluation shows that DsTORE supports the specification of requirements for a PDSS adequately. DsTORE emphasizes decision-specific data, and allows an early stakeholder feedback based on decision-specific UI prototyping.

Part III

Ontology Usage in Task-oriented Requirements Engineering

8 Introduction

This chapter introduces the details of design problem 2 for the improvement of requirements quality with domain ontologies. In Section 8.1 the refinement of design problem 2 into more specific RQs is explained. Section 8.2 first details the introduction of ontologies that was given in Section 2.3 on page 27, before it moves onto explaining different types of ontologies, and emphasizing domain ontologies.

8.1 Research Questions

The design problem 2 as introduced on page 7 is the following: *improve TORE by designing TOREOnto that satisfies the use of application domain ontologies, in order to support requirements engineers in improving requirements quality*. We refine this design problem into RQ.2.1 and RQ.2.2. Before we are able to design a TORE adaptation that uses domain ontologies to improve requirements quality, it is essential to understand the details of the use of ontologies in RE first. Thus, we investigate through the RQ.2.1 the state of the art in order to understand the strategies as to how domain ontologies are used in RE methods to improve the quality of software requirements. These strategies are described as DO usage patterns. As the foundation for the quality of software requirements, we use the requirements quality attributes that are based on the ISO/IEC 29148 standard [ISO11], as were presented

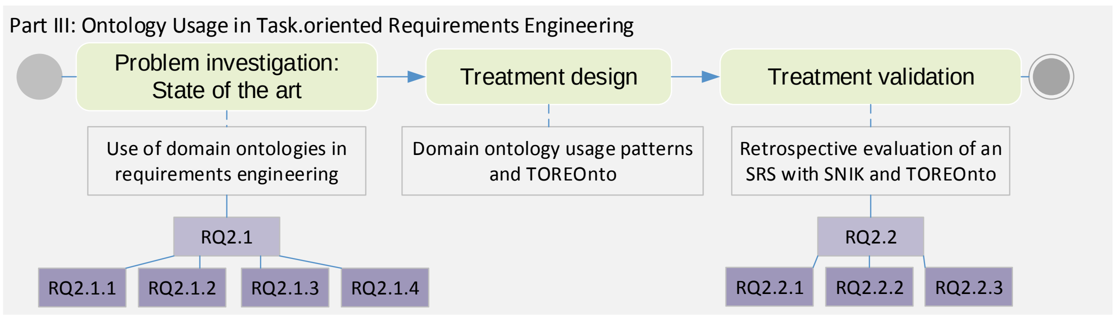

Part III: Ontology Usage in Task.oriented Requirements Engineering
Problem investigation: State of the art
Treatment design
Treatment validation
Use of domain ontologies in requirements engineering
Domain ontology usage patterns and TOREOnto
Retrospective evaluation of an SRS with SNIK and TOREOnto
RQ2.1
RQ2.2
RQ2.1.1
RQ2.1.2
RQ2.1.3
RQ2.1.4
RQ2.2.1
RQ2.2.2
RQ2.2.3

Figure 8.1: Overview of design science approach and RQs for Part III

in subsection 2.1.3 on page 17. The results of the literature review are not only useful for RE method designers who want to integrate existing techniques for InR and SRS-quality improvement into their RE approaches or techniques, but also for requirements engineers who want to improve the quality of the requirements specification they produce with domain ontology. Based on this knowledge, TOREOnto is designed as a concrete instance of TORE to use domain knowledge.

In order to validate the design of TOREOnto, we raise RQ.2.2 that queries TOREOnto's ability to support the specification of requirements. Figure 8.1 shows the design science cycle that is used to answer both RQs in Part III of this thesis. The RQ.2.1 is answered in the problem investigation. Next, in the treatment design, TOREOnto and the DO usage patterns are defined. Finally, in the treatment validation, TOREOnto is retrospectively evaluated in order to answer RQ.2.2.

RQ.2.1: How can domain ontologies be applied in an RE method to improve the quality of software requirements?

RQ.2.2: How well does TOREOnto support the specification of requirements?

8.2 Types of Ontologies

In order to complete the introduction to ontologies as was earlier presented in Section 2.3 on page 27, the different classification schemes of ontologies are presented in this section. Studer et al. [SBF98] provide a general classification of ontologies, showing an early occurrence of a DO. Happel et al. [HS06] highlight that ontologies that are relevant to RE contain both a semantic description of requirements specification documents and a formal representation of requirements knowledge. Castañeda et al. have a more specific focus on ontologies in RE and distinguish three types of ontologies [CBCG10]. First, a *requirements ontology* contains the taxonomy of requirements-types in terms of a hierarchy, for example functional or nonfunctional requirements, or organizational requirements. Second, a *requirements specification document ontology* contains the elements of a requirements specification document and their relations as concepts. For example, such an ontology describes that a requirements specification document consists of actors, requirements, scenarios and goals, along with the rule that each scenario must have at least one goal. Third, and lastly, an *application DO* represents the domain knowledge needed to build software applications within a specific domain.

Domain Ontologies In contrast to Castañeda et al. [CBCG10], we follow the definition of Studer et al. [SBF98]. We define a *domain ontology (DO)* as a description of an application domain with domain concepts in general, rather than offering a definition that is necessarily specific to software or ISs. Examples of application domains include aerospace, e-business, or IM. Domain concepts can specialize the terms introduced in a top-level ontology [Gua98]. A DO may comprise, among other things, typical roles in the application domain[1], functions or tasks, common application components, and data entities. An example of a DO is the SNIK DO [JSPW14], as was presented in subsection 2.5.2 on page 32, which contains concepts and their relations of the application domain IM in hospitals.

[1]Note that in the primary studies what we call the application domain is often called the problem domain.

9 State of the Art - Literature Review

This chapter describes the literature review which constitutes the answer to RQ.2.1, namely: How can domain ontologies be applied in an RE method to improve the quality of software requirements? In Section 9.1 details about the research method, including the research questions of the literature review, the selected sources and their description, as well as the data extraction and synthesis, are all individually described. The results of the literature review are presented in Section 9.2, as an answer to the research questions. In Section 9.3 the principal usage scenarios of domain ontologies to improve requirements quality are described in the form of domain ontology usage patterns. These results are then discussed in Section 9.4, along with related works and threats to validity. Finally, this chapter is summarized in Section 9.5.

9.1 Research Method

This section provides a description of the literature review method we used, presents and explains the motivation of the research questions we will subsequently answer, describes the sources' selection and an overview of additional studies, and finally describes the data extraction.

9.1.1 Our Approach

The main purpose for conducting a literature review is to gain an overview of the current approaches as the basis for drawing conclusions about a research topic. In this review, we follow the principles of a systematic literature review (SLR) set out by Kitchenham and Charters [KC07]. A *primary study*[1] is an article selected for inclusion in the review because it provides information that is needed to address the reviewers' research questions. For a study that reviews all primary studies with the aim of integrating and synthesizing evidence

[1]We use this term even for articles which do not provide any form of empirical evidence.

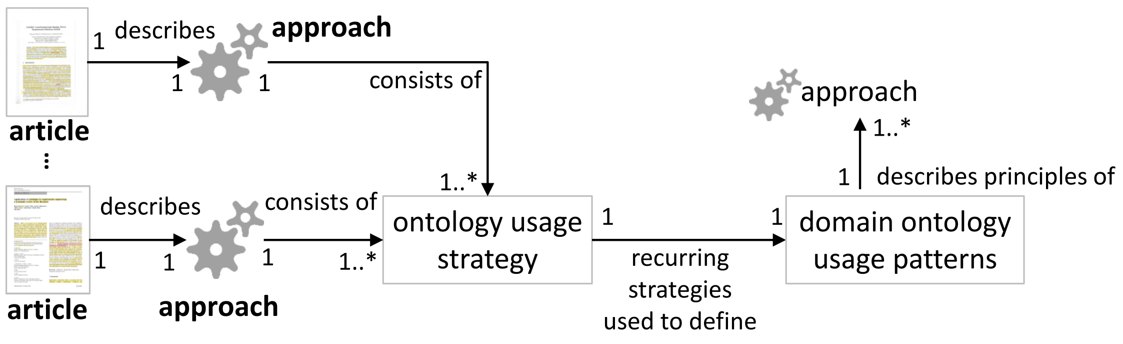


Figure 9.1: Our approach to describe various ontology usage strategies as patterns

related to a specific research question, we use the term *secondary study*. An SLR is a form of secondary study. Figure 9.1 shows our synthesis process with the goal of answering RQ.2.1. The numbers at the edge of the arrows indicate the cardinality of the connected artifacts, where 1..* means at least one. Each article describes a single approach that consists of one or several *ontology usage strategies*. We have analyzed the recurrence of DO usage strategies, and defined *domain ontology usage patterns* as similar to a software design pattern [Gam08].

9.1.2 Research Questions

This problem investigation constitutes an answer to RQ.2.1: *How can domain ontologies be applied in an RE method to improve the quality of software requirements?* We refine this RQ into the following five more specific RQs:

RQ.2.1: How can domain ontologies be applied in an RE method to improve the quality of software requirements?

- RQ.2.1.1: Which ontology types are used for RE?
- RQ.2.1.2: What kind of RE artifacts and which quality attributes are improved by using a DO?
- RQ.2.1.3: How exactly are DOs used?
- RQ.2.1.4: What is the applicability and maturity of the DO-based approach?
- RQ.2.1.5: Which DO usage strategies improve which requirements quality attributes?

First, we need to understand in RQ.2.1.1 the content of the ontologies used in the approaches in order to distinguish between different types of used ontologies. The types of ontologies are introduced in Section 8.2 on page 94. More specifically, the content of an ontology describes a domain of discourse through describing its knowledge elements. Second, as we are interested in the improvement of requirements, we need to understand through answering RQ.2.1.2 what kind of requirements the approaches are able to improve with regard to the RQAs of ISO/IEC 29148 [ISO11], as is presented in Table 2.1. Third, in order to understand how the quality of requirements are improved with the DOs, we need to provide details about the exact DO usage throughout the approaches, as was queried by RQ.2.1.3. Fourth, in order to rate the maturity and applicability of the improvement strategies, we investigate the evidence that is presented, its existing limitations and the preconditions of the described approach in RQ.2.1.4. Fifth, the coherence between DO usage strategies and the improved RQAs are semi-formally described by the answer to RQ.2.1.5.

In order to refine RQ.2.1.1 – RQ.2.1.4, we use the GQM scheme [BCR94] as is presented in Table 9.1 to define the metrics $m_{i.j.k}$ for the characterization of each primary study. The motivation to refine RQs and the metrics can be explained in the following way. In order to answer the question of which ontology types are used for RE, we determine the ontology's type (RQ.2.1.1.1 and m1.1.1) through recourse to the knowledge elements (RQ.2.1.1.2 and m1.2.1) contained in the ontology. We then split RQ.2.1.2 into two aspects: 1) The improved RQAs according to ISO/IEC 29148 (m3.2.1) and the number of improved RQAs (m2.1.1) in RQ.2.1.2.1. This aspect is chosen because the combination of the metrics m2.1.1 and m2.1.2 allows us to catch effects where the same ontology usage strategy improves multiple RQAs; and 2) The artifact that is in focus of the improvement (RQ.2.1.2.2). We distinguish between text- (m2.2.1) and model-based (m2.2.2) requirements documentations according to Pohl and Rupp [PR15]. In order to understand the details of the DO usage, we refine RQ.2.1.3 into six finely grained RQs: 1) We determine the application domain of the DO (m3.1.1). 2) The organization of the DO defines its principal extent to improve requirements quality. In particular this is a metamodel that is used to organize the ontology, including entity types, relations (m3.2.1), and individuals (m3.2.2). 3) The exact use of ontologies is restricted by the definition of ontologies, as was introduced in Section 2.3 on page 27, namely as ontology modification (m3.3.1), querying (m3.3.2) and reasoning (m3.3.3). 4) The output of modification (m3.4.1) and input for the querying (m3.4.2) as well as the input (m3.4.3) and rules for reasoning (m3.4.4) each sharpen our understanding of the DO usage. 5) As the goal is to understand the requirements quality improvement, at some point the requirements artifacts must be modified (m3.5.1). 6) Possible tools and their features provide more details for the understanding of DO usage for requirements quality improvement. In order to assess the applicability and maturity of the approaches for RQ.2.1.4, we investigate possible preconditions in RQ.2.1.4.1 of the artifact (m4.1.1), the ontology (m4.1.2) and the requirements in RQ.2.1.4.2 (m4.1.3). Finally, the presented evaluation (m4.2.1) is questioned in RQ.2.1.4.1 in order to indicate the maturity of the approach.

Table 9.1: Research Questions and their refinement with goal, question, metric.

Goal	Question	Metric
RQ.2.1.1: Which ontology types are used for RE?		
	RQ.2.1.1.1: What type of ontology is used?	m1.1.1: type of ontology
	RQ.2.1.1.2: Which knowledge is contained in the ontology?	m1.2.1: the knowledge contained in the ontology
RQ.2.1.2: What kind of RE artifacts and which quality attributes are improved by using a DO?		
	RQ.2.1.2.1: Which requirements quality attribute is improved by using the approach with the DO?	m2.1.1: the number of requirements quality attributes that are addressed
		m2.1.2: number of addressed requirements quality attributes
	RQ.2.1.2.2: Which type of RE artifact is improved with the DO?	m2.2.1: type of text-based artifacts
		m2.2.2: type of model-based artifacts
RQ.2.1.3: How exactly is the DO used?		
	RQ.2.1.3.1: What are the domains of the DO used?	m3.1.1: domain of DO
	RQ.2.1.3.2: How is the DO organized?	m3.2.1: DO metamodel
		m3.2.2: type of individuals in DO
	RQ.2.1.3.3: How exactly is the DO used?	m3.3.1: modification of DO during RE
		m3.3.2: querying of DO during RE
		m3.3.3: reasoning of DO during RE
	RQ.2.1.3.4: Which input or output is related to DO usage?	m3.4.1: output generated by modification
		m3.4.2: input used for querying
		m3.4.3: input used for reasoning
		m3.4.4: kind of rules used
	RQ.2.1.3.5: How are artifacts modified using a DO?	m3.5.1: kind of artifact modification with DO
	RQ.2.1.3.6: What are the features of a possibly existing tool regarding the DO?	m3.6.1: tool exists
		m3.6.2: tool features described
RQ.2.1.4: What is the applicability and maturity of the DO-based approach?		
	RQ.2.1.4.1: Which preconditions do exist for the approach?	m4.1.1: type of artifact-related preconditions
		m4.1.2: type of DO-related preconditions
		m4.1.3: details of preconditions for requirements
	RQ.2.1.4.2: How is the DO approach evaluated?	m4.2.1: type of evaluation
-	Summary of approach	short description of approach

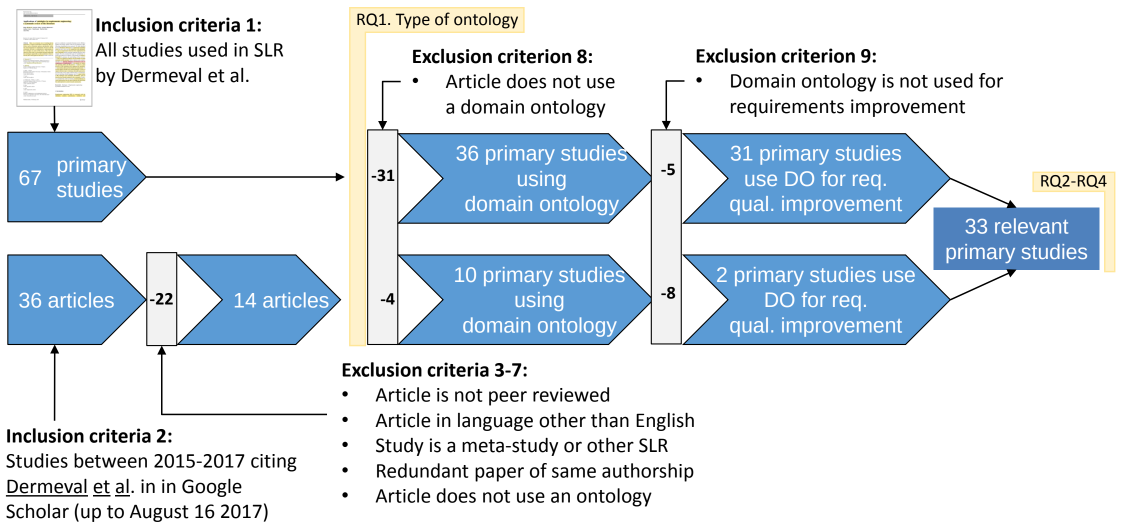

Inclusion criteria 1:
All studies used in SLR by Dermeval et al.
67 primary studies
36 articles
-22
14 articles
Inclusion criteria 2:
Studies between 2015-2017 citing Dermeval et al. in in Google Scholar (up to August 16 2017)
RQ1. Type of ontology
Exclusion criterion 8:
• Article does not use a domain ontology
-31
-4
36 primary studies using domain ontology
10 primary studies using domain ontology
Exclusion criterion 9:
• Domain ontology is not used for requirements improvement
-5
-8
31 primary studies use DO for req. qual. improvement
2 primary studies use DO for req. qual. improvement
RQ2-RQ4
33 relevant primary studies
Exclusion criteria 3-7:
• Article is not peer reviewed
• Article in language other than English
• Study is a meta-study or other SLR
• Redundant paper of same authorship
• Article does not use an ontology

Figure 9.2: Primary study selection flowchart

9.1.3 Sources Selection and Gap

When we began our research on DO usage in RE, we identified the SLR by Dermeval et al. [DVB+15] that provides a comprehensive overview of the applications of ontologies in RE. Figure 9.2 shows the steps of our selection process and the number of included and excluded studies in each step. Table 9.4 shows the inclusion and exclusion criteria used in our selection process. We used all 67 primary studies investigated by Dermeval et al. (criterion 1). We filled the gap between the publication date of [DVB+15] in 2015 and August 16, 2017, by means of a snowballing-search in Google Scholar that resulted in 36 articles that reference Dermeval's SLR (criterion 2). We applied the exclusion criteria 3 to 8 to these 36 studies, resulting in 14 studies that match the criteria of the 67 selected studies by Dermeval et al., as is shown in Table 9.2. In total, we now have 81 primary studies. We classified the used ontologies in each of the 81 studies in order to answer RQ.2.1.1, as explained in subsection 9.1.5, and we applied exclusion criterion 9 to focus on studies that use *domain knowledge*. This resulted in 46 studies as shown in the row *domain ontology* in Table E.1 on page 243. Not all of these studies use a DO to improve requirements quality, which is however our focus according the main RQ.2.1. Therefore, we applied exclusion criterion 9 to select those primary studies which use a DO to improve requirements quality, resulting in 33 primary studies that we now use for the synthesis to answer RQ.2.1.2 – RQ.2.1.4, as is shown in columns 1 to 3 inTable 9.3. Study IDs starting with *S* correlate to the studies used in Dermeval et al., while those with *SNB* were added as the result of the snowball-search.

Table 9.4 shows the relation of our criteria to those of the SLR by Dermeval et al. in

Table 9.2: List of additional primary studies included in the review

ID	Author	Reference
SNB01	Nguyen, Tuong Huan et al.	[NGA16]
SNB02	Casagrande, Erik et al.	[CAW+16]
SNB03	Morales-Ramirez, Itzel et al.	[MRPG15]
SNB04	Liu, Chi-Lun	[Liu16]
SNB05	Dwivedi, Ashish Kumar et al.	[DTR16]
SNB06	Yuan, Xiaobu et al.	[YT15]
SNB07	Mahmoud, Anas et al.	[MC15]
SNB08	Leukel, Joerg et al.	[LSH16]
SNB09	Parreira Jr, Paulo Afonso et al.	[PP15]
SNB10	Zolotas, Christoforos, et al.	[ZDCS16]
SNB11	Olmos-Sánchez, Karla, et al.	[OSJM+16]
SNB12	Soares et al.	[SVG+16]
SNB13	Holanda O et al.	[HIB+17]
SNB14	Chimalakonda, S	[Chi17]

the third column *(crit. in Dermeval)*. Our inclusion criteria 1 and 2 are comparable to Dermeval's criteria 1 (primary studies) and 4 (publication date). Our exclusion criteria refer to Dermeval's criteria in the following way (and are in the following denoted as →): 3 → 2 (peer reviewed) and 17 (grey literature), 4 → 10 (non-English papers), 5 → 6 (secondary studies), 7 → 3 and 20 (20 is negation of 3; no ontology in RE), 6 → 19. Our criterion 8 and 9 are specific to our RQ.2.1. We did not adapt Dermeval's criteria 7 (short papers) and 9 (duplicates) as there were none. We did not apply Dermeval's detailed quality questions for criterion 5 (minimum quality threshold) for both additional articles SNB01 and SNB07. We did not consider Dermeval's criteria 11 to 16 as they are specific to particular technologies (service-oriented architecture, agent orientation, exclusive business process models, and IT exclusive papers) and so were less relevant to our research focus.

9.1.4 Overview of the Studies from the Gap

The details of the studies S1-S67 are described in Dermeval et al [DVB+15]. Here we give a brief overview of the additional studies SNB01 and SNB07 that use a DO to improve requirements quality.

In SNB01 [NGA16], the authors propose an integration framework by which to improve goal models and use-case models regarding correctness, consistency, and completeness. The framework is based on a functional grammar to enable the semi-automated transformation of natural language specifications into OWL syntax through automated reasoning. The defects

Table 9.3: Primary studies included in this review, supported quality attributes and assigned DO usage patterns (p).

Primary study meta data			Supp. quality attrib.							Associated DO usage pattern									
ID	Author	Ref.	InR.complete	InR.correct	InR.unambiguous	InR.consistent	InR.traceable	SRS.complete	SRS.consistent	ontology learning p	ontology population p	consolidate req. p	review DO p	glossary terms p	knowledge base p	complete template p	reasoning p	filter req. text p
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
S01	Al Balushi et al.	[ASL13]						✗	✗						✗	✗	✗	
S03	Aranda et al.	[AVP09]			✗									✗				
S05	Bagheri et al.	[BAE+11]			✗									✗				
S07	Boukhari et al.	[BBJ12]			✗									✗		✗		
S09	Castañeda et al.	[CBC12]			✗		✗	✗	✗	✗			✗		✗		✗	
S12	Daramola et al.	[DSM12]	✗	✗	✗											✗		
S13	Daramola et al.	[DSOS13]			✗			✗			✗				✗	✗		
S14	de Lima et al.	[DRXD10]			✗	✗	✗							✗				
S15	Diallo et al.	[DNA+08]							✗				✗	✗				
S16	Dzung et al.	[DO12]		✗				✗			✗						✗	
S17	Dzung et al.	[DO09]		✗				✗			✗						✗	
S19	Farfeleder et al.	[FMK+11]			✗				✗							✗		
S20	Gailly et al.	[GEPP08]		✗			✗							✗				
S22	Ghaisas et al.	[GA13]		✗	✗			✗	✗		✗						✗	
S33	Kaiya et al.	[KSY+10]	✗	✗						✗					✗			
S36	Kitamura et al.	[KHKS08]			✗	✗		✗		✗	✗						✗	
S37	Kof et al.	[KGRS10]							✗	✗		✗						
S38	Kossmann et al.	[KO10]	✗			✗	✗			✗				✗	✗			
S39	Kroha et al.	[KJL09]							✗		✗		✗				✗	
S41	Li et al.	[LJXL11]				✗				✗			✗					
S43	Liu	[Liu10]				✗				✗							✗	
S44	Liu et al.	[LY12]				✗								✗		✗		
S45	López et al.	[LAC08]				✗									✗			
S50	Nguyen et al.	[NVLG14]							✗		✗						✗	
S53	Osis et al.	[OSJ12]	✗		✗							✗						
S54	Pires et al.	[PDC+11]			✗	✗	✗			✗				✗			✗	
S55	Polpinij	[Pol09]			✗					✗								✗
S58	Riechert et al.	[RB09]									✗			✗				
S61	Shibaoka et al.	[SKS07]						✗			✗						✗	
S62	Souag et al.	[SSWM13]						✗									✗	
S67	Li et al.	[LWZX07]		✗	✗												✗	
SNB01	Nguyen et al.	[NGA16]	✗	✗		✗		✗	✗							✗	✗	
SNB07	Mahmoud et al.	[MC15]			✗			✗	✗		✗		✗				✗	
Total	33 studies	–	5	8	15	9	5	11	10	8	10	1	6	10	6	7	14	1

Table 9.4: Inclusion and exclusion criteria, Crit.=criterion

#	Inclusion criterion	Crit. in Dermeval
1	All primary studies used in SLR by Dermeval et al.	1 and 4
2	Articles between 2015–2017 citing Dermeval et al.	–
#	**Exclusion criterion**	
3	Non-peer-reviewed articles	2 and 17
4	Articles in languages other than English	10
5	Secondary studies (i.e. meta-study or other SLRs)	6
6	Redundant paper of same authorship	19
7	Article does not use an ontology	3 and 20
8	Article does not use a domain ontology	–
9	Ontology not used for requirements improvement	–

of incompleteness and incorrectness are detected by one of five detection rules and are notated as syntactic or semantic problems. For each defect, the authors propose a resolution strategy, such as adding objects if incomplete, removal of requirements or change requests if incorrect, and removal or change the weakening of restrictions in the requirement if inconsistent.

The proposed approach in SNB07 [MC15] uses Wikipedia as a taxonomy and corpus for several natural language processing (NLP) tasks. NLP tasks use information extraction to extract terms from natural language documents and they consider word and text similarity to enable ontology learning. Their principal approach is to use DOs, extract terms, and use the Wikimedia categories to create semantic word nets. These semantic word nets can then be used to create models or to detect conflicts, related requirements – in particular non functional requirements (NFRs) – and to avoid ambiguity. The authors present a research agenda, but no tool or concrete approach with evaluation.

9.1.5 Data Extraction and Synthesis

We extracted details of the ontologies (m1.1.1 – m1.2.1) used in the snowball studies (SNB01-SNB14) prior to applying exclusion criterion 10. Further to this, we extracted all details of the 67 studies (S01-S67) and the two studies that match criteria 8 and 9 (SNB01, SNB07). Based on this data set, we classified the used ontology in all 81 primary studies, as is shown in subsection 9.2.1.

We used a spreadsheet-based extraction form, which is presented in Table E.5 in Appendix E to record all details of the primary studies with the goal of answering the RQs. The extraction form is structured according to the metrics from the GQM scheme (see Table 9.1 in subsection 9.1.2). Each primary study was read, and keywords or text passages that contribute to one of the metrics were highlighted in the primary study and manually added to the extraction form. The spreadsheet contains the study identifier, authors and title as general information about each study. The metric specific data comprises the following information:

1. The used type (m1.1.1) and the knowledge contained in the ontology (m1.2.1). These two metrics are used to classify the ontologies for the application of exclusion criterion 9 in Table 9.4.

2. The supported RQAs (m2.1.1 and m2.1.2) and RE artifacts (m2.2.1 and m2.2.2). If a quality attribute is explicitly mentioned in the primary study and also matches with the primary study's content, we adopted the stated quality attribute. Otherwise we used our own interpretation. We considered both the supported text-based artifacts (m2.2.1), such as use-case text or a specification document, and model-based artifacts (m2.2.2) such as UML class diagrams or goal graphs. From the supported artifacts and the information about possibly existing tools (m3.6.1. and m3.6.2), such as screen shots or feature descriptions in text, we also deducted the contribution of the approach to InRs or to SRSs quality.

3. The details about how DOs are used to improve requirements quality are the following: the ontology details (m3.1.1 – m3.2.2), the ontology usage (m3.3.1 – m3.3.3), along with the input and output related to its usage (m3.4.1 – m3.4.4), the kind of artifact modification (m3.5.1), and details about possible tool features (m3.6.1 – m3.6.2).

4. Preconditions of the approach (m4.1.1 – m4.1.3).

5. The type of evaluation presented (m4.2.1).

6. A summary of the approach.

Percentages presented in the next section are rounded to an accuracy of zero decimal places in tables and in the text. Due to rounding errors, the sum might beyond 100 %, in which case it is given as a *rounding error* explicitly in each table when not zero. The completed extraction sheet providing the basis for the analysis presented in the next section is publicly available here[2].

[2]http://se.ifi.uni-heidelberg.de/fileadmin/pdf/publications/2017_DataExtraction_OntologyUsageInRE_kuecherer.xlsx.

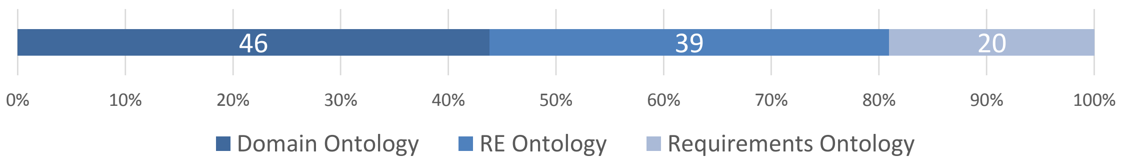


Figure 9.3: The distribution of ontology types among the studies

9.2 Results and Analysis

In this section, the results for each RQ are presented. Note that for RQ.2.1.1 we use 81 and for RQ.2.1.2 – RQ.2.1.4 we use 33 studies respectively.

9.2.1 RQ.2.1.1: Ontology Types Used for RE

RQ.2.1.1.1: What type of ontology is used? Table E.1 in Appendix E shows the ontology types identified in the 81 primary studies, answering RQ.2.1.1.1. Dermeval et al. classified the used ontologies of 26 studies (out of 67) into 13 groups, whereas the ontologies of the other 41 studies were not classified. For the classification of the ontologies (m1.1.1), we follow the proposed groups of Castañeda et al. [CBCG10], with some deviations. The first group *DO* contains a semantic description of a specific application domain, as introduced in Section 8.2. As the second group, we generalize Castañeda's *requirements specification document ontology* to the more comprehensive *RE ontology* that comprises domain-independent RE knowledge. The members of the third group *requirements ontology* contain terms of the requirements as individuals for a specific system or a class of systems. Figure 9.3 shows the distribution of ontology types identified among the studies. In summary 46 approaches (44 %) use a DO, 39 approaches (37 %) use a RE ontology, and 20 approaches (19 %) use a requirements ontology. The use of multiple ontology types in one approach is possible. 58 approaches (72 %) use one type of ontology, 22 approaches (27 %) use two types of ontology, and one approach (1 %) uses all three types of ontology. For the remaining part of this chapter, only those approaches using a DO to improve requirements quality are considered for analysis, as is shown in Table 9.3.

RQ.2.1.1.2: Which knowledge is contained in the ontology? The answer to RQ.2.1.1.2 is given in Table 9.5, Table E.3, and Table E.2, the latter two of which are presented in Appendix E. On a conceptual level, an ontology contains knowledge elements about some phenomena. The knowledge contained in the ontologies that were used in the approaches (m1.2.1) shows a great variety. In order to provide a clearer presentation, we show the

Table 9.5: The knowledge contained in domain ontologies (multiple mentions possible)

Knowledge in ontology	Studies	Count	%
domain (dom.) entities (terms), concepts and relations and rules	S01, S03, S07, S09, S13, S15, S16, S17, S19, S20, S22, S26, S29, S33, S36, S37, S39, S41, S43, S44, S50, S52, S53, S54, S55, S57, S58, S59, S61, S62, S67, SNB01, SNB02, SNB03, SNB06, SNB07, SNB08, SNB10, SNB11, SNB13, SNB14	41	89 %
stakeholders' needs, goal hierarchies	S38, S45	2	4 %
features of software product lines	S05	1	2 %
business processes knowledge	S14	1	2 %
none given	S12	1	2 %
total	46 studies, rounding error = −1 %	46	100 %

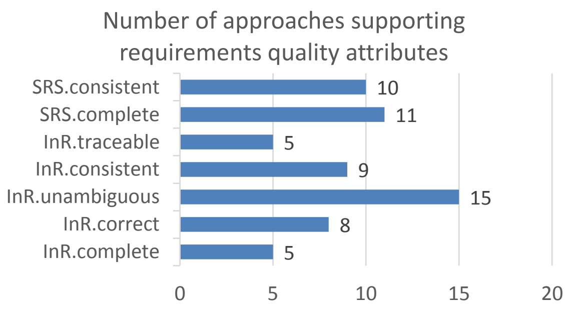
knowledge contained in the ontologies according to the ontology types (m1.1.1). Details of RE methods and processes, semantics, categories and guidelines of how to write requirements, the content of RE artifacts (such as an SRS), semantics of UML models and software architecture, and details of NFRs aspects or metrics can all be found in RE ontologies. The knowledge contained in requirements ontologies focuses on extracted requirements, business rules, and the relations and interactions of system components. The requirements are in particular functional requirements and NFRs, which sometimes can be reused in other contexts. The knowledge of the DOs are domain entities and their relations to each other, along with rules that describe principles of the application domain. Some DOs explicitly contain knowledge about stakeholders' needs and goal hierarchies, or details about business processes.

9.2.2 RQ.2.1.2: Supported RE Artifacts and Requirements Quality Attributes

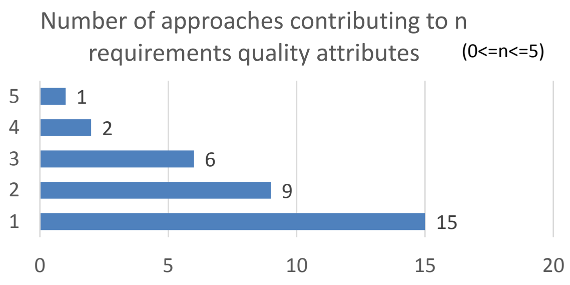
The purpose of this research question is to understand which of the ISO 29148 quality attributes (see Table 2.1), and of which RE artifacts, can be improved by the use of DOs.

RQ.2.1.2.1: Which RQA is improved by the approach with the DO? Figure 9.4a answers RQ.2.1.2.1. The requirements InRs quality attributes completeness (5 approaches), correctness (8 approaches), unambiguity (15 approaches), consistency (9 approaches), and traceability (5 approaches) can all be improved by the use of DOs. We found no support for InR quality attributes verifiable, necessary, singular, or feasible. The SRSs quality attributes completeness (11 approaches) and consistency (10 approaches) can be improved with DOs, but we found no support for affordable or bounded. Note also that one approach can contribute to one or more quality attributes.

The number of supported quality attributes in each study (m2.1.2) is depicted in Figure 9.4b. The approach described by SNB01 supported the improvement of five different quality attributes (InR.complete, InR.correct, InR.consistent, SRS.complete, SRS.consistent) at the same time, which was the maximum number we found. The improvement of four quality attributes at the same time was described in S09 (InR.unambiguous, InR.traceable, SRS.complete, SRS.consistent) and S22 (InR.correct, InR.unambiguous, SRS.complete, SRS.-consistent). Six approaches support three quality attributes, nine approaches support two quality attributes, and fifteen approaches support one quality attribute. Therefore, we add the following answer to RQ.2.1.2.1: The majority of the approaches improve one quality attribute. There is no approach that supports more than 5 quality attributes at the same time.



(a) Supported requirements quality attributes



(b) Number of quality attributes per study

Figure 9.4: Details of requirements quality attributes

RQ.2.1.2.2: Which type of RE artifact is improved with the DO? We distinguish between text-based artifacts (m2.2.1) and model-based artifacts (m2.2.2) as the focus of an approach. Figure 9.5 shows the distribution between the artifacts.

Text-based artifacts are used in 26 out of 33 (79 %) of the approaches. Table 9.6 shows the identified text-based artifacts. Natural language requirements are expressed by written natural language, such as lists of requirements, features, or scenarios, which can be found in 15

Table 9.6: Details of the text-based artifacts used in approaches. (multiple mentions)

Text-based artifacts	Studies	Count	%
Natural language requirements	S01, S16, S17, S22, S33, S36, S37, S38, S41, S43, S44, S50, S55, SNB01, SNB07	15	56 %
Controlled natural language	S19, S54, S38	3	11 %
SRS	S03, S39	2	7 %
Any type of RE documents	S09, S13, S14	3	11 %
Use case text	S12, S15, S53	3	11 %
Text content in wiki	S58	1	4 %
Total	26 studies, rounding error = 2 %	27	100 %

approaches (56 %). Controlled natural language (CNL) allows the structured documentation of natural language requirements with predefined positions for certain term types, such as roles, activities or objects. In some articles, the term *boilerplates* is used synonymously with the term CNL. CNL can be found in 3 approaches (11 %). Two approaches (7 %) mentioned an SRS, in one case according to the ISO/IEC standard 830-1998. Three approaches (11 %) use RE documents in general, which might also contain models. Three approaches (11 %) build on use case text, and one approach (4 %) focuses on text in wikis.

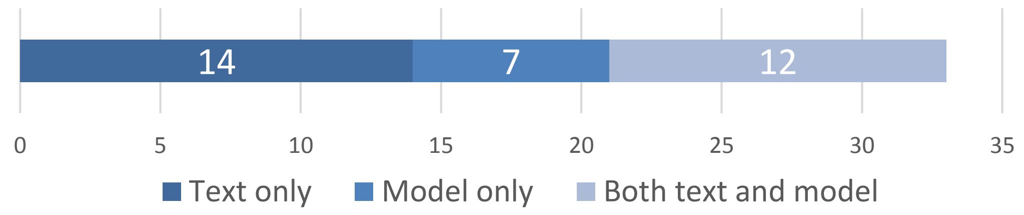


Figure 9.5: The distribution of text-based and model-based approaches

Model artifacts are used in 19 out of 33 (58 %) of the approaches. The used model-based artifacts in the approaches are depicted in Table 9.7. Both models, UML use case diagrams and goal graphs (also comprising i* goal models or softgoal dependency graphs) are used in 7 approaches each.

UML activity diagrams are in the focus of 2 approaches, UML class diagrams are the focus of 4 approaches, and UML component diagrams are the focus of 1 approach. A general data model or domain model is used in one approach. One approach focuses on message sequence charts, two approaches support a feature model, and three approaches focus on (business-)process models.

Table 9.7: Details of the model-based artifacts used in approaches. (multiple mentions)

Model-based artifacts	Studies	Count	%
UML use case diagram	S03, S07, S14, S22, S44, S53, SNB01	7	25 %
UML activity diagram	S14, S43	2	7 %
UML class diagram	S14, S39, S54, S67	4	14 %
UML component diagram	S14	1	4 %
Data/domain model	S22	1	4 %
Message sequence chart	S37	1	4 %
Feature model	S05, S22	2	7 %
Goal graph	S07, S20, S45, S50, S61, S62, SNB01	7	25 %
(Business-) process model	S07, S22, S41	3	11 %
Total	19 studies	28	100 %

In summary, we can answer RQ.2.1.2.2 (see Table 9.1) by stating that the majority of approaches focus on text-based artifacts, which are in particular natural language requirements, controlled natural language, and requirements documents. In the case of model-based artifacts, the most prominent models are goal graphs, UML use case models, class diagrams, and activity diagrams. Our results concerning the RE artifacts which are improved conform to the findings of Dermeval et al. [DVB+15] for any kind of ontology. This indicates that there is no specific relation between the use of DOs and the improved RE artifacts.

9.2.3 RQ.2.1.3: Usage of Domain Ontologies for Requirements Improvement

This section answers how DOs are used exactly to improve requirements quality.

RQ.2.1.3.1: What are the domains of the used DO? The domains (m3.1.1) of the studies which use a DO are shown in Table E.4 in Appendix E. The wide variety of domains shows that the use of DO in RE is not specific to any application domain.

RQ.2.1.3.2: How is the DO organized? We were able to find the presence and details of a metamodel (m3.2.1) in 25 out of 33 (76 %) of the studies. All other studies did not use or mention any kind of metamodel. We distinguish between general concepts, which are presented as an overview[3] in Figure E.1[4] on page 245, and relations used in the ontologies, which are presented in Figure E.2 on page 245. Both figures are contained in the Appendix E. The frequency of the terms is indicated by the number in parenthesis, ranging from 1 (lowest) to 3 to 6 (highest). This shows that classes for categorization such as function, goal, action, actor, process, feature, task are often used. The classes *concept*, *constraint*, *object*, and *property* provide the classification of domain knowledge. The used relations *isA*, *aggregation*, *inheritance*, *subClassOf*, *contains*, and *isPartOf* indicate relations through which to build hierarchies in the domain. The relations *complement*, *symmetric*, *transitive*, *reflective* are used to define fundamental domain semantics. Other relations are for example *dependsOn*, *needed*, *isInvalidFor*, and *hasA*. They are used to describe dependencies and valid or invalid combinations of concepts to assess requirements.

Fifteen of the studies (45 %) provide information about individuals (m3.2.2) of the used ontologies. Individuals are extracted terms from requirements and are added to the ontology. In the primary studies, the process of term extraction and ontology population is often called respectively the formalization of requirements, requirements import, or transformation. Examples of individuals are: (i) nodes or terms of a feature model; (ii) nodes of a goal graph; (iii) specific requirements or elements from UML diagrams (for example, *requirement* related to a role *subject* impacts on *other requirement*; (iv) concrete functions and their relationships to each other; and the (v) textual values of attributes.

Thus, we can answer RQ.2.1.3.2 by stating that the majority of the DOs follow some kind of metamodel that is used to classify or group individuals and to restrict the scope of the ontology. Individuals that can be found in DOs are terms and attributes of the InRs or SRSs. These terms and attributes are extracted from the requirements during the activity *ontology population*, as was introduced in Section 2.3 on page 28.

RQ.2.1.3.3: How exactly is the DO used? Table 9.8 shows the types of modifications of DOs (m3.3.1) that we were able to find. In 15 (45 %) of the 33 approaches a DO is created, modified, or extended by new concepts and relations (first two rows). *Ontology learning* was found in 31 % of the approaches. In four (12 %) of the 33 approaches, *ontology population* was found, where extracted terms from requirements (text and models) are added to the DO as individuals. Closely related to ontology population is the group which is called *mapping of ontology concepts to requirements*, that can be found in 11 % of the approaches, where a mapping is created from concepts in the DO to relate certain requirements. Although this

[3] Tag cloud generated with tagcrowd `https://tagcrowd.com/`.

[4] OMG-ODM is the Ontology Definition Metamodel of the Object Management Group (OMG). See `http://www.omg.org/spec/ODM/` for further details.

Table 9.8: The modification of domain ontologies during RE (multiple mentions)

Modification of DO	Studies	Count	%
ontology learning	S09, S22, S33, S37, S38, S39, S41, S43, S54, S55, S67	11	31 %
ontology population	S07, S09, S50, S54	4	11 %
mapping of DO concepts to req.	S15, S16, S17, S19	4	11 %
no modification	S01, S03, S05, S12, S13, S14, S20, S36, S44, S45, S53, S61, S58, S62, SNB01, SNB07	16	49 %
Total	33 studies; rounding error = 2 %	35	100 %

mapping can be stored anywhere either in the ontology or externally, the DO concepts point to certain requirements, and thus we count this as a modification of the ontology. In 16 out of 33 (48 %) approaches, we could not find any modification of the DO.

Whether the DO is queried (m3.3.2) and what kind of information is used for querying (m3.4.2) was difficult to extract from the primary studies, as this issue is not explained explicitly. In 11 out of 33 (33 %) of the approaches, we could not identify any explicit querying (m3.3.2) of the ontology. Those approaches which used querying (22 out of 33, 67 %) access the domain knowledge, the terms and their definitions, necessary attributes for requirements, the individuals, and concepts (such as workflows or roles) without individuals.

Reasoning (m3.3.3) is used in 19 out of 33 (58 %) approaches. Attempts are made to discover the following situations and defects through reasoning: identify missing requirements, identify incomplete requirements with vague or missing information, identify redundant or overlapping requirements, and identify inconsistent requirements. Additionally, new relations between concepts or mappings from DO concepts to attributes are inferred.

RQ.2.1.3.4: Which input or output is related to DO usage? The output of any DO modification (m3.4.1) was difficult to extract from the primary studies, since this question is only answered either implicitly or else not at all. The inputs for modification are textual and model-based RE artifacts, as shown in Table 9.6 and Table 9.7. In most cases these RE artifacts are the source for the activities of *ontology learning* and *ontology population* (see Section 2.3 on page 28), which either create an ontology or in which an existing ontology is extended by the respective individuals (S01, S09, S33, S38, S39, S54, S61).

Table 9.9: The type of rules found. reas.=reasoning.

Type of rule	reas.	Studies	Count	%
map goals to requirements	✗	S07	1	5 %
SRS guidelines, SRS quality	✗	S09, SNB01	2	11 %
term extraction and DO population	✗	S33, S55	2	11 %
identify incomplete, inconsistent, or redundant, incorrect requirements	✓	S01, S16, S20, S36, S39, S44, S43, S54, S61, S62	10	53 %
domain-specific rules	✓	S17, S22, S50, S67	4	21 %
Total, rounding error = 1 %		19 studies	19	100 %

The input used to query the DOs (m3.4.2) was again difficult to extract from the primary studies, as this issue is not explained explicitly. As input we found the terms from requirements artifacts and models after pre-processing, i.e. stemmed and stop word removed terms (S03, S05, S12, S14, S16, S17, S22, S36, S55, S61, SNB07). The input for reasoning (m3.4.3) is both the ontology and the rules. In two cases (S50, S61) the rules are not contained in the ontology, but are provided separately, for example in description logic or as prolog rules. We found five different types of rules (m3.4.4:), as are shown in Table 9.9: (i) rules to map goals to requirements, which are not used for reasoning; (ii) rules to measure the quality of an SRS or to guide the creation of RE documents in terms of completeness, but which are not used for reasoning; (iii) rules for term extraction and ontology population, which is also not suited to reasoning; (iv) rules to identify incomplete, inconsistent, redundant, or incorrect requirements, which is always combined with reasoning; and (v) domain specific rules, which also incorporate business rules, and which are suited to reasoning.

The answers to RQ.2.1.3.3 (use of DO) and RQ.2.1.3.4 (input and output of ontology usage, on which see further Table 9.1) are that half of the approaches do not modify the ontology, and, if the ontology is modified, it is done in the form of ontology learning and population, and in some cases with a mapping between ontology concepts and requirements. The input used for modifications are terms from the requirements artifacts. In more than half of the approaches, ontologies are queried and reasoning is used with either domain rules or requirements quality rules.

RQ.2.1.3.5: How are artifacts modified using a DO? In several cases only implicit details about artifact modification (m3.5.1) with the use of ontologies are given by the studies. We were able to identify eight types of artifact modifications which are presented

Table 9.10: Requirements artifacts modification using a DO. (multiple mentions)

Artifact's modification	Studies	Count	%
manual revision of requirements based on defects list	S01, S13, S37, S44, S50, S43, S54, SNB01	8	18 %
manual revision of requirements based on recommendations	S12, S19, S36, S44, S61	5	11 %
guidance for manual requirements specification	S01, S14, S16, S17, S33, S38, S41, S44, S45, S43, S53, S67, SNB01, SNB07	14	31 %
automatic correction of requirements texts	S39, S45, S55, S62	4	9 %
use of correct language terms	S03, S07	2	4 %
annotation/mapping/linking of requirements with domain concepts	S05, S14, S15, S16, S17, S20, S58	7	16 %
generation of an SRS document	S09, S22, S55	3	7 %
recommendation of sentence template	S12, S43	2	4 %
Total, rounding error = 1 %	33 studies	45	100 %

in Table 9.10, and which constitute the answer to RQ.2.1.3.5. This artifact modification is in nearly all cases supported by the presented tool features. Eight approaches provide a list of requirements defects that can be used by a requirements engineer to revise requirements. Five approaches provide specific recommendations relating to the requirements which should be changed and in what way. General guidance such as the provisioning of possible features, goals, or underspecified requirements is given in 14 approaches. The automatic correction of requirements is executed in four approaches. The explicit use of correct domain terms is supported by two approaches, which are closely related to the annotation or linking of requirements with domain concepts, and these are provided by seven approaches. The generation of an SRS document (in one case according to IEEE-830-1998) from the ontology is supported by three approaches. Finally, a tool proposes a suitable template to specify a requirement in two approaches.

RQ.2.1.3.6: What are the features of a possibly existing tool related to the DO? A majority of 24 out of 33 (73 %) approaches present an individually developed tool that is used for evaluation or demonstration (m3.6.1). We follow the definition of **feature** by Kang et al. [KCH+90], who define a feature as a prominent or distinctive user-visible aspect, quality or characteristic of a software system or systems. In order to determine the provided features, we used the description in the texts as well as any available screen shots. The tool features we were able to identify (m3.6.2) are presented in Table E.6 in Appendix E, and they are grouped according to the usage categories that are explained in Section 9.3.

As some of the approaches use other types of ontologies in addition to DOs, the tool features cannot be related to the DOs usage clearly. Thus, the tool features discussed below relate rather to the use of ontologies in RE in general. The used tool features exhibit a focus on ontology learning and population and the management of ontologies. Several features support the annotation and linking of requirements with ontology concepts. Several features enable ontology individuals to be filled into requirements templates, in particular in combination with controlled natural language. Tool features that support the use of the ontology as a knowledge base are closely connected to tool features in order to review an ontology created from requirements. Most tool features concern the detection of RQAs' defects. Important supporting tool features are related to the import and export of textual requirements and models, the tool integration of requirements and ontology tools, and requirements management.

Details of the used libraries and frameworks for the implementation of the tools and the integration of other tools were given by 25 out of 33 (76 %) studies. These details are provided as a non-annotated overview in Section E.5 in Appendix E on page 248, as the technology details lie beyond the scope of this study.

RQ.2.1.3: How exactly are DOs used? Summarizing RQ.2.1.3.1 – RQ.2.1.3.6, we see a wide variety of domains supported by DOs for RE. The majority of the DOs follow some kind of metamodel in order to structure the ontology. About half of the approaches do not modify the ontology, whereas the other half do. The input used for modifications are extracted terms and attributes from the requirements artifacts. In more than half of the approaches, ontologies are queried and reasoning is used with domain rules or requirements quality rules. Requirements artifacts are modified manually on the basis of defects-list recommendations, or are otherwise guided by the DO, the automatic correction of requirements texts and use of correct language terms, as well as the linking of requirements with domain concepts. In some minor cases, SRSs documents are created, or sentence templates are recommended. Most approaches use tools.

Based on these findings, it can be said that although existing approaches use DOs in different ways, they share several common characteristics. Overall, we answer RQ.2.1.3 by stating that DOs are used to improve requirements quality in the following ways: (i) DOs are created from various knowledge sources and are populated with terms of requirements as individuals. Term extraction, word nets or other taxonomies, and similarity measures are used to identify respective classes in the ontology in order to add the requirements terms to them. (ii) The created DOs are reviewed by a requirements engineer to identify gaps or mistakes. (iii) The DOs are directly used as a knowledge base to look up domain details or to find past solutions. (iv) The taxonomy of DOs is used as a glossary to link or include ontology terms in the requirements. In particular DO classes or individuals are filled into variables of standardized natural language or templates. (v) DOs with the terms of requirements as individuals in combination with domain-specific rules and defect detection rules are used for reasoning in order to discover requirements defects. These different ways are presented in greater detail in Section 9.3.

9.2.4 RQ.2.1.4: Applicability and Maturity of Requirements Improvement Approach

The purpose of this research question is to understand the applicability and maturity of the approaches regarding existing limitations, preconditions, and the presented evaluation.

9.2.4.1 Results

RQ.2.1.4.1: Which preconditions exist for the approach? The approaches indicate one major artifact-related precondition (m4.1.1) that is the first part of the answer to RQ.2.1.4.1. Artifacts must be in the correct format, which means that images, models, tables must have been transformed into text (S13) or XMI[5] (S14), or otherwise they must contain requirements as text (S16, S17, S36, S50, S55). This precondition relates to the RE artifact which is in the focus of the approach, as is depicted in Table 9.6 and Table 9.7.

The second part of the answer to RQ.2.1.4.1 regarding the ontology preconditions (m4.1.2) consists of the following two conditions. First, the DO must be available, including their concepts and rules (S01, S03, S05, S07, S12, S15, S16, S17, S38, S44, S45, S50, S43, S53, S54, S61, S62, S67, SNB01, SNB07), or if it is not, it must be created in advance. Second, the ontology must comply with the given metamodel.

The third part of the answer to RQ.2.1.4.1 regarding the precondition related to the

[5]XML Metadata Interchange (XMI) is an Object Management Group standard used as the format for model interchange, in particular UML models.

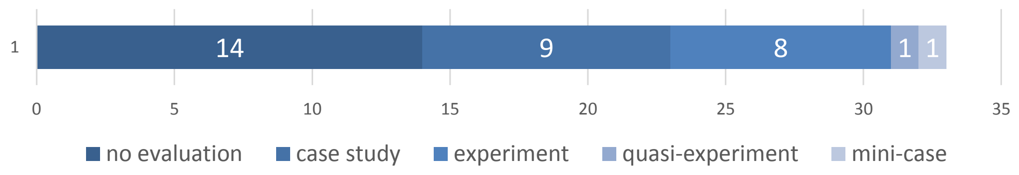


Figure 9.6: The type of presented empirical evaluation

requirements (m4.1.3) is that requirements must have the expected format and content. For example, the requirements must use the terms from the ontology (S07, S09), must contain boilerplates or templates (S19, S38, S54), must contain a feature model (S05, SNB01), contain a use case (S12, S15, S44, S43), contain an i* model (S20), or contain a goal graph (S61). This precondition also relates to the RE artifacts which are focus of the approach.

RQ.2.1.4.2: How is the DO approach evaluated? The empirical evidence of S1 – S67 is presented in [DVB+15]. Since we added SNB01 – SNB14 and we focus on approaches using a DO, the analysis of empirical evidence differs slightly. As shown in Figure 9.6 and Table 9.11 with details, we used the following categories. In 14 out of 33 (42 %) primary studies, *no evaluation* is presented, or the authors otherwise claim an evaluation without presenting any details. An evaluation as a *case study* requires the application of the approach in a real-world setting, it is described at a level of detail which other researchers can follow, and it provides the methods of execution and an objective presentation of the results. A case study is presented in 9 out of 33 (27 %) primary studies. An *experiment* comprises an experimental evaluation with a described setting where objective measures (such as count or numbers) are presented. This was the case in 8 out of 33 (24 %) primary studies. A *quasi-experiment* is an experiment without a control group, which was used by 1 out of 33 (3 %) primary studies. A *mini-case* is when the approach is applied to a continuous example, which was used by 1 out of 33 (3 %) primary studies.

We can answer RQ.2.1.4.2 (see Table 9.1) by stating that most of the approaches are evaluated with either a case study or an experiment. However, there is a significant number of approaches (42 %) that were not evaluated. Regardless of the type of evaluation, most of the studies (73 %) provide a real example to explain or demonstrate the presented approach.

9.2.4.2 Analysis and discussion

Since the approaches focus on requirements improvement, the creation of a DO is naturally not in focus for them, with the exception of a few cases. Some approaches are able to create a

Table 9.11: The type of empirical evaluation.

Type of evaluation	Studies	Count	%
case study	S01, S19, S22, S37, S44, S50 S61, S62, SNB01	9	27 %
experiment	S03, S12, S13, S16, S17, S33, S54, S55	8	24 %
quasi-experiment	S36	1	3 %
mini-case	S41	1	3 %
no empirical evaluation	S05, S07, S09, S14, S15, S20, S38, S39, S45, S43, S53, S58, S67, SNB07	14	42 %
Evaluation differs from Dermeval et al. for the following studies: S03, S19, S22, S41			

DO or otherwise extend an existing one. Obviously, the approach and the ontology are coupled in such a way that the approach requires certain ontology classes and relations. Moreover, the approach is designed for certain requirements artifacts following defined formats. The presented evaluation with case studies and experiments indicates a maturity that promotes their application in practice.

9.3 Domain Ontology Usage Patterns

The results presented in Section 9.2 show several recurring strategies in the usage of DOs among the investigated DO-based approaches. We synthesized these strategies to recurring *domain ontology usage patterns*, as is presented in this section. These patterns refine the answer to RQ.2.1.3 and constitute the answer to RQ.2.1.5 by relating the improved RQAs to the DO usage strategies. Each of the DO usage patterns described in this section shows a usage strategy of a DO in combination with RE artifacts, the information flow, and necessary activities in order to achieve an improvement of InRs's and SRSs' quality attributes.

9.3.1 Pattern Basics

We use the term *pattern* in accordance with to the definition of Gamma et al. [Gam08], who define that a software design pattern “names, abstracts, and identifies the key aspects of a common design structure that make it useful for creating a reusable object-oriented design”. Our focus is on the strategies to improve requirements quality instead of the “object-oriented design”.

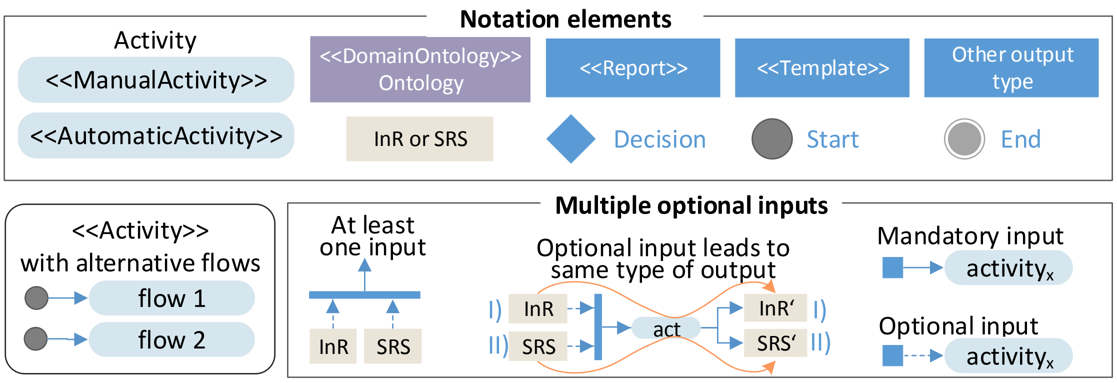


Figure 9.7: The used notation elements, based on UML 2.0

The patterns are documented by using the semi-formal notation of UML 2.0 activity diagrams [RBJ04] with three enhancements. Figure 9.7 shows the notation elements we used and our three necessary enhancements to the UML definition: (i) The UML standard only allows alternative flows with decision elements, which would overload our patterns. Therefore, we decided to introduce alternative flows by using a super-activity containing multiple ($n \geqslant 2$) start nodes connected to sub-activities. (ii) We also indicated multiple optional inputs to an activity by a thick line, where at least one input is mandatory. (iii) Each activity may consume *input data* and each activity produces *output data*. Activities are chained through *input and output data*. Alternative input types denoted with roman numbers (I) and II)) lead to output types denoted with the same number, as is shown in Figure 9.7 with the example I) InR→act.→InR' I), and II) SRS→act.→SRS' II), and where act. means activity.

Below, we explain each of the identified patterns in detail. The descriptions comprise the pattern's *name* (avoiding any abbreviation), the improved *RQAs*, its *input* and *output types* and a detailed textual description of the performed activities. Table 9.12 shows the patterns used in the studies. The patterns contain several options, , that are indicated by super-activities with multiple start nodes. For reasons of clarity, the patterns are modeled in one single execution flow only, although there are patterns that require multiple executions, such as the review domain ontology pattern. This pattern is iterated multiple times until an acceptable quality of the created DO is reached. We name the patterns following the convention: `<verb>` `<object>` *pattern*, except for the ontology learning and population pattern, as they are commonly accepted terms. In the next two subsections, we first present the basic patterns to create and populate DOs, and we then present the patterns to improve requirements quality with increasing complexity.

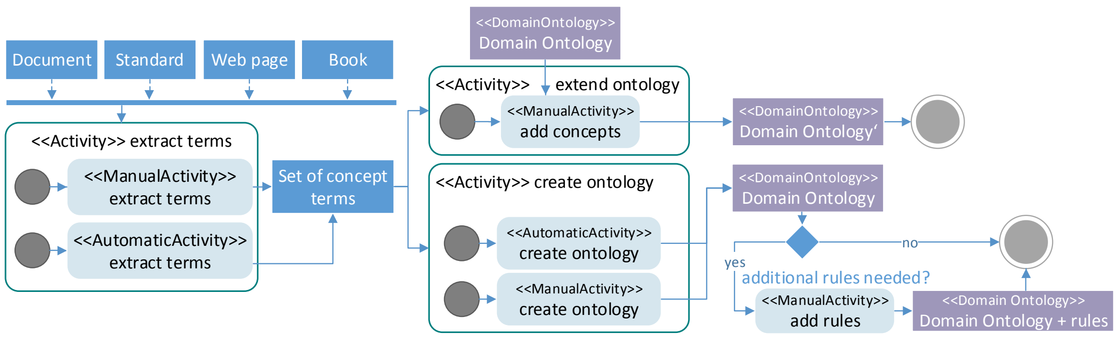

Document
Standard
Web page
Book
<<Activity>> extract terms
<<ManualActivity>> extract terms
<<AutomaticActivity>> extract terms
Set of concept terms
<<DomainOntology>> Domain Ontology
<<Activity>> extend ontology
<<ManualActivity>> add concepts
<<DomainOntology>> Domain Ontology'
<<Activity>> create ontology
<<AutomaticActivity>> create ontology
<<ManualActivity>> create ontology
<<DomainOntology>> Domain Ontology
yes
no
additional rules needed?
<<ManualActivity>> add rules
<<Domain Ontology>> Domain Ontology + rules

Figure 9.8: The ontology learning pattern

9.3.2 Creating or Modifying a Domain Ontology

The ontology learning pattern, as depicted in Figure 9.8, is a basic pattern with which either to create an ontology, in particular a DO, from scratch or to edit an existing one. The change and extend-activities in this pattern focus on ontology classes and axioms. The *inputs* are any kind of documents, standards, web pages, or books. The ontology learning produces a (domain) ontology as *output* that possibly contains rules. *Description:* The terms from input documents are extracted either automatically or manually as a set of concept terms used in the domain. These are then added as new ontology classes during ontology learning. The term extraction activities utilizes artificial intelligence techniques, information retrieval (IR) algorithms, machine learning (ML), or NLP, rule based term extraction, or part of speech (POS) tagging. Our impression was that these approaches require a great effort to determine the correct relations to existing ontology classes. Axioms (rules) are created and added manually to the ontology.

9.3.3 Adding Requirements Terms as Individuals to a Domain Ontology

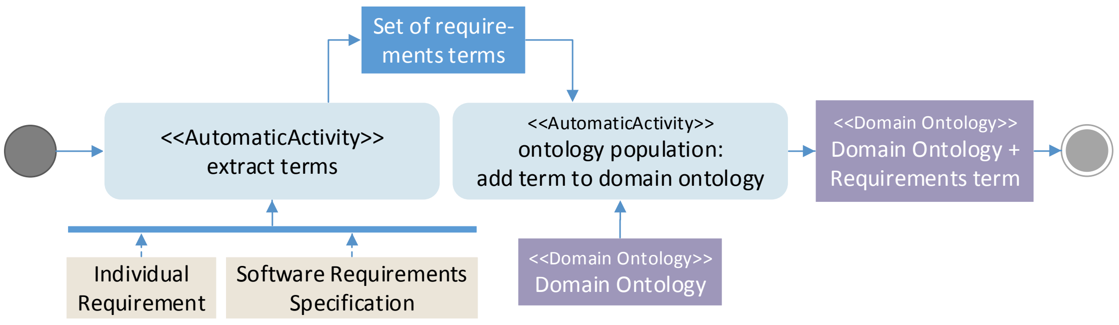
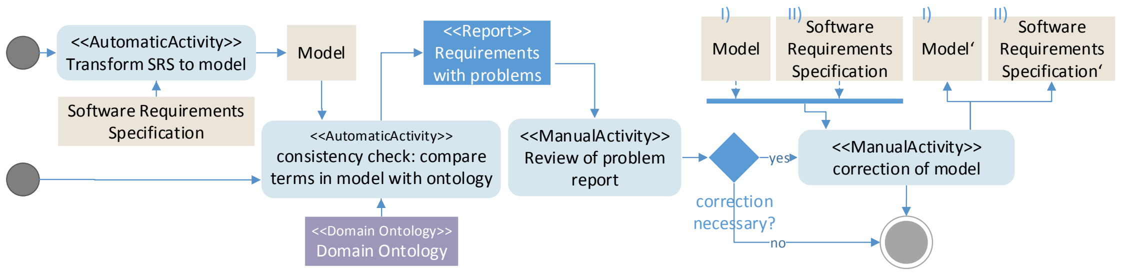

Set of require-ments terms
<<AutomaticActivity>> extract terms
<<AutomaticActivity>> ontology population: add term to domain ontology
<<Domain Ontology>> Domain Ontology + Requirements term
Individual Requirement
Software Requirements Specification
<<Domain Ontology>> Domain Ontology

Figure 9.9: The ontology population pattern

The ontology population pattern, as depicted in Figure 9.9, allows one to add requirements terms as individuals to a DO. A populated DO is the precondition for other DO usage patterns, in order to identify wrong or missing requirements with patterns presented below. The pattern uses as *input* InRs or SRSs and an existing DO. As *output* it produces the populated DO with requirements terms. *Description:* Key terms in InRs or SRSs, such as named entities or technical terms, are extracted in the activity *extract terms* by IR algorithms, often in combination with NLP techniques, rule based term extraction, or POS algorithms. These terms are then added as individuals to the classes in the DO. Our impression during the analysis of the approaches was that the identification of the correct ontology class to add individuals is a challenging and complex NLP task.

9.3.4 Consolidating Requirements



<<AutomaticActivity>> Transform SRS to model
Software Requirements Specification
Model
<<Report>> Requirements with problems
<<AutomaticActivity>> consistency check: compare terms in model with ontology
<<Domain Ontology>> Domain Ontology
<<ManualActivity>> Review of problem report
correction necessary?
yes
no
I) Model
II) Software Requirements Specification
I) Model‘
II) Software Requirements Specification‘
<<ManualActivity>> correction of model

Figure 9.10: The consolidate requirements pattern

The consolidate requirements pattern, as depicted in Figure 9.10, allows one to compare the details of a model to predefined details in the DO. *Description:* Details of an UML model such as a sequence chart are compared to terms in the DO. An SRS can be transformed into a model that will be consolidated. Necessary *input* is the DO, a model, or an SRS that contains models. As *output*, a report with parts of the model that are inconsistent with the DO is created. A requirements engineer then uses the report to improve the model or the SRS. The RQA InR.consistent and SRS.consistent are improved.

9.3.5 Reviewing a Domain Ontology

The review domain ontology pattern, as depicted in Figure 9.11, allows one to identify wrong or missing requirements by a manual review of the populated DO. *Description:* A DO is populated with requirements terms as individuals by executing the ontology population pattern. Then a manual review of the populated DP is performed by a requirements engineer or a domain expert. In this review, two types of defects are detected: First, misplaced individuals – these are requirements terms that are populated to incorrect DO classes – indicate weaknesses

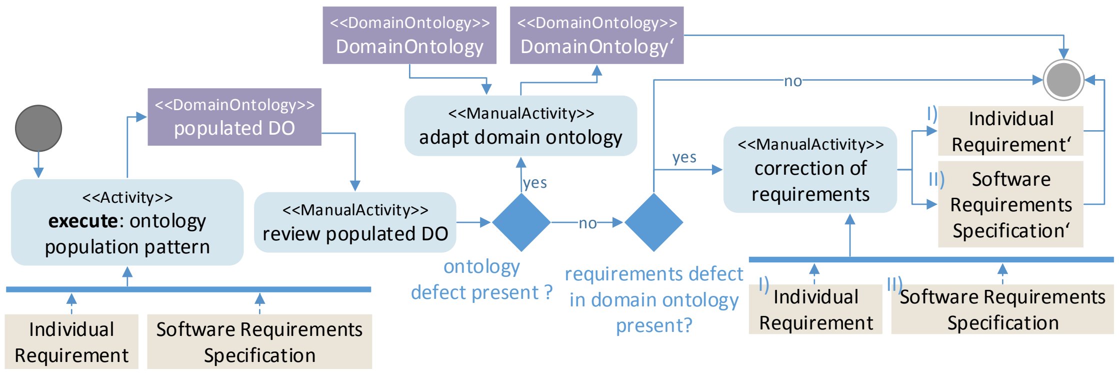

<<DomainOntology>>
DomainOntology
<<DomainOntology>>
DomainOntology'
<<DomainOntology>>
populated DO
<<ManualActivity>>
adapt domain ontology
<<Activity>>
execute: ontology population pattern
<<ManualActivity>>
review populated DO
yes
no
ontology defect present ?
requirements defect in domain ontology present?
yes
no
<<ManualActivity>>
correction of requirements
I) Individual Requirement'
II) Software Requirements Specification'
Individual Requirement
Software Requirements Specification
I) Individual Requirement
II) Software Requirements Specification

Figure 9.11: The review domain ontology pattern

in the documentation of the requirements, such as synonyms or wrong terms. Second, DO concepts with either no or missing individuals show missing or underspecified requirements. Both these kinds of defect have their origin in either the requirements or in the DO. When the requirements engineer or domain expert detects mistakes in the DO, s/he corrects the unpopulated DO and executes the whole review domain ontology pattern again. When the requirements engineer or domain expert detects misplaced individuals or concepts with no or missing individuals, s/he revises, adds, or changes the requirements. In summary, the manual review is based on the visualization of the assignment of extracted terms to DO concepts. However, conflicts of InRs to other InRs cannot be detected on the basis of extracted and populated requirements terms. Some approaches are able to generate an improved SRS after the modification. The *input* is InR or SRS and a populated DO. The *output* are modified InR or SRS, or a modified DO. The populated DO is obtained by executing the ontology population pattern. The improved *RQA* are InR.unambiguous, InR.complete, SRS.consistent, SRS.complete.

9.3.6 Using Glossary Terms in Requirements

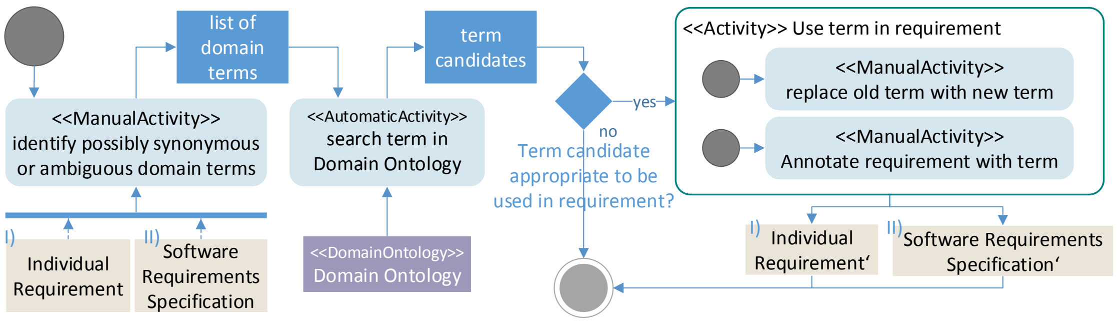
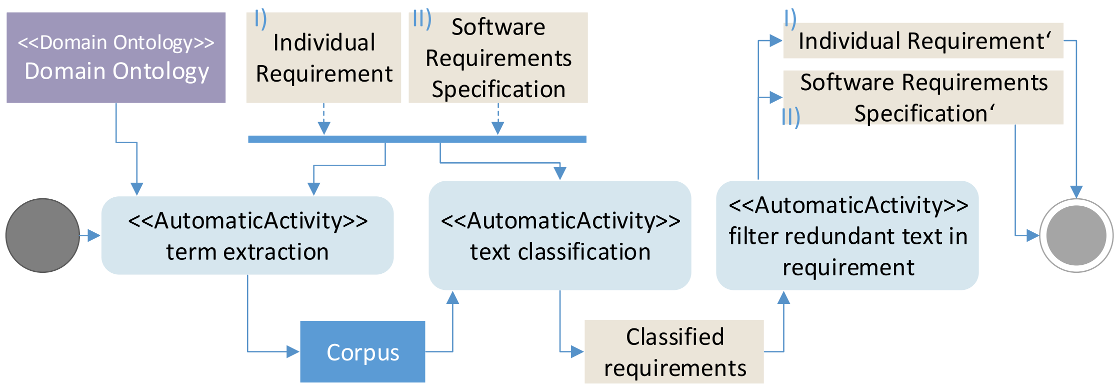

list of domain terms
term candidates
<<Activity>> Use term in requirement
<<ManualActivity>>
replace old term with new term
<<ManualActivity>>
Annotate requirement with term
<<ManualActivity>>
identify possibly synonymous or ambiguous domain terms
<<AutomaticActivity>>
search term in Domain Ontology
yes
no
Term candidate appropriate to be used in requirement?
I) Individual Requirement
II) Software Requirements Specification
<<DomainOntology>>
Domain Ontology
I) Individual Requirement'
II) Software Requirements Specification'

Figure 9.12: The use individuals as glossary terms pattern

The use individuals as glossary terms pattern, as depicted in Figure 9.12, aims to use the standard domain terms in the requirements. *Description:* Possibly synonymous or ambiguous terms are identified manually within the requirements. Standard domain term candidates from the DO are presented to the requirements engineer. S/he can then decide to use the term in the requirements, or alternatively, to annotate the requirements term with the domain term. The *input* is a DO and InRs or SRSs. The *output* is an improved InRs or SRSs with DO terms or annotations in the requirements. This pattern improves the *RQA* InR.unambiguous.

9.3.7 Removing Redundant Parts of Requirements



<<Domain Ontology>> Domain Ontology
I) Individual Requirement
II) Software Requirements Specification
<<AutomaticActivity>> term extraction
<<AutomaticActivity>> text classification
<<AutomaticActivity>> filter redundant text in requirement
Corpus
Classified requirements
I) Individual Requirement‘
Software Requirements Specification‘ II)

Figure 9.13: The filter redundant requirements text pattern

The filter redundant requirements text pattern, depicted in Figure 9.13, aims to remove redundant parts of requirements text. *Description:* A corpus is created from a DO and requirements text. A corpus is a large collection of natural language text used for linguistic analysis. With a text-classifier, redundant parts of text in requirements are identified, which can then be eliminated automatically. The principal assumption of the pattern is that the removal of redundant text leads to fewer different ways of interpreting of the requirement. However, the requirement can still be ambiguous. The *input* is a DO acting as source for a corpus and the InR or SRS respectively. The *output* is an improved InR and SRS. The improved *RQA* is InR.unambiguous.

9.3.8 Using the Domain Ontology as Knowledge Base

The use domain ontology as knowledge base pattern, as depicted in Figure 9.14, describes the use of the DO as a knowledge base to search for past solutions, relations between domain concepts, or hints for additional requirements, in particular NFR. The term *past solutions* refers to problems and their successful solutions in an application domain in the context of

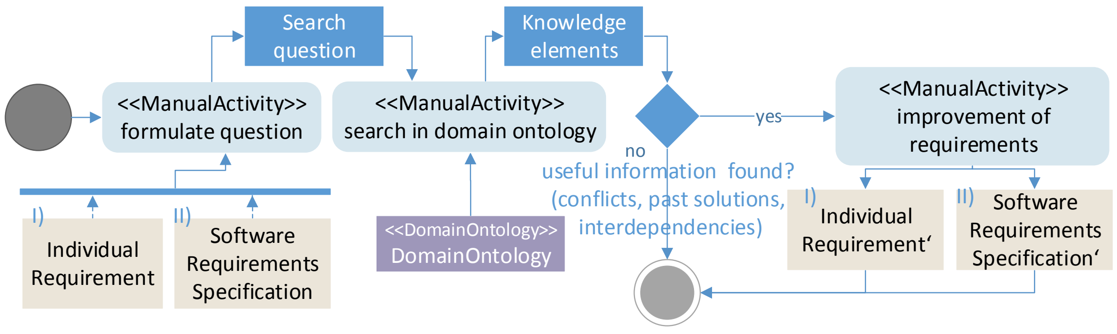

Search question
Knowledge elements
<<ManualActivity>> formulate question
<<ManualActivity>> search in domain ontology
yes
no
useful information found? (conflicts, past solutions, interdependencies)
<<ManualActivity>> improvement of requirements
I) Individual Requirement
II) Software Requirements Specification
<<DomainOntology>> DomainOntology
I) Individual Requirement‘
II) Software Requirements Specification‘

Figure 9.14: The use domain ontology as knowledge base pattern

other projects, stored in a DO. *Description:* Based on existing requirements, the requirements engineer asks him/herself a question or thinks about an unclear relationship. S/he searches the DO with keywords of the question and identifies relevant knowledge elements. Afterwards s/he can add the gained knowledge to the requirements. The *input* is a DO, InRs, or a SRS. The *output* is an improved InR or SRS. The pattern improves RQA SRS.complete, SRS.consistent, InR.complete, InR.correct, InR.unambiguous, InR.consistent, and InR.traceable. Although the requirements engineer profits from the use domain ontology as knowledge base pattern in manifold ways, the support for the RQA SRS.consistent is limited. The ontology as knowledge base neither supports the identification of duplicate requirements in the SRS, nor does it support the assurance that the same term is used for the same item in all requirements.

9.3.9 Completing Requirements Templates with Individuals

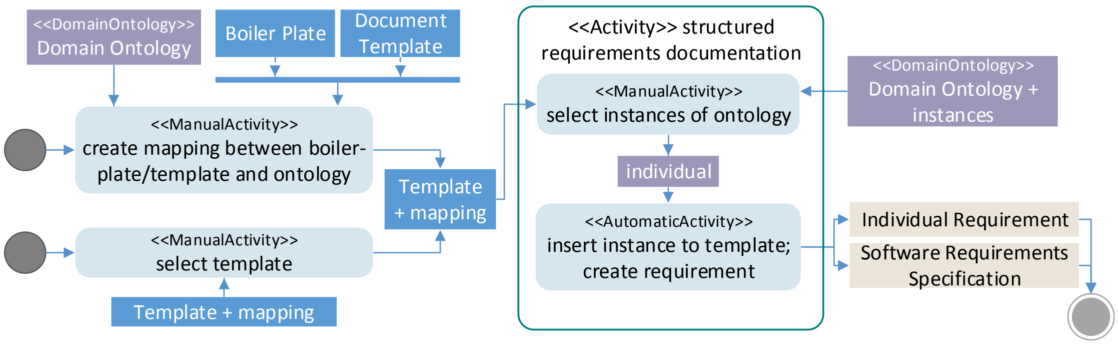
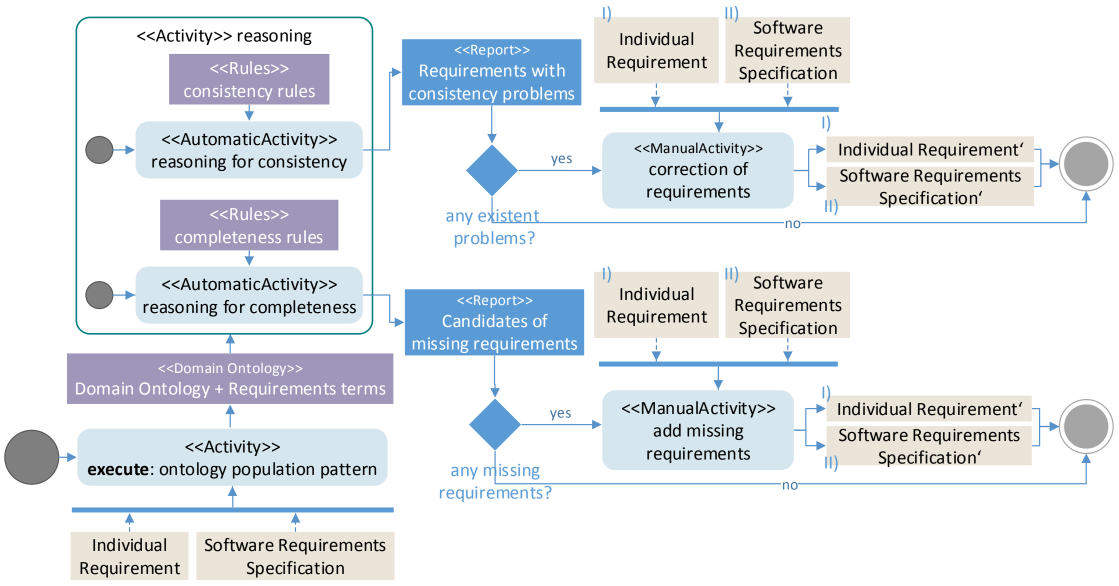

<<DomainOntology>> Domain Ontology
Boiler Plate
Document Template
<<ManualActivity>> create mapping between boiler-plate/template and ontology
Template + mapping
<<ManualActivity>> select template
Template + mapping
<<Activity>> structured requirements documentation
<<ManualActivity>> select instances of ontology
individual
<<AutomaticActivity>> insert instance to template; create requirement
<<DomainOntology>> Domain Ontology + instances
Individual Requirement
Software Requirements Specification

Figure 9.15: The complete requirements template pattern

The complete requirements template pattern, as depicted in Figure 9.15, describes the use of individuals or attributes of a DOs at predefined positions in CNL such as boilerplates or document templates. *Description:* A mapping is created between variables at predefined positions in existing boilerplates (CNL) or document templates to DO classes. During the documentation of requirements, the requirements engineer chooses individuals or attributes from the ontology and inserts them into the text-based requirements. The unambiguity and consistency of InRs is improved, as standard terms in combination with templates are used. The correctness is improved as only validated individuals or attributes from the DO can be used to complete the templates. The *input* for this pattern are a populated DO, the CNL, or the document template. The *output* are individuals of the DO, an improved InR or SRS. The pattern improves the *RQAs* InR.correct, InR.consistent, InR.unambiguous.

9.3.10 Deducting Consistency or Completeness Defects

The reason about defects pattern, as depicted in Figure 9.16, describes the deduction of requirements defects through a reasoner. It is the most powerful usage of a DO in RE to deduce inconsistent or missing requirements. As *input*, the pattern requires a populated DO with consistency rules or completeness rules. *Description:* With the help of consistency or completeness rules, a reasoner is able to create a report of inconsistent or missing requirements. The requirements engineer manually modifies the requirements appropriately. The *output* is a report with requirements problems, candidates, or improved InR or SRS. The pattern uses ontology population to provide the populated DO, containing terms and details of the requirements. The improved *RQAs* are SRS.consistent and SRS.complete.



<<Activity>> reasoning
<<Rules>> consistency rules
<<AutomaticActivity>> reasoning for consistency
<<Rules>> completeness rules
<<AutomaticActivity>> reasoning for completeness
<<Domain Ontology>> Domain Ontology + Requirements terms
<<Activity>> execute: ontology population pattern
Individual Requirement
Software Requirements Specification
<<Report>> Requirements with consistency problems
any existent problems?
yes
no
I)
II)
<<ManualActivity>> correction of requirements
Individual Requirement'
Software Requirements Specification'
<<Report>> Candidates of missing requirements
any missing requirements?
<<ManualActivity>> add missing requirements

Figure 9.16: The reason about defects pattern

9.3.11 Infrastructure Needs of DO Usage Patterns

All patterns can be applied manually to requirements. An automation of some or all activities is time-saving. In order to automate some activities, some infrastructure by tool support is necessary. Table 9.12 shows in the column titled *infrastructure* the patterns' need for infrastructure in automatic activities or when they are not applied manually. An overview of such tools is given in Section E.5 in the Appendix on page 248. In the following section, we discuss this infrastructure need in brief. The information regarding the infrastructure needs is helpful to choose the DO usage pattern and to decide on the intended automation for a method adaptation. Both the ontology learning and the ontology population pattern requires NLP and machine learning techniques to extract terms and relations to build the ontology and to find corresponding concepts. We rate this as a high infrastructure need due to its complexity and its combination of several tools used in the primary studies. The consolidate requirements pattern requires term extraction techniques and techniques to search requirements terms within the DO. We regard this as a medium infrastructure need. The automated activity of the review domain ontology pattern is based on the ontology population pattern, and thus it has high infrastructure needs. The use individuals as glossary terms pattern requires an automated search in order to provide the requirements engineer with appropriate and linked terms from the DO during the specification. We rate this as a low infrastructure need. The filter redundant requirements text is also based on NLP and machine learning, and thus has high infrastructure needs. The use domain ontology as knowledge base pattern has no automation capabilities, and requires an ontology visualization only. Thus, we rated this as a low infrastructure need. The complete requirements template pattern requires templates, the templates' mapping, and search functionality to find appropriate terms from the DO. We rated this as a medium infrastructure need. Finally, the reason about defects pattern uses the ontology population pattern, uses rules and an automated reasoner. We rated this as a high infrastructure need.

9.3.12 Summary

Table 9.12 shows the association between the DO usage patterns and the primary studies. The inverse representation, showing the supported RQA and the DO usage patterns of each approach, is shown in the right half of Table 9.3. Columns 4 to 10 indicate the improved RQA with ✗. Columns 11 to 19 contain the assigned DO usage patterns. It is not surprising that studies following the ontology population pattern mostly also use reasoning (S16, S17, S22, S36, S39, S50, S61, SNB07).

Table 9.12: Domain ontology usage pattern related to primary studies. Infrastr. = infrastructure needs: ● low, ●● medium, ●●● high

Pattern	**Study**	**Infrastr.**	**Count**	**%**
ontology learning pattern	S09, S33, S36, S37, S38, S41, S43, S54, S55	●●●	9	14 %
ontology population pattern	S13, S16, S17, S22, S36, S39, S50, S61, S58, SNB07	●●●	10	15 %
consolidate requirements pattern	S37, S53	●●	2	3 %
review domain ontology pattern	S09, S15, S39, S41, SNB07	●●●	5	8 %
use individuals as glossary terms pattern	S03, S05, S07, S14, S15, S20, S38, S44, S54, S58	●	10	15 %
filter redundant requirements text pattern	S55	●●●	1	2 %
use domain ontology as knowledge base pattern	S01, S09, S13, S33, S38, S45	●	6	9 %
complete requirements template pattern	S01, S07, S12, S13, S19, S44, SNB01	●●	7	11 %
reason about defects pattern	S01, S09, S16, S17, S22, S36, S39, S43, S50, S54, S61, S62, S67, SNB01, SNB07	●●●	15	23 %
9 patterns	33 studies		65	100 %

In nine approaches a DO is created, and in ten approaches the terms of requirements are added to a DO as individuals. In two approaches behavioral UML models are compared to individuals of the DO and are thus consolidated. In five approaches the created or populated DO is manually reviewed in order to find requirements problems. Ten approaches use a DO as a glossary and link or include ontology terms into requirements. One approach uses the DO as a corpus for extensive language processing to remove redundant parts of the requirements text. The DO is used as a knowledge base to look into domain details in the case of six approaches. Seven approaches use the DO in combination with standardized natural language to fill in variables in boilerplates or templates. Fifteen approaches use reasoning capabilities to identify requirements defects.

In summary, as an answer to RQ.2.1, we can see that the most prominent usage of DO is reasoning about defects (23 %), its use as glossary to use or link ontology terms in requirements (15 %), and its use to complete requirements templates and controlled natural language with ontology terms or values (11 %). The reasoning capabilities of ontologies in combination with the relations of domain knowledge and the opportunity to specify axioms are the main motivation behind using them.

Even if two approaches follow basically the same pattern, the approaches vary in certain details that are not captured by the pattern. An example of such a variance are the differences in the use of NLP techniques and algorithms.

9.4 Discussion

In summary we have seen that the ontology types (RQ.2.1.1) *DO*, *RE ontology* and *requirements ontology* are used in RE. Nearly half of the approaches use a formalized domain knowledge contained in DOs, indicating the demand for such knowledge in RE. The majority of approaches focus on natural language requirements, which confirms its importance in RE (RQ.2.1.2). Models in RE detail natural language specifications in a more formal or semi-formal way. As the approaches also largely support model-based artifacts, this indicates the need for both text and models.

The use of some DO usage patterns improves more than one RQA, as is shown in an overview in Table 10.1 in Chapter 10. In particular the approaches that use reasoning support improve more than one RQA (S01, S09, S16, S17, S22, S36, S54, S67, SNB01, SNB07, as is shown in Table 9.3). These approaches use additional rules to address more RQAs. For example they use rules for completeness and add other rules to improve correctness. An explanation of this effect is that once a basic infrastructure to realize the approach is present, additional rules for checking completeness, consistency, ambiguity and consistency can be realized with a low level of effort. From our point of view, the approaches described the activities of ontology learning, ontology population, and the algorithms for term extraction (such as NLP, POS-tagging) as challenging tasks. Requirements can be improved with regard to completeness, correctness, unambiguity, consistency, and traceability with the use of DOs. DOs are used (RQ.2.1.3) in nine different ways, as are described by the DO usage patterns in Section 9.3.

The identified patterns partially confirm the results of Dermeval et al. who identified the following three main benefits of ontology usage in RE: 1) The reduction of ambiguity, inconsistency and/or incompleteness of requirements. These RQAs are improved by the use of a standardized terminology and the reasoning for missing requirements. In particular the use individuals as glossary terms, use domain ontology as knowledge base, and the reason about defects pattern support such improvements; 2) Support for domain knowledge representation to guide requirements elicitation. The requirements elicitation is supported by the use of formalized domain knowledge in the use DO as knowledge base pattern. Requirements engineers are supported by such knowledge to prepare interviews and gain insights to the important domain concepts; and 3) Assistance in requirements management and evolution. We did not focus on this aspect in analyzing with the DO usage patterns.

The following paragraph discusses each of the DO usage patterns except for the ontology learning and ontology population patterns. The use of the DO to consolidate requirements is novel, but it requires an exact description in DO that relates to the requirements. It is questionable if DO contains such knowledge, or if it only contains such knowledge in cases where this kind of use is intended from the beginning. Such use supposes a high quality of the DO. The idea to extract terms from requirements, populate them to an DO and review the populated DO afterwards is novel, though it is similar to documenting review techniques. Such approaches can be seen as a semi-automated verification. The use of glossaries is well known in the RE domain and thus, it is no new idea to use the taxonomy part of ontologies for this purpose. However, the presence of relations and axioms within DOs are beneficial for requirements engineers, as they see important connections to other concepts and restrictions which they have missed. To categorize and filter text requirements is much more specific to NLP techniques than it is specific to RE. Also, the content of DO needs to contain natural language aspects. The use of DO as a knowledge base is very common, as this is one of the arguments why ontologies have emerged. For RE this is also beneficial, as it is a universal way for requirements engineers to access and use domain knowledge for various aspects of requirements. The use of CNL and templates in RE is also common. However, the use of DOs to feed valid attributes into boilerplates and templates allows an increased correctness and completeness among the specified requirements. Obviously, however, this benefit depends on the quality of the DO that is used. The use of reasoning and inference to deduce requirements defects is novel, and another reason for the emerge of ontologies. Reasoning requires specialized rules and has a high demand on the scope of the DO. The DOs used for reasoning need to match the scope of the requirements, otherwise the reasoner will detect irrelevant requirements defects.

Although the principles of DO usage are not new, we are convinced that the abstracted knowledge about *DO usage* in the form of patterns provides a valuable contribution to the RE community. In particular the condensed usage principles of DOs can be used to classify and compare present and future approaches. Certainly the DO usage patterns are neither complete nor precise at a detailed level. Thus, some approaches might show deviations in details, which cannot be captured by the DO usage patterns. If we consider the whole RE process, there might also be more cases for DO usage patterns. For example, it could be beneficial to provide a list of questions based on DOs for a requirements engineer preparing interviews. It could also be interesting to limit the DO to the scope of the to-be system in order to reduce the complexity for requirements validation. It would seem further interesting to look at other usage strategies for ontologies in general to adapt them to the RE domain. For example, in cancer research and Wikipedia, ontologies are used to formally describe the semantics of values, which could be interesting for data modeling in RE activities. This is yet not captured as a usage pattern.

The maturity of the approaches (RQ.2.1.4) seems to have reached a level where they can be applied in practice, in particular in supporting widely used requirements management tools.

9.4.1 Related Works

Other literature reviews of ontologies in RE The snowball search revealed one other interesting recent systematic review that investigates the roles of ontologies in RE [VRM16]. These authors intend to broaden the understanding of the roles in which ontologies are used in RE. They analyze 60 articles, of which six[6] articles have already been considered in Dermeval et al. [DVB+15]. Nineteen articles are older as 2007, which was exclusion criteria 4 in Dermeval et al. The search terms of Valaski et al. and Dermeval et al. are similar, whereas the most significant difference between them is that Dermeval et al. restricts the search to

[6]One additional paper is an earlier work of an article included in Dermeval et al.

the term *software engineering*. From the selected papers in Valaski et al. 19 are relevant to this work according to their title and abstract. We intend to analyze them as a part of future work. We have based our work on Dermeval et al. for two reasons. First, Dermeval et al. provide inclusion and exclusion criteria which are missing in Valaski et al., and thus the former authors provide a more reliable paper selection. Second, Dermeval's SLR provides more detailed research questions and more details of the analyzed studies.

There is another related work on the use of ontologies in RE by Castañeda et al. [CBCG10]. These authors discuss that ontologies can be used in RE to reduce the negative effects of ambiguous and incomplete requirements, insufficient specifications, and dynamic and changing requirements. Castañeda et al. present a classification of the ontologies' content in three classes, which can be confirmed by the example of several approaches: Ontologies describe (i) requirements specification documents, (ii) formally representing requirements, and (iii) formally representing application domain knowledge. We adapted this classification in our work, as was discussed in subsection 9.2.1. The referenced publications do not overlap with Dermeval et al. Only six articles are published later than 2007, which is Dermeval's inclusion criterion 4.

9.4.2 Threats to Validity

There is no clear set of validity threats for literature reviews. Kitchenham et al. [KC07] describe bias, internal and external validity as the main quality concepts for any empirical study. Internal validity tries to minimize bias and prevent systematic errors caused by the design and conduct of the study, while external validity refers to the generalizability and applicability of the study beyond itself.

Internal validity: According to [RH09], internal validity aims to ensure that the collected data enables researchers to draw valid conclusions, and in particular to minimize the effects of unknown factors that influence an investigated factor. One threat to the internal validity of our study is that the data was extracted by the author of this thesis, who had a large influence on the results. Mistakes could have been made during the extraction, leading to wrong conclusions. We mitigated this threat through four actions: (1) the definition, refinement and review of the presented GQM scheme with several iterations between two researchers; (2) by revisiting the primary studies during the analysis and synthesis, whenever we noticed discrepancies or unclear extraction information; (3) through continuous discussion with the supervisor of this thesis; and (4) the publication of the extraction spreadsheet. A second threat to the internal validity of our study is that we missed articles pertaining to our research question published between October 2013 (deadline of study selection of [DVB+15]) and February 2015 (publication date of [DVB+15]), which cannot be found with the snowball search. We can only mitigate this shortcoming by the repetition of a fully-fledged search with the original parameters of [DVB+15] for this period. However, we consider this risk as a minimal one. For even if we missed some relevant approaches, the results and the DO usage pattern are unlikely to change much because of the small changes the snowball articles impacted on our work. A third threat is that subjective decisions may have occurred during the assessment concerning the inclusion and exclusion criteria. In order to minimize selection and extraction mistakes, we performed the analysis process iteratively and the author and the supervisor regularly discussed the analysis results. Through this approach we tried to minimize the personal bias. A fourth threat is a fuzzy classification of the ontologies, as the description of the knowledge contained in ontologies was not always very clear, and mixed types do exist. As the classification used is largely in line with Castañeda et al. [CBCG10] we assume that the main classes are correct and that our classification is good enough to formulate the DO usage patterns.

External validity: According to [RH09], external validity refers to the generalizability and applicability of this study beyond itself. First, the proposed patterns might not cover all details of the investigated approaches. We minimized a potential error here by discussing the patterns in a post-graduate seminar with four doctoral students and one supervisor. We want to further mitigate this risk to external validity by the application of the DO usage patterns to a SRS of a real project as a part of future work. Second, we minimize the risk that the findings can be generalized in a wrong way through the detailed description of our primary study scope and the related research questions.

9.5 Chapter Summary

In this literature review, we extended the existing SLR of Dermeval et al. [DVB+15] to understand how domain ontologies are used in RE to improve requirements quality. Thirty-three studies were finally included, which improve requirements quality with domain ontologies. The attributes completeness, correctness, unambiguity, consistency and traceability of InRs, as well as the attributes completeness and consistency of SRSs, can all be improved with the use of domain ontologies. These requirements quality criteria are improved by nine different strategies that we described as domain ontology usage patterns, using the notation of UML activity diagrams. The usage strategies cover the following aspects: (i) the creation of a domain ontology; (ii) its population with the terms of requirements, which are obtained by natural language processing; (iii) the use of consolidating requirements in models against a reference in the domain ontology; (iv) the manual review of learned or populated domain ontologies to identify defects in requirements; (v) its use as a glossary to link or include ontology terms in requirements; (vi) its use as corpus to remove the redundant text of requirements; (vii) its use as a knowledge base to look into domain details and to find past solutions; (viii) to fill controlled natural language or templates with ontology classes or individuals; and (ix) its use of reasoning to discover requirements defects. An overview of the tool features available in the studies' prototypes further shows at a high level of abstraction how the domain ontology usage patterns are implemented in practice.

10 Treatment Design - A TORE Extension to Improve Requirements Quality

This chapter presents the TOREOnto as the application domain-specific adaptation of TORE, on the basis of the results of the literature review in Chapter 9. In Section 10.1 the DO usage framework is presented which provides the basis for TOREOnto and shows the possible application of DO usage patterns in TORE's DPs. In Section 10.2 TOREOnto is presented as one specific instance of TORE which is the solution for design problem 2. TOREOnto defines how requirements engineers use domain knowledge at TORE's specification levels in order to improve requirements quality. In Section 10.3 the specific TOREOnto is discussed. Finally, this chapter is summarized in Section 10.4.

10.1 Domain Ontology Usage Framework

Table 10.1 shows in the form of a summary the contribution to requirements quality of each of the seven DO usage patterns. The patterns ontology learning and ontology population are omitted since they do not directly contribute to requirements quality. Instead, they are used as a part of other patterns, such as the reason about defects pattern and the review domain ontology pattern.

Principally, the use of DOs is possible and beneficial in each of TORE's DPs. A possible utilization of the DO usage patterns depends on the pattern input and the used artifact type in each respective DP. Some artifacts such as models or prototypes cannot be used in combination with all patterns. Table 10.2 shows the possible ontology usage on each of TORE's DPs. All cases in which a pattern cannot be applied are discussed in the following section.

Table 10.1: Domain ontology usage patterns related to requirements quality

Pattern	InR.complete	InR.correct	InR.unambiguous	InR.consistent	InR.traceable	SRS.complete	SRS.consistent
consolidate requirements pattern				✗			✗
review domain ontology pattern	✗		✗			✗	✗
use individuals as glossary terms pattern			✗				
filter redundant requirements text pattern			✗				
use domain ontology as knowledge base pattern	✗	✗	✗	✗	✗	✗	✗
complete requirements template pattern		✗	✗	✗			
reason about defects pattern						✗	✗

10.1.1 DO Usage Patterns on TORE's Abstraction levels

The following discussion of the reasons why certain DO usage patterns cannot be applied in some DPs is structured in accordance with TORE's abstraction levels.

Goal & Task Level: *Stakeholders* are documented with a persona, being described by attributes such as age, knowledge, needs, frustrations, ideal features, and a description. As persona descriptions typically contain rather short and no detailed text, the filter requirements text pattern cannot be applied. *Stakeholders' Goals* are either documented as a persona's attribute or as goal graphs (see GORE on page 258 for further details). As there is no template used to document the stakeholders' goals, the complete requirements template pattern cannot be applied.

Domain Level: In case of the DP *Domain Data* on the domain level, the complete requirements template pattern cannot be applied, since a ER diagram is not related to any kind of textual template. For the same DP, the filter requirements text pattern cannot be applied, since there is no detailed natural language text available in a DDM.

Interaction Level: In the DP *Interaction-Data*, class models are used instead of ER diagrams, but for this DP the same restrictions apply as for the DP *Domain Data*. As the DP *UI-structure* is documented in workspace models which contain no detailed natural language text, the filter requirements text pattern cannot be applied.

System level: The application of DO usage patterns to DPs at the system level differs largely from its application at the other specification levels. The primary notation for the DPs *navigation/supporting functions*, *UI-data* and *screen structure* is a prototype. There is no DO usage pattern that explicitly supports UI-prototyping. When DOs are used in the upper levels (i.e. goal & task level, domain level, and interaction level), the requirements shape the design of the system level DPs. The DPs *internal actions*, *architecture* and *internal data* must not necessarily follow application domain-specific knowledge. They are rather influenced by the system domain. However, for all three DPs, general knowledge from the DO is usable. In many cases DOs provide information about similar problems and their solutions as well as about NFRs in general. Such knowledge is in particular relevant for the architecture, which is largely influenced by NFRs. Requirements engineers can access such knowledge by the use DO as knowledge base pattern.

Table 10.2: TORE DO usage framework: principally possible use of DO usage patterns in each TORE DP (Spec. l. – specification level; p – pattern.)

Spec. l.	**decision point**	consolidate req. p	review DO p	glossary terms p	knowledge base p	complete template p	reasoning p	filter req. text p
Goal & Task level	Supported Stakeholders	✗	✗	✗	✗	✗	✗	-
	Stakeholder Goals	✗	✗	✗	✗	-	✗	-
	Stakeholder Task	✗	✗	✗	✗	✗	✗	✗
Domain level	As-Is Activities	✗	✗	✗	✗	✗	✗	✗
	To-Be Activities	✗	✗	✗	✗	✗	✗	✗
	System Responsibilities	-	✗	✗	✗	✗	✗	✗
	Domain Data	✗	✗	✗	✗	-	✗	-
Interaction level	System Functions	✗	✗	✗	✗	✗	✗	✗
	Interaction	✗	✗	✗	✗	✗	✗	✗
	Interaction-Data	✗	✗	✗	✗	-	✗	-
	UI-Structure	✗	✗	✗	✗	-	✗	-
System level	Nav./Supp.Functions	-	-	✗	-	-	-	-
	Dialog	-	-	✗	-	-	-	-
	UI-Data	-	-	✗	-	-	-	-
	Screen Structure	-	-	✗	-	-	-	-
	Internal Actions	✗	-	-	✗	-	-	-
	Architecture	✗	-	-	✗	-	-	-
	Internal Data	✗	-	-	✗	-	-	-

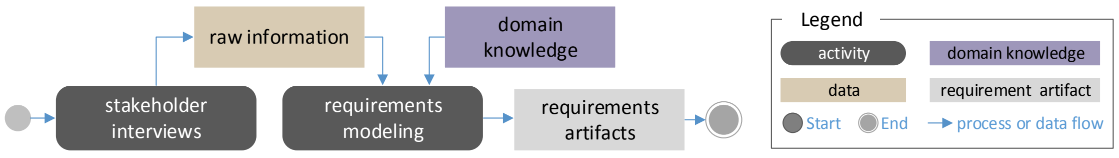


Figure 10.1: Immediate use of domain knowledge during the specification

10.1.2 Embedding the Use of Domain Knowledge in the RE Process

There are two different ways to apply the DO usage patterns – and thus two different ways to apply domain knowledge to requirements – to the overall RE process with TORE. Both Figure 10.1 and Figure 10.2, as described below, are depicted following the notation of UML activity diagrams. All patterns except for the complete requirements template pattern consume as input either InRs or SRSs, as is summarized in Table E.8 in the Appendix on page 250.

Figure 10.1 shows the *immediate use of domain knowledge* during the requirements specification. After gathering information from stakeholders through interviews, apprenticeships, questionnaires and other kind of requirements elicitation techniques, a requirements engineer creates the respective RE artifacts using the domain knowledge immediately. Depending on the requirements engineers' work-flow, there might exist intermediate RE artifacts that are the basis to apply the DO usage patterns. In contrast with this procedure, Figure 10.2 shows the *subsequent use of domain knowledge*. In a first step, the requirements are documented as guided by TORE and its artifact types. In a second step, the resulting RE artifacts are improved by the application of domain knowledge. Depending on the context, either the *immediate use* or the *subsequent use of domain knowledge* is more useful. The definition of TOREOnto in Section 10.2 is based on the *subsequent use of domain knowledge* approach. We use the existing SRS for CIONx to demonstrate how its existing artifacts can be improved after an initial version of the requirements has been specified.

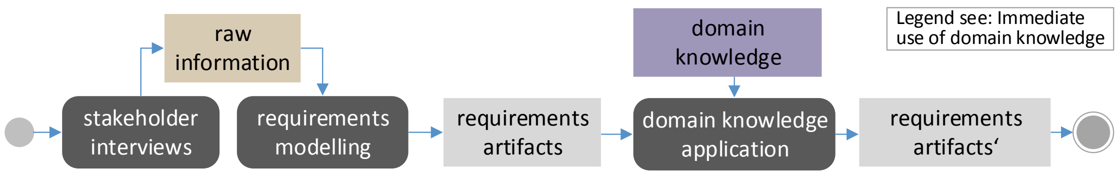


Figure 10.2: Subsequent use of domain knowledge during the specification

10.2 TOREOnto - A Specific Instance of TORE

The DO usage framework presented in Table 10.2 shows the possible use of the DO usage pattern in all of TORE's DPs. In the given case of the SNIK project, we propose TOREOnto as a specific instance of TORE. TOREOnto defines the intended usage of the SNIK ontology on each DP. The selection of the DO usage pattern is motivated by the RQAs (as introduced in Table 2.1 on page 18) that the SRS is to provide. The SRS for CIONx is supposed to be *complete*, *correct*, *consistent*, and *unambiguous* for the selected user tasks. Table 10.3 shows this specific instance of TOREOnto. All those TORE DPs in which a DO usage pattern is used in TOREOnto, as shown in Table 10.3, is called an **extended Decision Point** that links DO usage pattern *and* DP. Principally, TOREOnto allows one to apply multiple patterns in one DP. In such cases it is necessary to define the order in which the DO usage patterns are applied. The reason *why* the combination of DP and DO usage pattern was chosen is next given in Table 10.3 in the row *reason*, and is discussed in the following subsections. According to the overall quality goals of the SRS, the improvement of multiple RQAs is an important factor in all DPs. We decided to focus on the most important RQA in each modified DP, that resulted in the application of one DO usage pattern per DP. The context of the TOREOnto instance that is presented is the specification of the task *IT strategy* which was obtained in Part II of this thesis, which is a part of the CIONx SRS. As a result of this, not all of TORE's DPs are relevant. Although the artifacts were created with DsTORE, the application of the patterns requires no decision-specifics and works for both artifacts created with TORE and artifacts created with DsTORE.

10.2.1 Selection of Patterns

The selection of the DO usage patterns is motivated by the SNIK project, with the goal of creating a complete, correct, consistent and unambiguous SRS for CIONx. TORE's specification levels have different levels of abstraction. The upper levels are coarsely grained, while lower levels are more specific and detailed. This fact influences the selection of patterns. The use domain ontology as knowledge base pattern is the least specific or most coarse pattern, and is thus well suited for the *goal & task level*. Both the use individuals as glossary terms pattern and the consolidate requirements pattern are more specific, and thus are well suited for the *domain level*. The complete requirements template pattern is the most concrete pattern, and is thus suited for the *interaction level*. The abstraction levels are also related to the abstraction level that is provided by the SNIK ontology. Some DPs reach a level of detail which exceeds the range of the SNIK domain ontology.

Table 10.3: The chosen DO usage patterns in several DPs Spec. l. = specification level

Spec. l.	Decision Point	DO Usage Pattern
Goal & Task level	Supported Stakeholders	use domain ontology as knowledge base pattern reason: consider all necessary stakeholders
	Stakeholder Goals	–
	Stakeholder Task	use domain ontology as knowledge base pattern reason: gain complete and correct list of task
Domain level	As-Is Activities	use individuals as glossary terms pattern reason: gain unambiguous and complete task descr.
	To-Be Activities	use individuals as glossary terms pattern reason: gain unambiguous and complete task descr.
	System Responsibilities	–
	Domain Data	consolidate requirements pattern reason: use domain entity types and relations consistently
Interaction level	System Functions	complete requirements template pattern reason: gain consistency with data model
	Interaction	–
	Interaction-Data	–
	UI-Structure	–
System level: – not relevant in this RE phase –		

The following subsections describe for what reasons the DO usage patterns are chosen from the TORE DO usage framework in Table 10.2. They result in the specific TOREOnto instance in Table 10.3.

10.2.1.1 Goal & Task Level

In the DP *Supported Stakeholders*, all necessary stakeholders shall be considered for the intended system scope. Thus, we apply the use domain ontology as knowledge base pattern to validate existing roles and search for additional roles, which will be considered in the requirements elicitation. Closely related roles of the CIO are considered as an initial set of candidates for stakeholders. In the case of CIONx, the *stakeholder goals* are sufficiently described as a result of the domain analysis. Thus, we require no ontology support for this DP. Tasks are in particular relevant to the SRS. Thus we ensure the completeness and correctness of the specified tasks with the domain ontology as knowledge base pattern in the DP *Stakeholders' Tasks*.

10.2.1.2 Domain Level

In SRSs the terms and their relations familiar to domain experts shall be used. As the *domain level* refines the *goal & task level*, it is important to ensure that a precise terminology is used on the lower levels. Also, the *domain level* influences the refinement and use of terminology used in the lower levels. The DP *Stakeholders' Tasks* and their subtasks in the DPs *As-is Activities* and *To-be Activities* are both important for a detailed understanding of the user tasks and the supporting features of the system. The activities described in these two DPs significantly influence the user acceptance of the to-be system, and we need to ensure their documentation's unambiguity and completeness. Therefore, we apply the use individuals as glossary terms pattern to the DPs *As-is Activities* and To-be Activities, and thereby ensure a continuous terminology in the requirements. The task description of the DP *As-is Activities* would have been a candidate for the complete requirements template pattern. However, we did not choose this pattern here, since the major contribution of the task description are the free text fields *as-is solution* and *example solution*. Instead, it was important for us to ensure the correct use of terminology within these free text fields. The DP *Domain Data* enables the definition of the ER diagram that acts as a blueprint for a concrete system data model. The entities of the DDM shall be consistent with the domain entities contained in the DO. Since the domain data is documented by a semi-formal data model, it is well suited to be compared to the entity types that are defined in the DO with the consolidate requirements pattern. For this reason we apply the consolidate requirements pattern.

10.2.1.3 Interaction Level

System functions consume and create data that must be consistent with the data model. A common notation for system functions are *system function descriptions*. Such templates are well suited to be used with the complete requirements template pattern. The fields of a *system function description* are assigned to concepts in the DO, in which the instances of concepts provide the details that are to be entered into the template fields. This enables us to select instances of the DO and enter them into the template. In particular the entity types of the SNIK DO define the input and output data of a system function. Thus, we use the complete requirements template pattern for the DP *System Functions*.

For CIONx as a PDSS, interaction plays a subordinate role. The main goal of the CIONx specification is to provide a system UI prototype in order to illustrate the decision-specific presentation of data. Thus, there is an emphasis on the relation of stakeholder, their tasks, and the data model. The specification of these models shapes the UI prototype, where no

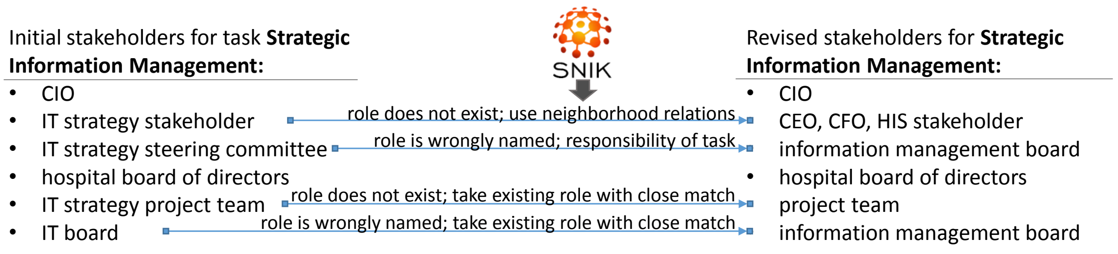

Initial stakeholders for task **Strategic Information Management:**
CIO
IT strategy stakeholder
IT strategy steering committee
hospital board of directors
IT strategy project team
IT board
SNIK
role does not exist; use neighborhood relations
role is wrongly named; responsibility of task
role does not exist; take existing role with close match
role is wrongly named; take existing role with close match
Revised stakeholders for **Strategic Information Management:**
CIO
CEO, CFO, HIS stakeholder
information management board
hospital board of directors
project team
information management board

Figure 10.3: Application of the use the DO as a knowledge base pattern to stakeholders

additional application of domain knowledge is necessary. Finally, due to the goal of specifying the UI prototype, TORE's system level is not relevant in this RE phase.

10.2.2 Application of Patterns in each Decision Point

The following subsections describe the application of the selected DO usage patterns in each extended DP. This is an application by way of example of the patterns used to demonstrate the principal process of the domain knowledge application. The entire application of TOREOnto for revising the RE artifacts and its results is presented in the evaluation in Chapter 11 on page 151.

10.2.2.1 DP Supported Stakeholders: Use Domain Ontology as Knowledge Base Pattern

Figure 9.14 on page 124 describes the use the DO as knowledge base pattern. Our goal here is to complete the list of stakeholders. An example of the application of this pattern is given in Figure 10.3. The left side contains the initially defined stakeholders. Some of them are not contained in the DO. As these stakeholder roles are not defined in the DO, they might be incorrect or ambiguous in the original requirements artifact. Furthermore, some stakeholders are missing in the original artifact who are however relevant, for example the chief executive officer (CEO) and the chief financial officer (CFO). In order to complete the list of stakeholders, we execute three activities.

First, the identification of the responsibilities for tasks. In order to achieve this, we raise the knowledge question: "Who is responsible for the tasks in focus?"[1]. The SNIK DO contains such knowledge about the responsibility of tasks as IM functions. Given a list of tasks, we query the DO for each task's responsibilities and thereby obtain the required

[1]This question assumes that some or even all of the tasks in focus are already known.

information about the roles of responsibility. In this example we query the DO for *strategic information management* and find the *Information Management Staff* to be responsible for it, whereas the CIO is a member of it. From the interviews, however, we know that the *CIO* is responsible for that task. Thus, we expand the CIO to include collaborating roles. Thereby we identify the CEO and CFO. Another collaborating role is the *Information management board*, which is a subclass of the role *HIS stakeholder*. Second, the identification of further relevant stakeholders by their relations. The neighborhood relations of the stakeholders who were identified are now assessed in order to find new stakeholders who appear influential. In this example, the role *IT steering committee* does not exist in the ontology, but the *Information management board* is related to the *CIO*. The description of the *Information management board* indicates the influential role which covers the *strategic information management*, and justifies its use. Third, the identification of similar roles by their names. In this example, the role *IT strategy project team* does not exist in the ontology, but the role *Project Team* does. As the creation of the *strategic IM plan* can be seen as a project, this role is a good choice. The role *IT board* also does not exist in the SNIK ontology. As IT is part of IM, the ontology is queried for the role *Information management board*, which does exist. The result is a completed and corrected list of stakeholders on the right side of Figure 10.3.

10.2.2.2 DP Stakeholders' Tasks: Use Domain Ontology as Knowledge Base Pattern

For the DP *Stakeholder' Tasks*, the same pattern is used as in the last subsection. The correct naming and description of user tasks is essential for further RE activities. According to the Table B.1 in Appendix B on page 206, stakeholder tasks in TORE are documented by task descriptions, task & support notation, or business process diagrams. When the use the DO as knowledge base pattern is applied at the very beginning of the RE activities, the results will be unspecific. To query any task in the DO will result in a high amount of irrelevant results. Thus, a first approximate idea of the tasks in focus is necessary. The knowledge of neighboring concepts reveals correct names and missing tasks and subtasks that can be added to the requirements. Thus, the domain knowledge helps to improve the completeness and correctness of InR and SRS.

Proceeding outwardly from the terms of the task name, we query the DO for existing tasks. For example, the SNIK DO does not provide a function with the exact name IT strategy, but there are functions related to "strategy" and "strategic", such as strategic information management, or service strategy. From the description of *strategic information management*, we know that this task matches the meaning of *IT strategy*. Thus, we replace

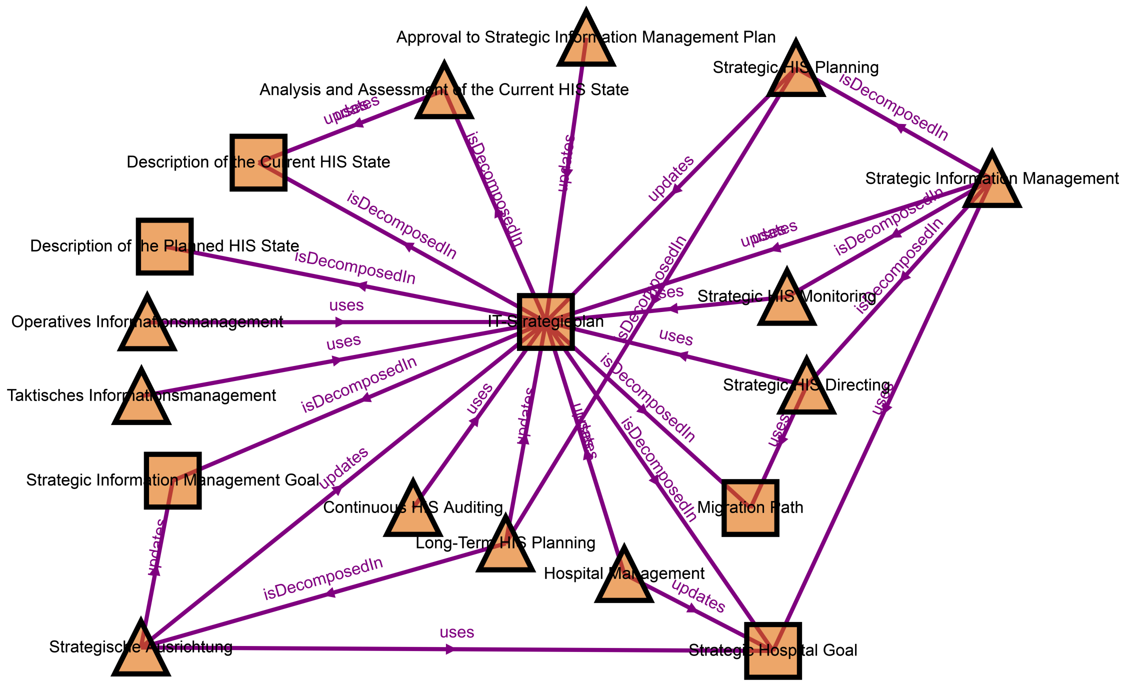

Approval to Strategic Information Management Plan
Strategic HIS Planning
Analysis and Assessment of the Current HIS State
Description of the Current HIS State
Strategic Information Management
Description of the Planned HIS State
Strategic HIS Monitoring
Operatives Informationsmanagement
IT Strategieplan
Taktisches Informationsmanagement
Strategic HIS Directing
Strategic Information Management Goal
Continuous HIS Auditing
Migration Path
Long-Term HIS Planning
Hospital Management
Strategische Ausrichtung
Strategic Hospital Goal
updates
uses
isDecomposedIn

Figure 10.4: The neighboring concepts of the strategic IM plan

the task name *IT strategy* with the name *strategic information management*. By assessing the neighbors of selected tasks, we identify additional tasks and entity types that are relevant to the SRS. From the neighbors of the strategic information management plan, which is presented in Figure 10.4, it is known that the task strategic information management *uses and updates* the *strategic IM plan*. In the initial requirement, the term *IT strategy* is used for both the task *strategic IM* and the document *strategic IM plan*. Besides its use for DP *Stakeholders' Tasks*, this view is useful for other DPs. The neighbors provides concepts related to strategic IM that should be considered in the DP *As-is and To-be Activities* and the DP *Domain Data*. The initial task *IT strategy*, as described in Table D.3 on page 232, consists of several subtasks that can be categorized by the *creation of strategic information management plan* and *use of strategic information management plan*. These tasks are exact subclasses of the task life cycle management of strategic information management plan, being neighbors of the *life cycle strategic IM plan*. Figure 10.5 shows the details of this life cycle. This information from the DO leads to the addition of two subtasks, namely *deployment* and *updating of the strategic IM plan*. Figure 10.6[2] shows the stakeholders' tasks before and after the application of the use the DO as knowledge base.

[2]Please note that both Figure 10.5 and Figure 10.6 are excerpts of the SNIK ontology visualization. They also contain German terms, which is a known issue.

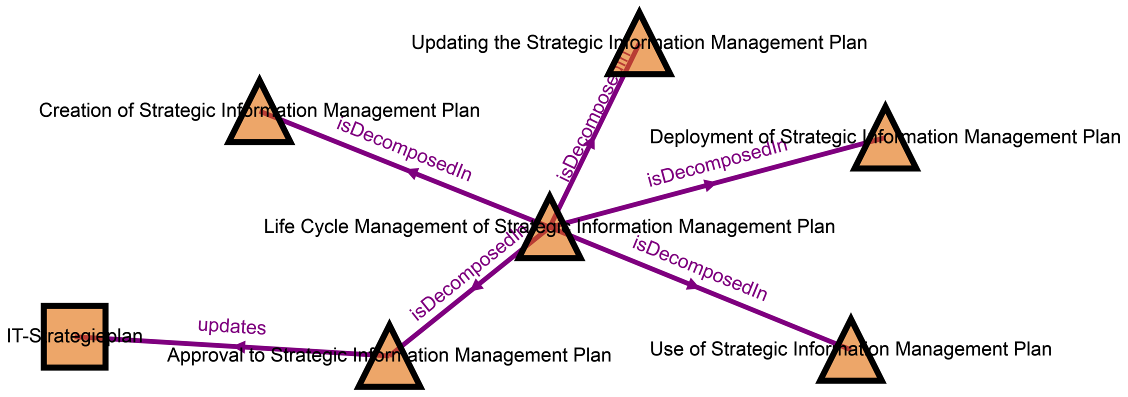

Updating the Strategic Information Management Plan
Creation of Strategic Information Management Plan
Deployment of Strategic Information Management Plan
Life Cycle Management of Strategic Information Management Plan
IT-Strategieplan
Approval to Strategic Information Management Plan
Use of Strategic Information Management Plan
isDecomposedIn
isDecomposedIn
isDecomposedIn
isDecomposedIn
isDecomposedIn
updates

Figure 10.5: The life cycle management of the strategic IM plan

1	creation of IT strategy
2	use of IT strategy

(a) Initial stakeholders' tasks.

1	creation of strategic information management plan
2	use of strategic information management plan
3	updating of the strategic information management plan
4	deployment of strategic information management plan

(b) Revised stakeholders' tasks.

Figure 10.6: Excerpt of the stakeholders' tasks both before and after the DO's use as a knowledge base

10.2.2.3 DP As-Is Activities and To-Be Activities: Use Individuals as Glossary Terms Pattern

Figure 9.12 on page 122 shows the use individuals as glossary terms pattern. The main purpose of this pattern in combination with the TORE DPs *As-is Activities* and *To-be Activities* is to use the DO as a glossary. The terms of the detailed task description are compared and possibly linked to the terms that are defined in the DO. For each significant term in the documentation of the DPs *As-is* and *To-be Activities*, suitable terms from the ontology need to be identified. Synonyms – which are different terms with the same meaning – are substituted with the terms of the DO. All terms of the task description that can be identified to match the terms in the DO are used and linked in the task description. Nevertheless, the term-candidates must be carefully considered by the requirements engineer, and finally, the linked terms in the requirements must be verified by the stakeholder.

As an example of this, we use the description of the *As-is Activities* and *To-be Activities* in Table D.4 on page 233 for the subtask IT strategy 1a: approval of the stakeholders strategic information management. By applying the use the DO as knowledge base pattern in the DP *Stakeholders' Tasks*, we identified the correct task name as *strategic IM* and the document as *strategic information management plan*. Now we use these terms with links to the SNIK ontology in the requirement. Figure 10.7 shows this subtask description before and after the use individuals as glossary terms pattern has been applied. The blue terms are the linked DO terms.

The IT strategy steering committee has to be formed prior to the creation of a new IT strategy. Participating stakeholders must be assigned to (but are not limited to) the divisions of *patient care*, *research and teaching*, and *information management*. Stakeholders then state their requirements and expectations regarding the IT strategy, representing the opinion of their associated divisions. The proposal of the IT strategy steering committee participants is then presented to the hospital's board of directors for approval.

(a) Initial subtask 1a description

The information management board has to be formed prior to the creation of a new strategic information management plan. Participating stakeholders must be assigned to (but are not limited to) the divisions of PatientCare, EducationResearch, InformationManagement. HIS stakeholder state their requirements and expectations regarding the strategic information management plan, representing the opinion of their associated organizational units of the hospital. The proposal of the information management board participants is presented to the hospital's board of directors for approval.

(b) Revised subtask 1a description

Figure 10.7: A subtask description with linked glossary DO terms

10.2.2.4 DP Domain Data: Consolidate Requirements Pattern

The consolidate requirements pattern is depicted in Figure 9.10 on page 121. Its main purpose is to consolidate a model with a given description in a DO. The TORE DP *Domain Data* is documented as an ER diagram or an UML class diagram. Both models have in common the fact that they display entities with their associations to each other. For the application of this pattern, both entities and their associations are compared between the data model and the knowledge contained in the DO. The comparison shows possible entities to be renamed, and missing or wrong associations between entities. However, in some cases the results of the comparison are misleading. As the DO might possess another level of details, entities or their associations cannot be found in the ontology, although they are correct. Thus, the requirements engineer in cooperation with the stakeholder needs to decide the entities on a case by case base.

Table 10.4 shows as an example some entities of the domain data model for the task IT strategy, as is presented in Figure D.25 on page 240. All entities that are shown are compared with the SNIK DO. The entity *person* does not exist in SNIK DO, but the meta-class *Role*

Table 10.4: Exemplary entities and their relation to the SNIK DO

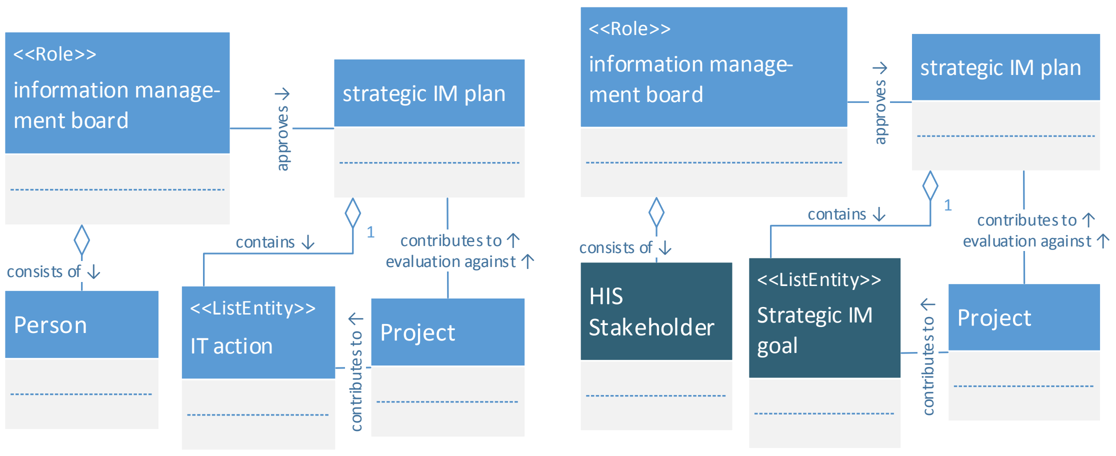
Entity in diagram	**Entity in domain ontology**	match
Person	Role	close
Person	HISStakeholder	more precise
Project	Project	exact
funding application	–	none

Table 10.5: Exemplary associations between entities in DDM (1st line in cell) and their occurrence in the SNIK DO (2nd line with gray background in cell).

From Entity	**Association**	**To Entity**
information management board	consists of	person
information management board	subclass of	HIS Stakeholder
strategic IM plan	contains	IT action
strategic IM plan	isDecomposed	Strategic IM goal
Project	contributes to	strategic IM plan
Strategic HIS directing	uses	strategic IM plan
Strategic HIS directing	updates	project

constitutes a close match. In the context of the task strategic IM, the role *HIS Stakeholder*, which is a subclass of *role*, is an even more precise hit. The entity *project* is exactly the same as it is in the SNIK DO. The entity *funding application* does not exist in the SNIK DO. Even when using different search methods (for example, string and substring search, as well as the neighboring relationships of closely related concepts, such as controlling or finances), there is no corresponding entity in the SNIK DO. Thus, we need to decide whether to keep this entity or to reassess the occurrences of this entity in the stakeholder communication. The same strategy applies to the relations between entities that are shown in Table 10.4. When applying the consolidate requirements pattern, there is always the risk that the DO is incomplete or that it misses entities and relations. In particular, the SNIK does not provide the same level of detail for associations as it does for entities. Thus, a relation that does not exist explicitly in the ontology should not lead to a dropped association in the DDM.

Table 10.5 shows an example of the comparison. The first entry in each cell is the triple *from entity*, *association*, *to entity* as an excerpt of the data model. The second cell entry with gray background shows the equivalent triple of the DO. In the first cell, the data model entity *person* is more specifically described in the ontology with the entity *HIS Stakeholder*. The second cell shows that the data model entity *IT action* is described in the DO as the entity *Strategic IM goal*. The example in the third cell is special. The DO has no direct relation between *project* and the *strategic IM plan*, although a relation must exist. In fact,



(a) Excerpt of the initial DDM (b) Excerpt of the revised DDM

Figure 10.8: Excerpt of the DDM before and after the DO's use

this relation is modeled in the DO via the task *strategic HIS directing* that uses the *strategic IM plan* and updates the *project*. As a consequence of this, the requirements engineer would modify the DDM with the entity found in the DO. Figure 10.8 shows on the left side an excerpt of the initial DDM for the task *IT strategy*, and on the right side the revised DDM. Changed entities are highlighted in dark blue.

10.2.2.5 DP System Functions: Complete Requirements Template Pattern

The complete requirements template pattern is depicted in Figure 9.15 on page 124. The patterns' purpose is to use DO concepts or their individuals to fill them into the fields of template. The first step in the application of the pattern is to create a mapping between template fields and concepts of the ontology. The second step of the application of the pattern is to fill the template with concepts from the DO.

As an example of this, the Figure 10.9 shows a manually filled template of a system function describing a data import in the context of a data warehouse system[3]. From the system function description, only the first four fields are suited to a mapping to the SNIK ontology. All other fields of the system function description contain a level of detail which is not provided by the SNIK ontology. The mapping is shown in the column *ontology mapping* Figure 10.10, that contains the mapping of concepts or types of the SNIK ontology to the respective template field. The field *name* is mapped onto *features* contained in SNIK ontology. The field *realized in* was added to the system function description to refer to a

[3]This system function is not specified explicitly in the requirements specification for the task IT Strategy.

Figure 10.9: Example of a manually specified system function description

Field	Requirements Content
Name	data import
Realized in	data warehouse
Input data	data from databases and documents
Output data	structured data warehouse data
Description	data from various structured databases and the text documents from the workspace server are imported to the data warehouse
Exceptions	none
Rules	only changed documents are to be imported
Quality requirements	the import process must be fail safe. availability of data is less important
Preconditions	the main production database is available
Post conditions	the data is loaded into the data warehouse

Figure 10.10: Example of a system function description and its mapping onto the SNIK DO

Field	Requirements Content	Ontology Mapping
Name	integrate data from different application components	feature
Realized in	Data Warehouse System	application component
Input data	Data	entity type
Output data	Structured data	entity type
Fields *Description* – *Post conditions* are unchanged		

specific component of a large system. The field is mapped onto an *application component* of the SNIK DO. Table 10.6 shows the definition of a *data warehouse* in the SNIK DO. The fields *input data* and *output data* are mapped onto *entity types* in the SNIK DO. When applying the complete requirements template pattern, the requirements engineer fills concepts from the SNIK ontology into the mapped fields of the system function description. The result is shown in Figure 10.10, which contains in the first four rows the linked terms from the SNIK ontology. The rows of Figure 10.10 replace the rows with the same name in Figure 10.9.

10.3 Discussion

The TOREOnto instance that is presented was adapted to the needs of the SNIK project in order to create a *complete*, *correct*, *consistent*, and *unambiguous* InR for the CIONx tool. The chosen TOREOnto configuration can be used in the same way for different specifications and various tasks, which are not limited to the task IT strategy that we have used as an example. The chosen TOREOnto configuration is the adaptation to a specific application domain in a defined context. This adaptation as described in this thesis provides a blueprint for method designers or requirements engineers who want to configure TORE to operate with other application domains and contexts. On TORE's highest level of abstraction, the

Table 10.6: Part of the definition showing the features of the application component *data warehouse system* in the SNIK ontology

Source	http://www.snik.eu/ontology/bb/DataWarehouseSystem
Name	Data Warehouse System http://www.snik.eu/ontology/bb/DataWarehouseSystem
skos:definition	Application component that contains data which have been extracted from other application components, in order to support either hospital management or clinical research. @en
rdfs:label	Data Warehouse System @en
bb:Chapter	Thesaurus
bb:ID	486
bb:page	129, 301
rdfs:subClassOf	bb:ComputerBasedApplicationComponent
meta:supports	Execution of Clinical Trials, Hospital Management, Research and Education
meta:typicalFeature	analyze data, integrate data from different application components, provide means for data mining, provide means for recruiting of patients, structure data

goal & task level, the most general pattern use DO as knowledge base is used. This pattern describes the case in which a requirements engineer browses and queries the ontology for any kind of problem. At this level of abstraction, it is beneficial to explore the domain knowledge in an unspecific way. In this way, the requirements engineer makes him/herself familiar with the application domain, some of their core concepts and their explicit relations.

10.4 Chapter Summary

This chapter presents the domain ontology usage framework and TOREOnto as a specific instance. As a basis for TOREOnto, the improved RQAs of each DO usage pattern are presented in a summary table. The possible use of each DO usage pattern within all of TORE's DPs is made explicit in TORE's domain ontology usage framework. Cases in which the application of some DO usage pattern within a DP is either not possible or not meaningful are then discussed, and the reason for the inappropriate use is indicated. The specific instance of TOREOnto that is presented is configured for the CIONx-specification in the context of the SNIK project as the given case. In TOREOnto, we have chosen the application of four different DO usage patterns in six DPs. In particular, the DPs *Supported Stakeholders* and *stakeholder task* are completed by the use domain ontology as knowledge base pattern. The DPs *As-is Activities* and *To-be Activities* are improved with regard to unambiguity with the use individuals as glossary terms pattern. The DP *Domain Data* is improved with regard to its consistency with the consolidate requirements pattern. The DP *Interaction* is improved with regard to its correctness with the complete requirements template pattern.

11 Treatment Validation - Evaluation of TOREOnto

In this chapter, the TOREOnto method, as it was presented in Table 10.3, is evaluated to validate RQ.2.2 in a case study. The application of the selected DO usage patterns is demonstrated by means of an example with RE artifacts of the CIO's task IT Strategy, as will be presented in Section D.3 and the SNIK domain ontology. Section 11.1 presents the design of the case study together with the research questions, a detailed case selection and the data collection procedure. Section 11.2 presents the results of the case study and the effects of cascading changes. In Section 11.3, the results of the case study are discussed. The threats to validity of this case study are discussed in Section 11.4. General experiences with the method adaptation are described in Section 11.5. Finally, the chapter is summarized in Section 11.6.

11.1 Study Design

We follow the guidelines for case studies in SE that are set down by Runeson and Höst [RH09]. TOREOnto defines when, where and how domain knowledge is used during RE for the specific problem context. The *objective* of this case study is to validate each of TOREOnto's extended DPs with regard to their **utility** and **usability** in a real-world context. In particular, we study the effects of applying the DO usage patterns in the extended DPs. Therefore, we raise and answer RQ.2.2: *How well does TOREOnto support the improvement of requirements?* Our case study has the following limitations: The stakeholder of hospital X is not available anymore. As a result, the requirements engineer had to decide on his own changes in the requirements without any validation or feedback. The evaluation of the results were carried out by two domain experts instead of stakeholder X. Therefore this case study is a retrospective evaluation. The evaluation is limited to the extended DP. Changes of one artifact lead to a cascading effect of changes in other artifacts. We describe the cascading effect for each DP, but the revisions of the respective artifacts are not executed and are not included in this study.

Table 11.1: Overview of the treatment validation activities

ID	Dur.	Activity	Description
1	4:25	Engineering	Improvement of the DPs *Stakeholder Tasks*, *Supported Stakeholder*, *As-Is Activities*, *To-Be Activities*, *Domain Data*, *System Functions*
2	—	Evaluation	Questionnaire of requirements engineer
3	—	Evaluation	Questionnaire of domain expert 1
4	—	Evaluation	Questionnaire of domain expert 2

The *domain ontology usage framework* that was presented in Section 10.1 allows method designers and requirements engineers to combine multiple patterns in each DP. However, the TOREOnto method, as it was described in Section 10.2, defines the application of a maximum of one DO usage pattern in the DPs. Thus, the effect of DO usage pattern combinations is not studied.

11.1.1 Research Question

Wieringa [Wie14] lays out two types of knowledge questions for validation. First, *usability questions* ask whether the measured effects satisfy usability with regard to understandability, ease of use, and ease of learning. Second, *utility questions* specify whether the artifact is able to achieve stakeholder goals. In context of the selected case, we did not apply TOREOnto directly during RE elicitation and documentation in a real-world context. Thus, we adapt the questions to usability and utility for extended DPs in TORE in the following way. We measure the *usability* in terms of the ease of use regarding the application of TOREOnto in general. A first aspect of usability is the time taken to apply the domain knowledge and to modify the artifacts in extended DPs. We query this in RQ.2.2.1. As requirements engineers apply TOREOnto, we asked them to rate their perceived ease of use of the corresponding method step in order to answer RQ.2.2.2. Next, we measured the *utility* by evaluating the revised requirements artifacts, as we have no system prototype. A higher quality of each artifact is accompanied by a higher quality of the SRS. The TOREOnto method has rated as having a high utility if it improves the quality of RE artifacts. Thus, we determine the effects of the TOREOnto application regarding both the original and the modified artifacts in RQ.2.2.3. After this, we investigated whether the use of TOREOnto increases the completeness, correctness, consistency, and unambiguity of the RE artifacts that are generated. In summary, we refine RQ.2.2 as follows:

Table 11.2: Metrics of data collection for the TOREOnto treatment validation case study. Column *question* refers to the questionnaire in Section A.6 on page 204.

Goal	Metric	Description of Metrics	Question
RQ.2.2.1 usability	m1	Time taken for the application of extended DPs to improve existing artifacts with domain knowledge	–
	m2	Requirements engineers' acceptance of the effort to revise the requirements.	QMeth.1
RQ.2.2.2	m3	Requirements engineers' feedback to the method	QMeth.2 – QMeth.5
utility	m4	Requirements engineers' rating of the artifact quality	see Table 11.3
RQ.2.2.3	m5	Requirements engineers' rating of the improvement effect on the artifact	QArtif.12
	m6	Domain experts' rating of the artifact quality	see Table 11.3
	m7	Domain experts' rating of the improvement effect on the artifact	QArtif.12

RQ.2.2.1: Is the overhead acceptable through the additional work?

RQ.2.2.2: What is the perceived ease of use for executing the improvement of requirements?

RQ.2.2.3: What is the quality improvement of the requirements in the corresponding method step?

Three persons were involved in this case study. First, a requirements engineer who is the author of this thesis. Second, two domain experts who give lectures in the application domain IM in hospitals and who were largely involved in the creation of the SNIK ontology. The requirements engineer followed TOREOnto as it was presented in Table 10.3, and applied the DO usage pattern as was described in subsection 10.2.2 in each DP to the respective artifact, in order to revise and improve these artifacts. The original artifacts (as are presented in Section D.3) and the revised artifacts were both assessed by the requirements engineer and two domain experts in activities 2 – 4. The assessment is based on a questionnaire for the method evaluation, which is presented in subsection A.6.1, and a questionnaire for each of the revised artifacts with TOREOnto, which is presented in subsection A.6.2. Based on the TOREOnto definition, the following DPs were in focus: *Stakeholder Tasks*, *Supported Stakeholder*, *As-Is Activities*, *To-Be Activities*, *System Functions*, and *Domain Data*. Since there were no system functions specified before with a template, we created one system function when applying TOREOnto.

We performed the data analysis after the requirements engineer had completed the questionnaire, and after both domain experts had completed their questionnaires. Table 11.1 shows the activities of the case study along with the DPs whose requirements have been improved. In order to define the metrics and thereby to answer RQ.2.2.1 – RQ.2.2.3, we used a GQM approach [BCR94]. Table 11.2 shows these metrics. We study TOREOnto's *usability* by taking into account the time taken for the revision of the artifacts (m1), the requirements engineer's acceptance of the effort (m2) and the feedback to the method (m3). Both metrics, i.e. the time taken and the acceptance of the effort, are important to determine if the effort for the revision is worth it for the results. We use the requirements engineer's feedback to the method to reveal potential difficulties, the easiness of the pattern applications and possible obstacles in the access of domain knowledge. The *utility* of TOREOnto is measured by means of the rating of the artifacts quality by the requirements engineer (m4) and the domain experts (m6), and the explicit rating of the improvement effect by the requirements engineer (m5) and the domain experts (m7). We argue that the method has a high utility if the artifacts quality is improved by the revision with the DO usage patterns. Additionally, we gather the improvement effect rating in order to estimate the participant's feeling of overall improvement of the artifacts. The artifacts quality relates to the ISO/IEC 29148 [ISO11] definition of the RQA, as was presented in Table 2.1 on page 18. The selection of the RQA that was emphasized in this evaluation is restricted by the DO usage patterns. They are able to improve the RQAs complete, correct, unambiguous, consistent, and traceable. Traceability is not a focus of this thesis, and thus the first four RQAs are selected. The metrics m4 and m6 are refined into these RQAs to capture the artifacts' quality, as shown in Table 11.3. The DO usage pattern improves only a subset of the RQAs, as was explained in Table 10.1 on page 136. Thus, the evaluation focuses on the improvement effect regarding the selected RQA, as is indicated by ✗in columns 2 – 6 of Table 11.3. However, in order to capture possible side-effects regarding other RQAs, we queried all RQAs for each artifact.

In Table 11.3, we further refine m4 and m6 on the basis of the definition of the RQAs, namely *InR.consistency*, *InR.completeness*, *InR.unambiguity*, and *InR.correctness* of ISO/IEC 29148 [ISO11], as was presented in Table 2.1 on page 18. As the InR of each DP are improved separately and we excluded cascading changes, we cannot obtain an improved SRS. For reasons of clarity, we provide a short summary of the RQAs of Table 2.1 on page 18 below. The given criteria of the RQAs explain why these metrics were used in Table 11.2, specifically for the following reasons: (1) *consistency*: The requirement is free of conflicts with other requirements. Thus, we query the *conflicts* (m(4|6).1.1 – m(4|6).1.2) of the requirements. (2) *completeness*: The stated requirement is measurable and sufficiently describes the capability and characteristics to meet the stakeholder's need. Thus, we query the *description sufficiency* of the artifact (m(4|6).2.1) and missing details (m(4|6).2.2) in the

Table 11.3: Detailed metrics for the requirements engineers' (m4) and domain experts' (m6) rating. ✗indicates the goal to be relevant to the applied pattern in the DP.

Goal	DP Supp.Stakeh.	DP Stakeh.Tasks	DP As-is & To-be Act.	DP Dom.Data	DP Sys.Func.	Metric	Description of Metrics	Question
consistency	✗	✗	–	✗	✗	m(4|6).1.1	details of conflicts	QArtif.1
						m(4|6).1.2	severity of conflicts	QArtif.2
completeness	✗	✗	–	–	–	m(4|6).2.1	details of sufficient description	QArtif.3
						m(4|6).2.2	severity of missing details	QArtif.4
unambiguity	✗	✗	✗	–	✗	m(4|6).3.1	details of different interpretations	QArtif.5
						m(4|6).3.2	severity of different interpretations	QArtif.6
						m(4|6).3.3	degree of understandability	QArtif.7
						m(4|6).3.4	details of synonyms found	QArtif.8
						m(4|6).3.5	severity of synonyms used	QArtif.9
correctness	✗	✗	–	–	✗	m(4|6).4.1	details of correctness	QArtif.10
						m(4|6).4.2	severity of correctness issues	QArtif.11

artifact. (3) *unambiguity*: There is only one way to interpret the requirement and it is simple and easy to understand. The ISO/IEC 29148 standard [ISO11] does not mention synonyms explicitly for InR, except for the completeness of SRS. We argue that synonyms used in an InR lead to additional interpretations. Thus, we query possible *different interpretations* (m(4|6).3.1 – m(4|6).3.2), the understandability of the requirement (m(4|6).3.3), and the details of possible synonyms (m(4|6).3.4 – m(4|6).3.5). (4) *correctness*: the stakeholder(s) have confirmed that the requirement is expressed correctly. Thus, we query the correctness (m(4|6).4.1 – m(4|6).4.2) of the requirements.

11.1.2 Case Selection

The case comprises the existing RE artifacts for CIONx of the SNIK project as were introduced in Section 2.5 on page 31. We selected this case for two reasons: first, on the basis of availability; and second, since RE artifacts were created with TORE in the application domain *IM in hospitals*, for which domain knowledge is available in the SNIK DO. Existing RE artifacts of the task *IT strategy* constitute the basis for this case study. Although the RE artifacts were created with DsTORE, the application of the DO usage patterns requires no decision specifics and works for both artifacts created with TORE and those created with DsTORE.

11.1.3 Data Collection Procedure

The revision of the RE artifacts was completed by the requirements engineer, who is the author of this thesis, and the creator of the existing RE artifacts laid out in Part II. As the domain knowledge source, the SNIK domain ontology is used as of December 2017. The DO usage patterns as described in Section 9.3 on page 118 are applied to the RE artifacts in the respective DP, as defined by TOREOnto in Table 10.3 on page 140. TOREOnto and thus the revision of all artifacts are based on the *subsequent use of domain knowledge*, as has been described in Figure 10.2 on page 138. As no *system function* artifact was available, we additionally evaluated the *immediate use of domain knowledge*, as was described in Figure 10.1 on page 138.

11.2 Study Results

The presentation of the results in this section is structured in accordance with the three RQs. The revised artifacts are shown with highlighted changes where possible, and closely to the respective DP in this section. The application of the DO usage pattern to the artifact is explained in brief at the beginning of each DP-related paragraph. The cascading effects of changes that are identified are presented at the end of this section.

11.2.1 Results for RQ.2.2.1

Table 11.4 shows the time taken (m1) to revise and improve the selected artifacts for this evaluation of TOREOnto. The initial effort to create the artifacts for the task strategic IM is seven hours, as is given in Table 7.2 on page 79. The effort for the revision took approximately four and a half hours. The requirements engineer rated the effort required to revise the requirements with domain knowledge as acceptable. In summary, we answer RQ.2.2.1 by stating that the overhead through the additional work to incorporate domain knowledge is acceptable from the viewpoint of the requirements engineer.

Table 11.4: Time taken (m1) for artifact improvement with TOREOnto in activity 1 of Table 11.1. (Duration in [h:mm])

Decision Point	**Artifact**	**Duration**
Supported Stakeholders	List of stakeholders	0:10
Stakeholders' Tasks	Task list	0:45
As-is and To-be Activities	Subtask description according to task & support	1:10
Domain Data	Class diagram	1:05
System function	System function description	1:15
Total		4:25

11.2.2 Results for RQ.2.2.2

The requirements engineer gave the following feedback to the method and the associated access to domain knowledge (m3): (1) The application of the extended DPs and DO usage pattern was straightforward. (2) The decision if an identified domain ontology concept is correct or not was sometimes difficult for the application of the use domain ontology as knowledge base and the use individuals as glossary terms patterns. There are two reasons for this. First, not all of the concepts have a sufficient description. Second, in some cases the requirements engineer could not decide whether the use of the identified domain ontology concept still describes the stakeholders' intention correctly. (3) The mixture of German and English concepts in SNIK DO is difficult in some cases (which is a known problem). (4) The inclusion of new domain knowledge in an artifact might affect the consistency of the requirement. Some changes have invalidated some aspects of the requirement. For example, the introduction of the deployment of strategic IM plan stands in conflict with the use of strategic IM plan. (5) The application of domain knowledge to the artifact was on the whole easy to very easy. In particular, the requirements artifacts could be easily changed to contain the newly identified or changed terms, tasks, or entities. (6) The search for domain knowledge was sometimes difficult. There were often no matches for search terms. It was also unclear if the concept does not exist in the knowledge base, or if there is a wording issue. The access to application component features for system functions was not possible in the web interface, but possible in the raw RDF/OWL data. (7) The overall access to domain knowledge was easy for the following reasons: First, the visualization of the SNIK ontology provides a very good way to browse through the domain knowledge. Second, details about

1. CIO
2. IT strategy stakeholder
3. IT strategy steering committee
4. Hospital's board of directors
5. IT strategy project team
6. IT board

(a) Initial supported stakeholders

1. CIO
2. CFO
3. CEO
4. Project management board
5. Information management board
6. Hospital board of directors
7. Project team

(b) Revised supported stakeholders

Figure 11.1: Supported stakeholders before and after revision

domain concepts was easy to access by the visualization's LodView feature[1]. (8) Lastly, in many cases, the list of linked domain terms was extremely helpful as a customized vocabulary. In summary, we answer RQ.2.2.2 by stating that the method is on the whole easy to use in order to execute the improvement of requirements.

11.2.3 Results for RQ.2.2.3

The ratings of the artifact quality by the requirements engineer (m4) and the domain experts (m6) are summarized in Table E.9 in the Appendix on page 251. Those ratings which are relevant to each DP according to the postulated RQAs are presented as separate tables in the following section. The ratings for each DP are discussed in the following section in greater detail. The ratings of domain expert 1 and domain expert 2 are summarized when they are similar, and given in detail when they differ.

DP Supported Stakeholders: The original artifact is shown on the left side of Figure 11.1, and the revised artifact on the right side of the same figure. With the SNIK DO as a knowledge base, the requirements engineer searched for stakeholders related to strategic information management and replaced unknown stakeholder roles with those from the DO. According to Table 11.3, all four RQAs are relevant to this DP. The ratings of the three participants (m4 and m6) are shown in Table 11.5. Both domain experts found a major conflict in the original artifact (m6.1.1), which they rate as problematic. Domain expert 1 found that the *IT strategy stakeholder* is the superclass of all other stakeholders. Domain expert 2 found the stakeholders *IT strategy steering committee* and the *IT board* to be similar or identical. This conflict was solved in the revised artifact (m6.1.2). All participants

[1] A web application to publish RDF data from a SPARQL endpoint such as the one from the SNIK ontology visualization is LodView: `https://github.com/dvcama/LodView`

Table 11.5: Rating of the requirements quality attributes for DP *Supported Stakeholders*. Metrics: **o**riginal / **i**mproved, n/a not applicable, – none given or found; **severity**: 👍 unproblematic, 👉 neutral, 👎 problematic; **completeness rating**: + sufficiently described, ○ partially sufficiently described, – insufficiently described; **understandability** (underst.): ○○ very easy, ○ easy, ● difficult, ●● very difficult; **correctness**: ✗ incorrect, ✓ correct; **details**: ∅ none, ↑major, ↓ minor.

Role		**metric**	Requirements engineer		Domain expert 1		Domain expert 2	
			o	i	o	i	o	i
conflicts	details	mx.1.1	↓	–	↑	–	↓	–
	severity	mx.1.2	👍	n/a	👎	n/a	👉	n/a
completeness	rating	mx.2.1	–	+	–	–	–	+
	severity	mx.2.2	👍	n/a	👎	👉	👎	n/a
interpretations	rating	mx.3.1	∅	∅	yes	∅	yes	yes
	severity	mx.3.2	n/a	n/a	👎	n/a	👍	👍
underst.	degree	mx.3.3	○○	○○	●	○○	○	○○
synonyms	details	mx.3.4	yes	no	yes	no	yes	no
	severity	mx.3.5	👍	n/a	👎	n/a	👎	n/a
correctness	rating	mx.4.1	✗	✗	✗	✓	✗	✓
	severity	mx.4.2	👍	👍	👎	n/a	👉	n/a

agree that the original artifact is insufficiently described, since role descriptions are missing (m4|6.2.1), although this is unproblematic (m4.2.2). As the revised artifact contains the stakeholders from the SNIK definition, providing a role description, the requirements engineer and domain expert 2 rated the revised artifact as sufficiently described. Domain expert 1 finds in the revised artifact some subset relations of stakeholders (*CIO*, *CFO* and *CEO* all make up a subset of the *project management board*, *information management board* and the *hospitals board of directors*). Domain expert 1 identified the different lines of interpretation in the original artifact unproblematic. The ambiguity was removed in the revision (m6.3.1 and m6.3.2). Domain expert 2 found no ambiguities. The original artifact was difficult to understand for domain expert 1 due to the unclear relations of the stakeholders. Domain expert 2 rated both artifacts as easy to understand. All participants rated the revised artifact as very easy to understand (m4|6.3.3). All participants found synonyms in the original artifact, and both domain experts rated them as problematic (m4|6.3.5). In the revised artifact, all participants agreed on the absence of synonyms (m4|6.3.4). All participants rated the original artifact as incorrect (m4|6.4.1), where the severity is problematic or neutral, as there were role descriptions missing and subset relations between the roles. In summary, the consistency, completeness, unambiguity, and correctness was improved in the revised artifact, but minor problems with subset relations were introduced.

Task	~~IT strategy~~ Strategic information management
Purpose	Creation and use of strategic information management plan
Source	Interview with CIO, 28. June 2016; revision with domain knowledge
Responsible	CIO Mr. X

ID	**Subtask**
1	~~Creation of IT strategy~~ creation of strategic information management plan
1a	Analysis and assessment of the current HIS state
1b	Description of the current HIS state
1c	Description of the planned HIS state
	includes former subtask 1a, 1b, 1c, 1d, 1e, 1f
1d	~~1g Presentation of IT strategy~~ Approval to Strategic Information Management Plan
2	Use of ~~IT strategy~~ strategic information management plan
2a	~~Prioritization of projects~~ Portfolio Management
2b	Evaluation of change requests
2c	~~Prospective and retrospective justification of IM department finances.~~ IT investment justification
2d	Evaluation of security-criticality of projects
3	added: updating the strategic information management plan
— needs to be elicited by the requirements engineer —	
4	added: deployment of strategic information management plan
— needs to be elicited by the requirements engineer —	

Figure 11.2: The revised task Strategic information management with domain knowledge.

DP Stakeholders' Tasks: The original artifact is presented in Table D.3 on page 232 and the revised artifact is presented in Figure 11.2. With the SNIK DO as a knowledge base, the requirements engineer searched for tasks related to strategic information management. The requirements engineer found several hierarchical tasks in this context in the ontology and used them to rename tasks of the task list. He found two additional tasks, which complete the original task list. According to Table 11.3, all four RQA are relevant to this DP. The ratings of the three participants (m4 and m6) are shown in Table 11.6. The requirements engineer found new unproblematic conflicts (m4.1.1 and m4.1.2) which are introduced in the revised artifact. In contrast, the domain expert 1 found that existing major problematic (m6.1.2) conflicts (m6.1.1) were solved in the revised artifact. The domain expert 2 found no conflicts. The requirements engineer found that the completeness has been improved in the revised artifact (m4.2.1). The domain expert 1 rated both artifacts as incomplete, but the improved artifact as less problematic (m6.2.1 and m6.2.2). In contrast, domain expert 2 rated the revised artifact as problematic in its incompleteness, since it provides a different view on the task and somehow mixes up the workflow- and contents layer (m6.2.2.). Whilst the requirements engineer found no different interpretations (m4.3.1), the domain expert

Table 11.6: Rating of the requirements quality attributes for DP *Stakeholders' Tasks*. Legend for metrics: see Table 11.5

Role		**metric**	Requirements engineer		Domain expert 1		Domain expert 2	
			o	i	o	i	o	i
conflicts	details	mx.1.1	∅	↓	↑	–a	–	–
	severity	mx.1.2	n/a	👍	👎	n/a	n/a	n/a
completeness	rating	mx.2.1	○	+	–	–	–	–
	severity	mx.2.2	👉	n/a	👎	👉	👍	👎
interpretations	rating	mx.3.1	∅	∅	yes	∅	no	yes
	severity	mx.3.2	n/a	n/a	👎	n/a	n/a	👎
underst.	degree	mx.3.3	○○	○○	●	○	○	●
synonyms	details	mx.3.4	yes	no	yes	no	yes	no
	severity	mx.3.5	👉	n/a	👎	n/a	👍	n/a
correctness	rating	mx.4.1	✗	✓	✓	✓	✗	✗
	severity	mx.4.2	👍	n/a	n/a	n/a	👍	👎

1 did. However, both rated the different interpretations as resolved in the revised artifact (m6.3.1). In contrast, domain expert 2 rated the revised artifact as problematic due to the different possible interpretations that are based on the mentioned conflicts in m.6.2.2. The understandability of the artifact has improved in the revised artifact in the eyes of domain expert 1 (m6.3.3). Not so for domain expert 2, who found the understandability of the original artifact as easy and that of the revised artifact as difficult. Both the requirements engineer and both domain experts found synonyms in the original artifact that were not present in the revised artifact. Both the requirements engineer and domain expert 1 rated the revised artifact as correct. In contrast, domain expert 2 rated the revised artifact as incorrect, as the artifact is ambiguous and insufficiently described and should be revised. In particular, domain expert 2 mentioned that the requirements engineer interpreted the SNIK ontology incorrectly, which led him to mixing up two different views to do with creating the strategic IM plan. In summary, there is no clear evidence of a requirements improvement. Domain expert 1 and 2 have basically different ratings. New conflicts were introduced in the revised artifact, as the SNIK ontology was obviously interpreted in an insufficient way. The resulting artifact still has plenty of room for improvement. The newly added subtasks *updating and deployment of the strategic IM plan* now need to be discussed with the stakeholder.

DP As-Is Activities and To-Be Activities: The decisions of both the DPs *As-Is Activities* and *To-Be Activities* are documented in a single artifact. The original artifact is presented in Table D.4 on page 233 and the revised artifact is presented in Table 11.11 at the end of this chapter, due to its length. This artifact was revised using the SNIK DO as a glossary. The requirements engineer searched the ontology for the terms that were taken from the original

Table 11.7: Rating of the requirements quality attributes for DP *As-is* and *To-be Activities*. Legend for metrics: see Table 11.5.

Role		**metric**	Requirements engineer		Domain expert 1		Domain expert 2	
			o	i	o	i	o	i
interpretations	rating	mx.3.1	yes	∅	yes	∅	no	no
	severity	mx.3.2	👎	n/a	👎	n/a	n/a	n/a
underst.	degree	mx.3.3	○	○○	●	○	○	●
synonyms	details	mx.3.4	yes	no	yes	no	no	no
	severity	mx.3.5	👎	n/a	👉	n/a	n/a	n/a

subtask description and linked the identical terms to the SNIK DO. When he could not find certain terms, he searched the ontology for similar concepts. When the requirements engineer was sure that a concept from the ontology matches the meaning in the subtask description, the ontology term was adopted and linked. According to Table 11.3, InR.unambiguity is relevant to this DP (m4|6.3.x), as shown in Table 11.7. Different interpretations were present in the original artifact. Both the requirements engineer and both domain experts agreed that the revised artifact is free from different interpretations (m4.3.1 m6.3.1). Both the requirements engineer and domain expert 1 agreed that the understandability of the revised artifact was improved (m4.3.3 and m6.3.3). However, domain expert 2 found that the revised artifact was more difficult to understand but gave no reason for this drop in understandability. Whilst the original artifact had synonyms, the revised artifact did not (m4.3.4 and m6.3.4). In summary, the unambiguity was improved in the revised artifact, but other requirements defects were introduced. In particular, these defects are missing information regarding the current and future HIS state and the migration path (planned actions). Overall, domain expert 2 made the following comment about the revised artifact: "The descriptions are more precise in the revised artifact. However, the planned actions (migration path) are missing.". The domain expert 2 wondered why the improvements from the artifacts of DP *Supported Stakeholder* and *Stakeholder' Tasks* were not included. As this is a cascading effect, we chose not to realize these kind of changes. However, one conflict was introduced into the revised artifact, as shown in Table E.9 in the Appendix on page 251.

DP Domain Data: The original artifact is presented in Figure D.25 on page 240 and the revised artifact is presented in Figure 11.3. Changed or newly integrated entities are denoted in dark blue. The SNIK DO was used to consolidate the DDM. The requirements engineer manually compared the DDM with the SNIK DO. Entities of the DDM that could not be found as *entity type*-concepts in the ontology were replaced by concepts from the ontology which had a similar meaning. For the purposes of comparison, string similarity were used as well as

Table 11.8: Rating of the requirements quality attributes for DP *Domain Data*. Legend for metrics: see Table 11.5.

Role		**metric**	Requirements engineer		Domain expert 1		Domain expert 2	
			o	i	o	i	o	i
conflicts	details	mx.1.1	↑	∅	↓	↓	↓	–
	severity	mx.1.2	👍	👍	👉	👉	👍	n/a

relations of entity types to the task of strategic IM. According to Table 11.3, InR.consistency is relevant to this DP (m4|6.1.1 and m4|6.1.2). The ratings of the three participants (m4 and m6) are shown in Table 11.8. The ratings of conflict differ between the requirements engineer and both domain experts. In the original artifact, the requirements engineer found one unproblematic (m4.1.2) and major conflict (m4.1.1), and which is solved in the revised artifact. The domain entity `investment` conflicts with `project`, as the budget of a project is an investment and an association is missing between both entities. The revised artifact contains the entity *exoneration protocol*, which is necessary but not explicitly mentioned in the task description, and thus it is not rated as a conflict. The domain expert 1 argued that CEO and CFO were missing in the revised artifact, which is the result of the omitted cascading effects. All three participants agreed that there were no different interpretations in the revised artifact, whereas domain expert 1 found an improvement (m4|6.3.1 and m4|6.3.2). All three participants unanimously agreed that the understandability of the revised artifact is made easier (m4|6.3.3). The requirements engineer and domain expert 2 both found synonyms in the original artifact that were eliminated in the revised artifacts (m4|6.3.4 and m4|6.3.5). All three participants unanimously found the original artifact to be incorrect and the revised artifact to be correct. Domain expert 1 found in the revised artifact the relationship between roles, tasks and entity types to be improved and corrected. In summary, the consistency, unambiguity and correctness of the revised artifact has been improved.

11.2.4 Additional Evaluation of DP System Functions

Since there was no original system function description, the requirements engineer created a new artifact following the *immediate use of domain knowledge* approach, as has been explained in subsection 10.1.2 on page 138. The result is a system function description with domain knowledge, as is presented below in Figure 11.4. The requirements engineer used the SNIK DO as a source for features, application components, hospital functions, and entity types that were used in the system function description. According to Table 11.3, the RQAs InR.consistency, InR.unambiguity, and InR.correctness are relevant to this DP (m4|6.1.x,

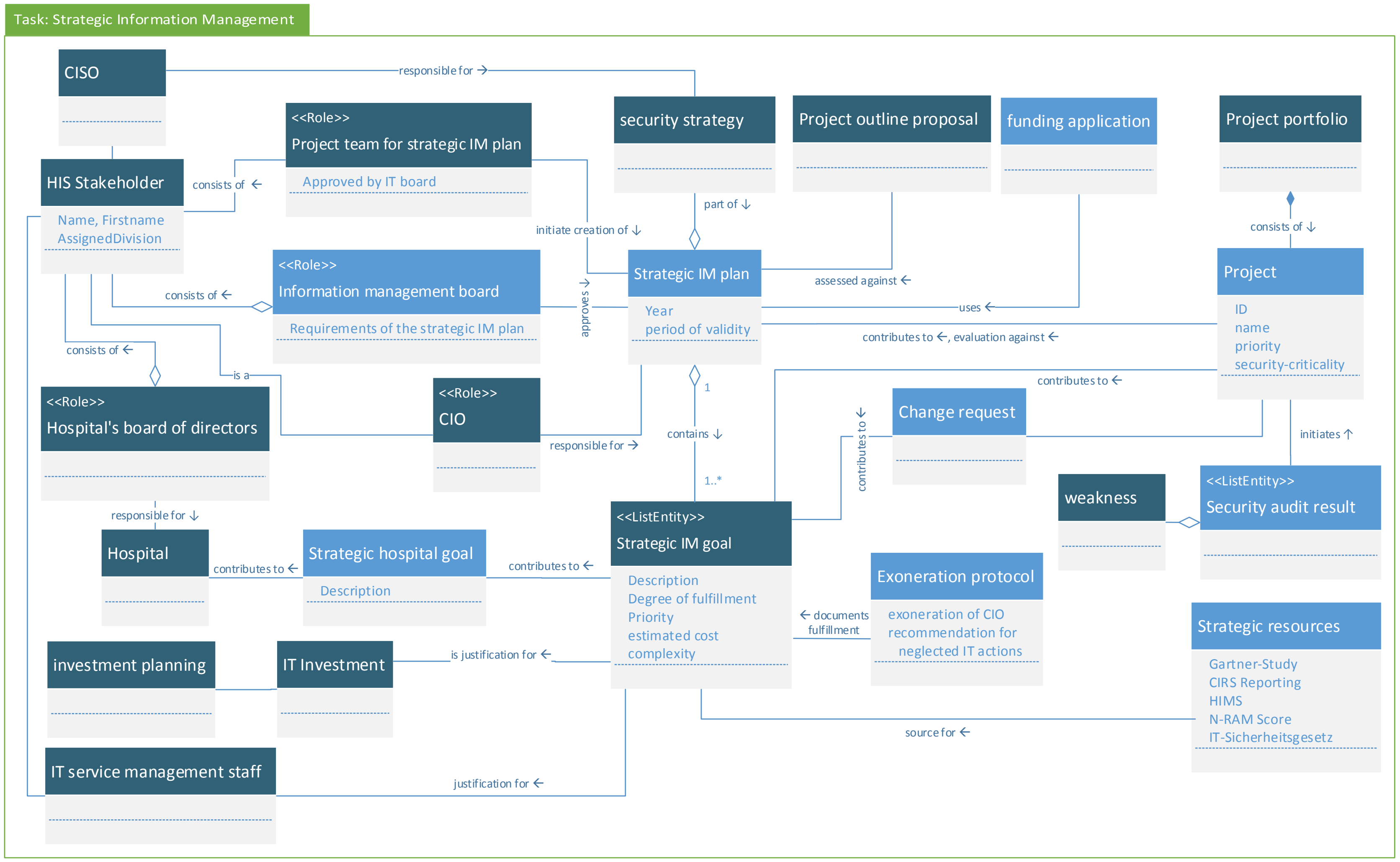


Figure 11.3: The revised domain data model

Table 11.9: Rating of the requirements quality attributes for DP *System Functions*. Legend for metrics: see Table 11.5

Role		**metric**	Requirements engineer i	Domain expert 1 i	Domain expert 2 i
conflicts	details	mx.1.1	↓	–	–
	severity	mx.1.2	👍	n/a	–
interpretations	rating	mx.3.1	∅	∅	no
	severity	mx.3.2	n/a	n/a	n/a
underst.	degree	mx.3.3	○○	○	●
synonyms	details	mx.3.4	yes	no	no
	severity	mx.3.5	👍	n/a	n/a
correctness	rating	mx.4.1	✓	✗	✓
	severity	mx.4.2	n/a	👎	n/a

m4|6.3.x, and m4|6.4.x). Table 11.9 shows the ratings of the three participants (m4 and m6).

The requirements engineer identified one conflict (m4.1.1), which is unproblematic (m4.1.2). The system function contributes to the task *strategic HIS directing*, which is mentioned in the revised task description of DP *Stakeholders' Tasks*, but is not further explained. Both domain experts rated the artifact as free from conflicts. No different interpretations of the artifact were found by the requirements engineer (m4.3.x) and both domain experts (m6.3.x).

The requirements engineer found the artifact as *very easy* (m4.3.3), the domain expert 1 as *easy*, and domain expert 2 as *difficult* to understand (m6.3.3). One unproblematic synonym (m4.3.4 and m4.3.5) was found by the requirements engineer: the terms *assemble* and *aggregate*. Both domain experts did not find any synonyms. The domain expert 1 rated the artifact as incorrect (m6.4.1) and problematic (m6.4.2). Instead of the term *strategic hospital goals*, it should be *strategic information management goals*. We rated this kind of defect as unproblematic, as this defect would have been resolved in the review of the requirements by the stakeholder. In summary, the artifact fulfills the quality attributes consistent and unambiguous, but has a minor deficiency in correctness.

11.2.5 Domain Expert's Feedback in General

Additional to the above, the domain expert 1 gave feedback on the use of domain knowledge in the artifacts in general. During the answering of the questionnaire, the domain expert 1 used the SNIK domain ontology for review purposes. The domain expert 1 stated that the

Figure 11.4: A system function description supporting the subtask Strategic Information Management 2c.

Field	**Requirements content**	**Ontology mapping**
Name	Assemble project costs for strategic hospital goals. (no adequate feature found in ontology)	feature
Realized in	internal reporting system	application component
Supports	strategic HIS directing	hospital function
Input data	strategic hospital goal, strategic information management goal, project portfolio, project cost	entity type
Output data	cost of strategic hospital goals	entity type
Description	The expenses and effort spent on projects are aggregated for the total cost and effort of the fulfillment of strategic information management and strategic hospital goals.	–
Quality req.	The presented data must be reliable.	–
Preconditions	The contribution of a project to strategic information management goals must be assigned in advance of the project portfolio. The contribution of strategic information management goals to strategic hospital goals is contained in the strategic information management plan. The degree to which a project contributes to strategic hospital goal must be predefined manually.	–
Post conditions	The actual total cost of the fulfillment of hospital goals is known.	–

ontology-based approach made sense in her eyes and is advantageous. The domain expert 1 realized some requirements defects, as the details of the as-is and to-be activities became altogether clearer in the revised artifact. Overall, the domain expert 1 rated the revised versions of the artifacts as more understandable. The use of familiar language terms from the ontology also revealed that the domain expert misunderstood some facts in the original artifacts. However, the domain expert 1 recognized a disadvantage of the use of the standard domain terms. Some stakeholders might be familiar with another domain 'dialect', that is to say the use of similar but different terms, or otherwise terms with a distinct meaning as contained in the ontology. Such stakeholders will however have disadvantages when the requirements are documented with standard domain terms and not in their familiar language. In any case the modified requirements must be reviewed with the stakeholders.

The domain expert 2 in general has a more negative view of the revisions. However, she admitted that in the DP *Supported Stakeholders* both synonyms and the very general stakeholder role "IT strategy stakeholder" were removed. In the DPs *As-is* and *To-be Activities* she found that descriptions were more precise in the revised artifact. Finally, in the DP *Domain Data*, inconsistencies were removed in the revised artifact.

Table 11.10: Requirements engineer and domain experts ratings of the improvement effect on the artifacts (m5 and m7)

Decision Point	Requirements engineer m5	Domain expert 1 m7	Domain expert 2 m7
Supported Stakeholders	slightly improved	greatly improved	slightly improved
Stakeholders' Tasks	greatly improved	greatly improved	slightly worsened
As-is and To-be Activities	greatly improved	slightly improved	slightly improved
Domain Data	greatly improved	greatly improved	greatly improved

Ratings of the Improvement Effect on the Artifacts: Both the requirements engineer's (m5) and the domain expert's (m7) ratings of the improvement effect on the artifact are given in Table 11.10. Altogether, the requirements engineer rated the revised artifacts predominantly (three out of four) as *greatly improved* (m5). The domain expert 1 also rated the revised artifacts predominantly (three out of four) as *greatly improved* (m7). The rating of domain expert 2 differs from the previous two ratings. Domain expert 2 rated one revised artifact as greatly improved, two as slightly improved, and one as slightly worsened (m7). Nevertheless, the application of domain knowledge led in most cases to an improvement of the RQAs. In summary, we answer RQ.2.2.3 by stating that TOREOnto provides a useful way to improve requirements quality attributes. The outcome of the revision provides a high utility for requirements engineers and stakeholders who are represented by the involved domain experts.

11.2.6 Cascading effects

Some changes in one artifact propagate changes in other artifacts. This is referred to as a cascading effect. When such a cascading effect occurs, the requirements engineer also needs to change the impacted artifacts. Below, the cascading effects of changes in one DP and their respective impacts are given:

1. *Supported Stakeholders*: Through the revision of the stakeholders, the *Information Management Board* gains a higher importance, as it is involved in more tasks than is originally visible. Thus, this group needs to be interviewed about both additional tasks, and existing ones (Strategic Information Management), which will be extended and refined. This influences the DPs *As-is* and and *To-be Activities*, *Domain Data*, and finally the DP *Interaction*.

2. *Stakeholders' Tasks*: The more detailed definition of the stakeholders' tasks directly influences the DPs *As-is* and *To-be Activities*.
3. *As-is* and *To-be Activities*: The replacement of terms and removal of synonyms leads to changes in the DPs *Domain Data* and *Stakeholders' Tasks*. As a result, this also affects the DP *interaction*.
4. *Domain Data*: Different entity terms lead to changes in the DP *Interaction* and to changes in DP *System functions* due to the different naming of entities.

11.2.7 Answer to RQ.2.2

In summary, we answer RQ.2.2 by stating that TOREOnto supports the application domain-specific improvement of requirements through the application of domain knowledge in a DO overall in a good way. The RQAs consistency, completeness, unambiguity and correctness are all improved. However, TOREOnto does not replace the verification of requirements with stakeholders.

11.3 Discussion

In this section, we discuss the results of the TOREOnto evaluation regarding the revised artifacts, the TOREOnto method, as well as the lessons learned during the method adaptation.

11.3.1 Discussion of the Revised Artifacts

Some requirements defects are mentioned multiple times by the study's participants in the questionnaire, for example CFO/CEO as conflict, synonyms and issues regarding correctness. This indicates difficulties in distinguishing between the requirements defects precisely. The improvement of consistency is very clear for the DP *Domain Data*. The improvement of unambiguity for the DP *As-is* and *To-be Activities* and *Stakeholders' Tasks* is also very clear due to the use and linking of standard domain terms. The improvement of completeness and correctness in the DP *Stakeholders' Tasks* is not reliably evident. The RQAs consistency, completeness, and correctness all improved for the DP *Supported Stakeholders*, although there were some subset relationships. Overall, the revised artifacts become more clear by the use of standardized domain terms in general. As a result, the domain experts found some other requirements defects that were previously hidden. On its own, this effect is an added value for both requirements engineers and stakeholders. Unfortunately, the revision did lead in some cases to a deterioration, as new defects are introduced to the requirement. This is clearly due to the human factor in the revision of the artifacts. Even when utilizing formalized knowledge, there is a risk of misunderstanding or applying the knowledge in an incorrect way. Consequently, it is essential to not only rely on the application of domain knowledge but also to review and approve the requirements with the stakeholders. However, one influence on this factor is the definition of TOREOnto, which focuses on manually applicable DO usage patterns. The results might change with patterns that allow a higher degree of automatism, such as the reason about defects pattern. The effort investigated for RQ.2.2.1 is very subjective. Requirements engineers with different degrees of experience in using the SNIK DO will influence the effort positively or negatively. The rating of whether the effort is acceptable is also influenced by other factors, such as time pressure and predefined quality restraints of the requirements.

11.3.2 Discussion of TOREOnto

The TOREOnto method, as is defined in Table 10.3 on page 140, provides the guidance for the requirements engineer to revise requirements with domain knowledge as an improvement step in RE activities. However, it is essential to define the appropriate DO usage patterns for each DP in advance of the revision. The decision as to where to apply which DO usage pattern must be motivated by the goal of which of the RQAs should be improved. All of the primary studies analyzed in the literature review in Chapter 9 on page 97 propose an approach to use DOs, but most of them are not integrated into larger RE processes. TOREOnto fills this gap.

The *TORE DO usage framework* supports two different types of approaches for applying the domain knowledge. TOREOnto follows the *subsequent use of domain knowledge* approach, which is applicable to all DO usage patterns that need some initial requirements which are to be improved. The *immediate use of domain knowledge* is applicable when requirements are initially documented, and the relevant RQAs are known in this early stage. This approach works for all DO usage pattern that need no requirements (InR or SRS) as an input, as is depicted in Table E.8 in the Appendix on page 250. The complete requirements template pattern is the only DO usage pattern which does not consume requirements as an input. However, this pattern can be re-applied by a requirements engineer to previously created requirements with that same pattern. The results of this case study shows that the *subsequent use of domain knowledge* for the DPs *Supported Stakeholders*, *Stakeholders' Tasks*, *As-is Activities*, *To-be Activities*, and *Domain Data* all led to improved requirements. The specification of DP

System functions shows as an additional evaluation the principal feasibility of the *immediate use of domain knowledge*. A requirements engineer should use the *immediate use of domain knowledge* approach when both the DO and the RQAs which should be improved are known at the point in time, when the initial requirements are documented. A requirements engineer should use the *subsequent use of domain knowledge* approach when the initially specified requirements do not fulfill the intended RQAs, or when the DO is not available at the time of initial requirements documentation.

The use of the DO as a knowledge base was very easy for the requirements engineer, but led to the above discussed conflicts that were introduced. The use of the DO as a glossary was also very easy for the requirements engineer and led to an prominent improvement. The improvement effect differs for some artifacts and some patterns. In general it is helpful to extract such domain terms which will be used in all requirements. As the vocabulary of domain terms in the requirements might constitute a subset of the whole DO, this makes easier the reusability of domain terms in other requirements artifacts. In order to provide consistency across the whole SRS it is helpful to have a clear understanding of what exactly has changed during the revision in one artifact. Such changes are candidates for changes in other requirements of the SRS.

11.3.3 Lessons learned

Overall, the ontology and the contained formalized domain knowledge are both helpful to improve the requirements artifacts. The guidance of TOREOnto was excellent in combination with the DO usage pattern. TOREOnto defines the *what* and the *how* aspects of the improvement. The *what* aspect is the RQAs, which shall be improved on the appropriate DP and thus on the appropriate requirements artifacts. The *how* aspect is defined by the DO usage pattern that provides the detailed guidance to apply domain knowledge to the requirement. Thus, it was very clear for the requirements engineer what to improve and how to use DO in the revision activities. However, the search for domain knowledge is not always easy. As a first approach, a string or substring search in the DO or its visualization provides some first hits. When these results are not useful, the search should be extended to synonyms. General language tools such as lexicons and dictionaries are helpful. As a second approach, it is beneficial to use the relations between concepts in the DO. In some cases, when two relevant concepts are far apart in the ontology, their relation to each other is not clear. Then the search for paths between these concepts is helpful. The SNIK ontology supports these searches for alternative paths by the *starpath* and *spiderworm* features. These features reveal relations by highlighting multiple other concept relations. This consideration of the concept's neighbors contributes to the requirements engineer's DO knowledge and supports the decision of whether the domain knowledge is correct or not.

The effort to apply the DO usage pattern varies according to the pattern. Patterns such as use DO as knowledge base and complete requirements template pattern are easy to apply with a low level of effort. The use individuals as glossary terms pattern requires – when deployed manually – a higher level of effort. The reason for this relates to the difficulties to search for terms and to find the correct correlating DO concepts.

11.4 Threats to validity

The following types of threats to validity are structured according to Runeson et al. [RH09].

Construct validity: Construct validity considers whether the study really measures what it claims. The main threat to construct validity is that the CIO as the stakeholder for the task strategic IM did not review the revised artifacts. Thus, the revised artifacts might not represent precisely his initial needs or thoughts. We mitigated this threat by considering the ratings of two domain experts, who at least understand the domain concepts that are inherently present in the requirements. However, the domain experts' ratings of the improvements differed in some cases.

Internal validity: Internal validity aims to ensure that the collected data enables researchers to draw valid conclusions, and in particular to minimize the effects of unknown factors that influence an investigated factor. The selected patterns in TOREOnto were executed manually without any tool support, except for the SNIK ontology, search facilities and the ontology visualization. The results of this study will change when using a higher degree of automation in the application of DO usage patterns. For example, the adoption of automated term extraction and ontology mapping techniques, supporting of the selected pattern, or the reason about defects pattern will lead to other results. The domain experts are neither especially familiar with the types of RE artifacts nor with the details of RQA. There is a chance that the domain experts had a different understanding of quality attributes and tended to rate the different RQAs as indistinct. For example, one of the domain experts had difficulties in making a clear distinction between completeness, unambiguity, correctness and consistency, since these attributes are closely related within one artifact. We mitigated this threat of internal validity by providing an explanation of the selected RQAs in the questionnaire. The translation of the RE artifacts, which were originally in German, might have influenced the quality of the resulting RE artifacts in English. This might have effects on the observed artifacts improvements.

External validity: External validity describes the generalizability of the findings and the transfer of our study results beyond the study. The case study is based on a single case only and involved only one requirements engineer and two domain experts. In order to be representative, a large group has to be selected, which was however unavailable. The results are specific to the application domain IM in hospitals. The change of study results in other application domains or with other DOs is unclear and is a point that is left to future work.

Reliability: Reliability considers the influence of the specific researcher and indicates threats to validity for a repetition of the study. The main threat to reliability is that the author of this thesis in his role as requirements engineer had a large influence on the development and process of the case study, in particular during the interviews and the data analysis. We mitigated this aspect through continuous discussion with the supervisor of this thesis. A major threat to reliability is that the requirements engineer is biased in two ways. First, he gained domain knowledge during his involvement in the creation of the SNIK DO, prior to the revision of artifacts with TOREOnto. Thus, he was familiar with the structure and some parts of the SNIK DO. Second, he created the initial RE artifacts which built the basis for the improvement. However, we believe that the second bias in fact contributes positively to the revision results. A minor threat to reliability is that another requirements engineer would reach to other revision results. We could not mitigate this alleged threat, but encourage other researchers to repeat the evaluation study with the same or other kinds of RE artifacts. The rating of the effort of TOREOnto was carried out only by the author of this thesis, which cannot be unprejudiced. We mitigated this aspect by presenting the absolute numbers of consumed time. Obviously, there is a bias, as the requirements engineer has created the DO usage patterns and is familiar with their fundamental ideas.

11.5 General Experiences with Method Adaptation

TORE is not extended by application domain knowledge directly. Rather, TOREOnto has extended some of TORE's DPs through the use of externalized domain knowledge in DOs. The use of DOs without coupling TORE directly to domain knowledge is an advantage, as it enables the adaptation to other application domains in a similar way, by simply using other DOs. The TOREOnto adaptation is based on the *extension* of an DP to re-validate previously made decisions. Elicited requirements are refined and specified according to domain knowledge. The DO usage patterns provide the details of the domain knowledge application to requirements artifacts. Thus, the essential aspect of the method adaptation is to decide on which DPs which DO usage patterns will be used. This decision depends on two aspects. First, it requires clear objectives about the intended RQA that is to be improved. This directly influences the set of possible patterns to choose from. Second, it requires the exact knowledge of artifact types that are used to document the requirements. Depending on the artifact type, some DO usage patterns cannot be applied. During the TOREOnto method adaptation, it was not questionable to modify the structure or notation of the requirements artifacts itself, as this was the case for the DsTORE adaptation. The reason is that the domain knowledge can be applied in manifold ways to requirements artifacts and there are several DO usage patterns which support natural language requirements and models.

Another aspect during the method adaptation is the decision about the intended level of automation. The effects of the TORE adaptation are also based on the level of automation and tool support for the requirements engineer. A higher degree of automation will reduce the effort for the revision activity. Such an automation is demonstrated by a Bachelor thesis, where the use DO as glossary pattern and the complete requirements template pattern are realized in a semi-automatic way in the context of the SNIK ontology [Rod17]. The approach that was implemented in the Bachelor thesis supports the requirements engineer during the documentation of tasks and subtasks. During the filling out of the task- and subtask templates in the issue-tracking system JIRA[2], the tool automatically searches for glossary entries in the SNIK ontology and automatically links appropriately selected terms.

The design science approach was helpful to structure the method adaptation. The *problem investigation* revealed the DO usage pattern that were coupled with TORE's DPs in the *treatment design*. The *treatment evaluation* showed the principal usability and utility of the treatment design. In summary, TORE could easily be adapted to an application domain by extending its DPs to connect to the DO usage patterns.

11.6 Chapter Summary

In this chapter, the treatment design TOREOnto was evaluated in a *case study* using a part of the requirements specification for CIONx. A requirements engineer revised the requirements of the CIO's task IT strategy according to TOREOnto, with the goal of improving the RQAs

[2] https://www.atlassian.com/software/jira.

consistency, completeness, unambiguity and correctness. The evaluation covered the DPs *Supported Stakeholders*, *Stakeholders' Tasks*, *As-is Activities*, *To-be Activities*, *Domain Data* and *System Functions* of the CIONx SRS. Two questionnaires, one for the requirements engineer and one for each of the two domain experts, were used to gather the feedback and acceptance of the method application and the improvement effect of the requirements quality. To sum up, the quality of the requirements has been improved, although the task description related to DP *Stakeholders' Tasks* has been slightly worsened due to a misinterpretation of the SNIK ontology. Overall, the application of the domain knowledge to the requirements artifacts is a good way to improve its quality. However, the application of domain knowledge does not replace the validation of requirements with stakeholders. The case study has shown that TOREOnto is a usable and utilisable method extension to enable the improvement of requirements.

Table 11.11: The improved task IT strategy in Task & Support template

Task	~~IT strategy~~strategic information management plan
Purpose	Creation and use of ~~IT strategy~~strategic information management plan
Source	Interview with CIO Mr. X, 28. June 2016; Translated by C.Kücherer, 2017-10-24, revision with domain knowledge
Responsible	CIO Mr. X

ID	**Subtask**	**As is solution**	**Example solution**
1	Creation of IT strategy		
1a	Approval of stakeholder for IT strategy. **Problem:** Important stakeholders for the creation of strategic information management plan must not be forgotten.	The information management board has to be formed prior to the creation of a new strategic information management plan. Participating stakeholders must be associated with the divisions (but are not limited to) PatientCare, EducationResearch, information management. HIS stakeholder state their requirements and expectations regarding the strategic information management plan, representing the opinion of their associated organizational units of the hospital. The proposal of information management board participants is presented to the hospital's board of directors for approval.	Task will not be supported by CIONx.

1b	Prioritization of future fields of action for strategic information management plan.	Possible areas and their respective priority that will be addressed in the strategic information management plan are gathered in workshops with the information management board. For the last strategic information management plan, 14 workshops have been carried out. Main questions are "which IT related problems currently exist in the organization?" and "which important topics exist in the IM department?". The identified topics are clustered in coarsely grained topics from which larger Strategic Information Management Goal are synthesized. Each Strategic Information Management Goal can be assigned to one or more strategic hospital goals.	The identification of relevant Strategic Information Management Goal can be supported by current Gartner studies and CIRS reporting (crititical incident reporting) for patient and patient care, which delivers indications for IT weaknesses. Further sources for Strategic Information Management Goal could be HIMSS and EMRAM.
1c	Conduct the workshops	The project team discusses the future fields of action in workshops, to gather Strategic Hospital Goals.	Task will not be supported by CIONx.
1d	Evaluation of workshop results	The analysis and synthesis of workshop results leads to the list of Strategic Hospital Goals. The CIO and department managers are involved in the synthesis.	Task will not be supported by CIONx
1e	Documentation of Strategic Hospital Goals	Based on the workshop results (subtask 1d), the project team prepares the catalog of Strategic Hospital Goals. These are large and broad topics that are textually described in the strategic information management plan.	Task will not be supported by CIONx.

1f	Prioritization of Strategic Information Management Goals. **Problem:** (1) consent about Strategic Information Management Goal; (2) Strategic Information Management Goals have a political dimension.	Each Strategic Information Management Goal is prioritized. Less important Strategic Information Management Goals are neglected. As the HIS stakeholders have different goals and requirements, there is not always consent about Strategic Information Management Goal. Some HIS stakeholders have concerns about the documented Strategic Information Management Goals.	Consensus can be reached by conflict resolution talks or social voting.
1g	Presentation of strategic information management plan	The CIO keeps the whole hospital informed about the existence and binding character of the strategic information management plan.	Task will not be supported by CIONx.
2	Use of IT strategy		
2a	Prioritization of project portfolio	New projects from project outline proposals or change requests are assessed against the existing strategic information management plan. Projects that contribute to high priority Strategic Information Management Goals are promoted and receive a high level of priority. Currently this assessment is implicitly made by the CIO.	CIONx could show the Strategic Information Management Goal which a project contributes to during the prioritization.

2b	Evaluation of change requests	The contribution of Change requests to Strategic Information Management Goal are evaluated by the CIO. Change requests that contribute to strategic information management goals will be accepted. Currently this assessment is implicitly made by the CIO.	Task will not be supported by CIONx.
2c	Prospective and retrospective justification of IM department investments. **Problem:** (1) the result of investment planning must be justified; (2) no explicit relation between projects and Strategic Information Management Goal; (3) Additional IT service management staff must be justified.	The hospital's board of directors questions planned investments. As the hospital's board of directors has committed to the strategic information management plan, investments or additional IT service management staff who contribute to Strategic Information Management Goals can be justified.	CIONx should show the contribution of projects to Strategic Hospital Goal for the use in presentations. Appropriate metrics should show the total cost and effort to reach a Strategic Hospital Goal.
2d	Evaluation of security-criticality of projects	Projects with an impact on IT security are evaluated in more detail.	A CISO requires the project contribution to security strategy (which is part of strategic information management). An overview of security relevant projects is a major result for the project prioritization.

2e	Use of strategic information management plan for funding applications	The strategic information management plan is used as the basis for funding applications.	Task will not be supported by CIONx.
2f	Evaluation of strategic information management plan adherence. **Problem:** Arguments for non-fulfilled strategic information management goals are missing.	Whenever a new strategic information management plan is created, the fulfillment of the current strategic information management plan is evaluated. Ideally, this evaluation would take place on a yearly base.	An information management board should be established who meet each half year to evaluate the fulfillment of the strategic information management goals. CIONx shows the relation of strategic information management goal to its planned budget, number of projects and total costs. This can provide arguments for a non-fulfilled strategic information management goal (too high costs).

Part IV

Summary

12 Insights into the Adaptation of a Generic RE-method

This chapter provides insights into the adaptation of a generic RE-method in order to answer the main research goal of this thesis. Section 12.1 presents the preconditions and the overview of the generic RE-method adaptation. The answer to the main research goal is given in Section 12.2, which presents the activities for the generic RE-method adaptation. These activities are based on the experiences gathered during the TORE method adaptations that have been described in Part II and Part III. Finally, the adaptations and their relation to design science are discussed in Section 12.3.

12.1 Preconditions and Overview

The research goal of this thesis, as was first described in Section 1.2 on page 6, can be restated as the following: *to investigate how a generic RE method can be adapted to consider system domain and application domain-specifics*. By a generic RE method, any method is meant that supports and guides requirements engineers in the use of notations or modeling techniques at different stages of the RE process [NE00]. Such RE methods have in common the goal of supporting the specification and documentation of requirements. We use the experiences of the TORE method adaptations DsTORE and TOREOnto and condense them to form principles of generic RE-method adaptation. These principles can be applied to other RE-methods, system domains and application domains in a similar way.

These insights have as their target audience *method designers* and *researchers*. We presuppose two aspects that researchers have in mind: first, a defined and existing **RE-method** to adapt, and second, the necessity to create a SRS in a defined application domain and for a certain system type. Initially, a researcher needs to decide whether to execute a system domain-specific or an application domain-specific adaptation. The *problem context* for the system domain-adaptation is an assumption of the system type. The *problem context* for the application domain-adaptation is to improve the requirements quality by domain knowledge. Figure 12.1 provides an overview of the RE-method adaptation activities as an activity diagram. The activities are then described in greater detail in the following section.

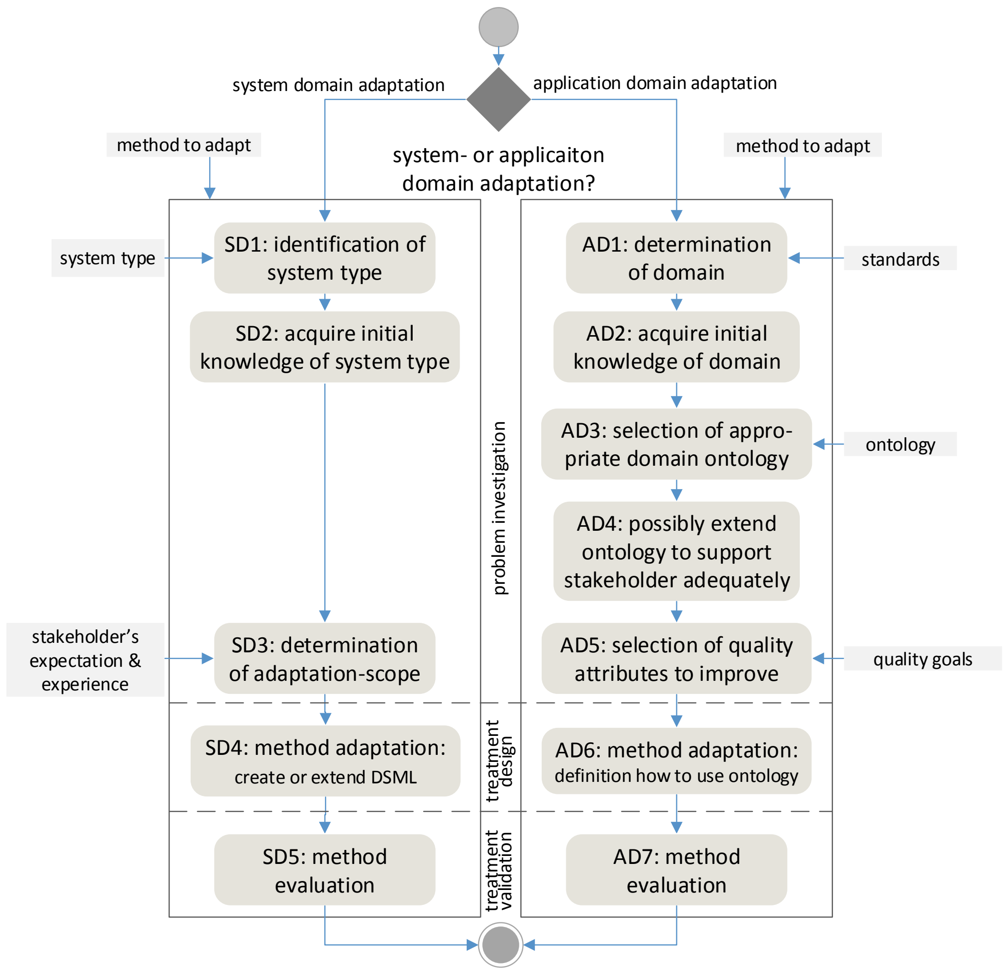

system domain adaptation
application domain adaptation
system- or applicaiton domain adaptation?
method to adapt
method to adapt
system type
SD1: identification of system type
SD2: acquire initial knowledge of system type
stakeholder's expectation & experience
SD3: determination of adaptation-scope
SD4: method adaptation: create or extend DSML
SD5: method evaluation
problem investigation
treatment design
treatment validation
AD1: determination of domain
standards
AD2: acquire initial knowledge of domain
AD3: selection of appro-priate domain ontology
ontology
AD4: possibly extend ontology to support stakeholder adequately
AD5: selection of quality attributes to improve
quality goals
AD6: method adaptation: definition how to use ontology
AD7: method evaluation

Figure 12.1: Overall activities of the RE-method adaptation

12.2 Principles of Generic RE-method Adaptation

This section explains our proposal of the principles to adapt generic RE-methods. The experiences of the system domain-specific adaptation, as was first presented in Section 7.6 on page 88, are condensed to the principles shown in the left side of Figure 12.1. The experiences of the application domain-specific method adaptation, as was first presented in Section 11.5 on page 172, are condensed to the principles shown in the right side in Figure 12.1.

12.2.1 The System Domain-specific Adaptation

Adaptation activities: Initially, a researcher requires first ideas about the character of the to-be system. As the first **activity SD1**, it is necessary to identify and determine the system type, which is difficult. Solid answers to the following questions might be helpful for the researcher: (1) What exact type of system is it? (2) What details make up this system type? (3) How are the details described in the literature? (4) What are typical features of this system type? (5) What kind of problems should be solved by this system type? In order to answer these questions, the researcher needs to acquire knowledge of the system type in **activity SD2**. Literature, practical books, experts, software engineers or stakeholders are all a good source of details about the system type. Software engineers often have experience with such system types, from having previously worked on similar problem contexts. Stakeholders might also have a clear understanding of the system type, as they previously have worked with similar systems. In **activity SD3**, the exact specifics which are the focus of the system type need to be determined to shape the DSML. It is important to consider the stakeholders' capabilities, as they need to comment and understand the requirements artifacts that are created with the DSML. In **activity SD4**, the method is adapted. Details of the method, such as TORE's DPs, are changed by the researcher to consider the exact specifics of the system type. These changes affect the used *requirements artifacts*, and the *scope* of the method. Finally, in **activity SD5**, the design is validated. If the design is not satisfactory, a new iteration of the adaptation should be executed.

Design Science for the System Domain-specific Adaptation: Figure 12.1 also provides an assignment of the activities to the design cycle. The *problem investigation* (activities SD1, SD2, and SD3) were focused on the identification of missing system specifics. The *treatment design* (activity SD4) followed the consideration of the system specifics in the selected RE-method. Overall it was helpful at this stage to realize answers to following questions: (1) What kind of problem does the system type solve? (2) What is the theory behind this kind of problem? (3) Which methods are used in the system domain? For the given example we investigated decisions and their systemic support. For other system domains, one would try to understand the challenges of this system type: for example, what the challenges of resource planning are, or what are the fields of use for ambient assisted living. There were no specialties in the treatment validation (activity SD5).

12.2.2 The Application Domain-specific Adaptation

Adaptation activities: As the first **activity AD1**, a researcher needs to determine the application domain to which the method shall be adapted. There is no restriction on the application domain in general. Table E.4 in the Appendix on page 244 shows the manifold domains where ontologies can be applied. However, the researcher requires a basic understanding of the application domain, as documented by **activity AD2**. The domain knowledge provided by the DO is otherwise useless for the researcher and in the end also for the requirements engineers. Possible sources of further details are literature, domain experts, initial training or first experiences in an application domain. Some of the following questions are helpful to gain a first understanding of the application specifics: (1) Which roles are interesting in that domain? (2) Which tasks are related to these roles? (3) Which useful system functions are typically present in the domain? (4) Which entity types are useful? (5) Are there any details to nonfunctional requirements provided by the domain knowledge? (6) Which standards are relevant? The selection of the formalized domain knowledge in the form of a DO is essential for the results of the application domain-specific adaptation in **activity AD3**. Existing DOs can be reused [FLGPSF13], or new DOs might be created [OSS+10]. A clear vision of the structure and quality of the DO is necessary. The ontology needs to contain knowledge of the application domain that is relevant to the particular requirements. The experiences gained in this thesis indicate that this knowledge should cover roles, stakeholders' goals and tasks, information entity types, software features, application components, and their relations. The results of the domain knowledge application depend on the DO's quality and the level of details. The selected DO should be suitable for the specific stakeholders. The use of domain knowledge lacks some advantages, if involved stakeholders have a different understanding of their domain as described by the domain knowledge. The ontology used in this thesis has showed that some stakeholders in part employ a vocabulary that deviates from other stakeholders, as well as their understanding of some concepts and relations. For this reason, we extended the SNIK ontology by concepts and relations for the specific CIO Mr. X, which is called the CIOx ontology[1]. It might be necessary in **activity AD4** to extend, narrow, adapt or enrich the selected DO to match with the familiarity of the specific stakeholders. Boukhari et al. [BBJ12] argue that roles involved in software development projects should be free to use their favorite vocabularies to express requirements. They provide an ontology-based solution to support such different vocabularies. In **activity AD5**, the researcher selects the RQAs which will be improved in the requirements. The ISO/IEC standard 29148 [ISO11] defines the set of RQA, as was first

[1]The CIOx ontology is a separate section of the SNIK ontology. Concepts of the CIOx ontology have been related to appropriate concepts in the SNIK ontology by inter-ontology relations. The CIOx ontology has not been used for the evaluation presented in Chapter 11.

introduced in Table 2.1 on page 18. With the available DO usage patterns, only a subset of these RQAs can be improved. Currently, the improvement of the seven RQAs, namely *InR.complete*, *InR.correct*, *InR.unambiguous*, *InR.consistent*, *InR.traceable*, *SRS.complete*, and *SRS.consistent*, can be improved. The researcher needs to decide about the level of automation and thus the necessary infrastructure. The infrastructure need of the DO usage patterns has been described in Table 9.12 on page 127. The investigation of further ontology usage scenarios to improve other RQAs is left to future research. The core of the RE-method adaptation in **activity AD6** is to determine how the ontology is used in combination with the RE method. Therefore the DO usage patterns provide the knowledge of how to apply to DO to requirements artifacts. The researcher needs to choose the DO usage patterns according the selected RQAs and embed them into the RE-method, as we did with TOREOnto. The effort required to apply the DO usage pattern varies according to the pattern. Patterns like the use individuals as glossary terms pattern and the use domain ontology as knowledge base pattern are easy to apply with a low level of effort. By contrast, the reason about defects pattern requires a high level of manual effort or a high level of infrastructure by tool support, in particular due to NLP tasks. Thus, the possible level of automation needs to be decided. Some patterns are deployed manually or with a high degree of automation. The thesis of Rode [Rod17] shows how the complete requirements template pattern and the use individuals as glossary terms pattern can both be automated by tool support. However, all measures to improve requirements quality by adapting RE-methods do not replace the requirements verification with stakeholders. The reason for that is that even semantic technologies do not replace human experience and rationale. In the end, the requirements must represent what stakeholders have in mind.

Design Science for the Application Domain-specific Adaptation: The *problem investigation* (activities AD1 – AD5) were focused on the understanding of the application domain, the structuring of typical tasks and entity types, the understanding of roles, the RQAs, and the DO. The *treatment design* (activity AD6) designs how the application domain knowledge is used in the RE-method. The principles regarding how domain knowledge is applied in RE is independent from the application domain. There were no specialties in the treatment validation (activity AD7).

12.3 Discussion

The principles of both kinds of domain-specific adaptations are different from each other. For the system domain-specific adaptation, some DPs of TORE were added or changed. This results in the modification of the *scope* of DPs. To explain in greater detail, this means that the adapted DPs for the system domain-specific adaptation focuses on exactly the chosen system type with even more specific questions. Thus, new DPs are introduced, and the scope of existing DP has narrowed. In contrast, for the application domain-specific adaptation, some DPs have been extended by additional activities to incorporate domain knowledge. This results in the *extension* of an DP to re-validate previously made decisions. Elicited requirements are refined and specified according to domain knowledge.

Wieringas *design science* is a well-suited methodological approach for the adaptation. Following the design cycle, the *problem investigation* helps to identify current problems by applying the generic RE-method to the problem context. Altogether, we found the design science approach to be similar for both types of adaptations, as was indicated by Figure 12.1. We are convinced that the details for design science presented in this thesis helps other researchers to adapt other generic RE-methods in a similar way. Our experience is not to apply the design science approach to the adaptation of the RE-method in general, but rather to apply it to the two problem contexts, namely the system domain and the application domain. The *treatment design* allows researchers and requirements engineers to shape the solution by addressing the missing domain specifics and domain knowledge in the design that have been identified. Finally, the *treatment validation* allows researchers and requirements engineers to measure to what extent the resulting artifact fulfills the design.

13 Conclusion and Outlook

This chapter provides a summary of the main contributions of this thesis in Section 13.1. Finally, this work is rounded up with mention of some ideas for future work, presented in Section 13.2.

13.1 Summary

At a general level, this thesis contributes to the body of knowledge in RE with respect to domain-specific adaptations of RE-methods to system domains and application domains. The summary of these experiences constitute the solution of the abstract design problems 1 and 2, as were both presented in Figure 1.2 on page 6.

The combination of several activities provides the solution to design problem 1. The design problem 1 is given as *to improve TORE by designing DsTORE that satisfies the use of a decision-specific domain-specific modeling language, in order to support requirements engineers in providing a good system domain specific software requirements specification*. We conducted a domain study, that was presented in Chapter 4 on page 47, in order to gain first insights into the application domain IM in hospitals, which is the context for the studies presented in this thesis. The domain study focuses on tasks of the IM department in which the CIO is involved, on the CIO's role and on entities used for communication with the hospital management. The extent to which TORE supports the RE-specification of PDSSs is assessed in a case study, which is presented in Chapter 5 on page 59. This study reveals following neglected system domain specifics: (a) the explicit distinction of decisions from conventional subtasks; (b) detailed description of the data necessary for decision-making; (c) decision-specific rules to choose from decision alternatives; (d) decision-specific patterns in data for tables and summary fields; (e) relation between content, format, and location of data sources; and (f) explicit consideration of typical PDSS-specific system functions. The design problem 1 is solved by the system domain-specific method extension DsTORE, which is presented in Chapter 6. DsTORE is a DSML based on TORE which emphasizes system domain-specific requirements of a PDSS in the following ways: (1) the classification of tasks to distinguish decisions from conventional subtasks; (2) the use of an extended subtask description for the documentation of decisions, clearly stating decision relevant information; (3) an extended DDM that supports decision-specific data, which is often provided in semi-structured spreadsheets and unstructured text documents. (4) a new data source model, combining domain entities, their format and location to provide the information for decisions; and (5) guidance for system functions oriented on typical functions of the system domain. The DsTORE method adaptation is validated with a case study that shows its efficiency, usability, and utility regarding the specification of requirements for a PDSS.

The design problem 2 is given as *to improve TORE by designing TOREOnto that satisfies the use of application domain ontologies, in order to support requirements engineers in improving requirements quality*. The state of the art regarding the use of DO to improve requirements quality is investigated in a systematic literature review, which is presented in Chapter 9. The design problem 2 is solved by the application domain-specific method extension TOREOnto, which is presented in Chapter 10 on page 135. TOREOnto defines the usage of formalized application domain-knowledge in DOs to improve requirements in the following ways: (1) DO usage patterns describe the principles of how DOs are used to improve the RQAs InR.*completeness*, InR.*consistency*, InR.*unambiguity*, InR.*correctness*; (2) TOREOnto defines the application of DO usage patterns in some of TORE's DPs; and (3) TOREOnto guides the revision and improvement of requirements utilizing application domain knowledge. The method adaptation TOREOnto is retrospectively validated with a case study in Chapter 11 on page 151. The study shows TOREOnto's usability and utility to improve requirements artifact by the use of formalized domain knowledge in DO. Requirements artifacts are revised according to the DO usage patterns and have an improved correctness, consistency, unambiguity and completeness. To our knowledge we are the first to have extracted usage strategies for DO. Through the use of TOREOnto, we have presented an example of how to describe the ontology usage systematically.

In Chapter 12, the insights into the adaptation of a generic RE-method are presented and discussed. From both types of method adaptations, i.e. for the system domain and the application domain, we deduced the principles that are necessary to adapt any kind of other generic RE-method. These principles are not validated empirically, but provide the essentials based on the experiences of DsTORE and TOREOnto. The activities for the method adaptation are structured according to Wieringa's design cycle [Wie14], and they guide researches in the adaptations of other RE-methods. Although both types of adaptations function differently, they share some common activities.

13.2 Future Work

The suggestions for future work are structured into concrete follow-ups related to DsTORE and TOREOnto, and more general and fundamental insights that might motivate other future lines of research. Our results are encouraging to continue with the following future work:

Regarding the system domain-specific adaptation DsTORE: The extent of DsTORE to support the specification of other types of DSSs should be further investigated. Other types of DSSs cover business intelligence or knowledge management-based DSSs, such as those which are presented in subsection 3.4.2 on page 42. The further adaptation of TORE to other system domains is interesting. In particular, we should understand in greater detail how other system domain-specifics shape new or extended DPs.

Regarding the application domain-specific adaptation TOREOnto: The presented literature review could be extended in some aspects. In order to complete the body of knowledge in the field of DO usage to improve requirements quality, we could extend the state of the art to the studies that are mentioned in Valaski et al. [VRM16]. They have additional primary studies, which should be investigated and related to the proposed DO usage patterns. More general *ontology usage patterns* should be extracted for other types of ontologies in order to understand their abilities for RE, and even in SE. Reasoning is the most powerful aspect among the DO usage patterns. So far it has not been evaluated how the structure of domain knowledge relates to reasoning for requirements defects. In particular, it is unknown which domain knowledge elements are necessary to find which requirements defects. This is directly related to the metamodel of the ontology. Therefore, the development of a conceptual model which describes the categories of knowledge elements and reasoning capabilities would be another interesting line of research to further explore. Since many approaches use more than one ontology, the interaction between such ontologies should be further investigated. The presented DO usage pattern support seven RQAs (see Table 10.1 on page 136). The ISO/IEC 29148 defines nine RQAs for InR and four RQAs for SRS. There is a necessity for research and further investigation to determine ontology usage scenarios that are able to improve these other RQAs.

The adaptation TOREOnto should additionally be applied to other domains. A further investigation of the DOs of other domains would also be interesting. How do they differ from the SNIK ontology? How are they structured in comparison to the main IM concepts? What is the impact of other DOs on the requirements quality? Do they provide the same level of requirements quality improvement as SNIK ontology for IM in hospitals? Furthermore, TOREOnto should be evaluated in another application domain, from the point of view of stakeholders and requirements engineers.

The domain knowledge has been applied manually in this thesis. The approaches that have been investigated provide several automatic and semi-automatic features to improve requirements quality. It would be interesting to apply these automatic features to the specification that was obtained with TORE. Moreover, are there any other ways to use a DO that are not discussed here? We did not focus on the DO usage to support other RE activities, such as elicitation or negotiation. Dermeval et al. [DVB$^+$15] showed support for such activities as well. It is interesting to extract usage pattern for those other activities. In a more general way, the domain knowledge could be used in other SE-phases and activities. For example, how could DOs support the software implementation or software-testing?

New opportunities for other research: Initially, the system type was considered as an integral part of the application domain [Bjø06]. For the method adaptation, a more differentiated view is helpful. Now that this diverse consideration is possible, a more formalized description of system types could be possible. There is even the possibility of creating a taxonomic view of system types that inputs to the method adaptations.

This thesis provides an example of a methodological approach of how to adapt a RE-method domain specifically. This could build the basis for future research that investigates other strategies of RE-method adaptation. Considering the experiences of researchers who deal with these other strategies will in turn help us to build a general approach for method adaptation.

Part V

Appendix

A Interview Questions and Questionnaires

A.1 Interview Questions for DsTORE Problem Investigation

To ensure an appropriate set of questions used in interviews of the problem investigation, we followed five goals:

G1	Create a set of detailed user task descriptions with existing problems for the CIO specifically.
G2	Determine existing deficits in currently used tools for the execution of user tasks.
G3	Understand the CIO's expectations regarding system functions and characteristics of CION.
G4	Understand the information and their relation necessary to execute the user tasks.
G5	Elicit existing data sources for system integration.

The following questions have been asked for each user task $\mathrm{project\ management}$ and $\mathrm{change\ management}$, except for question 10 which is not user task specific.

1. Which subtasks, data, systems and interfaces are related to the execution of the task <x> in the current state? (G1, as-is)
2. How do you imagine the future support of CION during the task <x>? Which activities and subtasks play a role during the task execution? Which data are important for the task execution? (G2, G3, G4, G5, to-be)
3. Which information, tools and computation or aggregation are you missing in the current tool-landscape for the execution of task <x>? (G2, as-is)
4. Which data shall be presented for task <x>? (G4, to-be)
5. Which system functions shall be provided by CION to edit data or information? (G3, to-be)
6. To what extent is configurability important for CION to interpret the data? (G3, to-be)
7. What are your expectations regarding the composition (manual or automatic) and presentation of these data? (G1, to-be)
8. Who provides you currently with the necessary data to execute task <x> and which

data are these? (G4, G5, as-is)

9. Who are you reporting to about the analysis results of CION to task <x> and in which form are they? (G3, G5, to-be)
10. In your opinion: Is CION a dashboard, a data warehouse or a tool to create data relations? (G3, to-be)

A.2 Interview Questions for DsTORE Treatment Validation

The goal of treatment validation is a better understanding of the decision subtasks of project management, change management and IT strategy. We used the extended subtask template as presented in Figure 6.3 on page 71 to fill in each attribute together with the CIO.

1. What is the decision-outcome?
2. What information is necessary to make the decision?
3. Which combined and computed data is necessary to make the decision?
4. What are possible rules for the decision-making?

A.3 Questionnaire for DsTORE Evaluation

This questionnaire is designed to evaluate the DsTORE method along with the modified artifacts to capture functional requirements.

A.3.1 First Phase of Interviews with the Tasks Project Management and Change Management

1. To what degree are you remembering the interviews on 12.10.2015, 21.12.2015, 20.1.2016, and 29.3.2016?

   ☐ very good ☐ good ☐ partially ☐ not at all

2. How did you feel the execution of the first interviews regarding:

- The identification of decisions

  free text

- the consideration of decision specific information

  free text

3. How well did you like the interviews altogether?

   ☐ very good ☐ good ☐ partially ☐ not at all

4. what did you like in the interviews overall good / less good?

   free text

5. Were information not mentioned in the interviews relevant to the decision?

   ☐ Yes, there were not mentioned information ☐ no If yes, which were these?

   free text

A.3.2 Second Phase of Interviews with the Tasks IT Strategy

5. To what degree are you remembering the interviews on 28.6.2016 and 29.6.2016?

   ☐ very good ☐ good ☐ partially ☐ not at all

6. How did you feel about conducting the first interviews regarding:

   - the identification of decisions

     free text

   - the consideration of decisional data

     free text

7. How well did you like the interviews altogether?

   ☐ very good ☐ good ☐ partially ☐ not at all

8. what did you like in the interviews overall good / less good?

   free text

9. Were there any decisional relevant information that have not been considered in the interviews?

☐ Yes there were not mentioned information ☐ no If yes, which were these?

free text

A.3.3 RE Artifacts

During the RE phase you were in contact with different RE artifacts. Please rate each of the following artifacts regarding *ease of use* and *importance*. The ease of use is the degree to which it was easy for you to understand and comment the artifact. The importance means how important the artifact is from your point of view to represent your requirements correctly.

A.3.3.1 Task Description with Task & Support

1. How easy was it for you to understand the task description according task & support?
   ☐ very easy ☐ easy ☐ difficult ☐ not at all
2. How easy was it for you to comment on the task description according task & support?
   ☐ very easy ☐ easy ☐ difficult ☐ not at all
3. How important is the task description according task & support to give your requirements?
   ☐ very important ☐ important ☐ less important ☐ not important at all
4. Did your estimate of the ease of use change between the first and second phase?
   ☐ yes ☐ no ☐ not sure If yes, how did your estimation change?
   free text
5. Did your estimate of the importance change between the first and second phase?
   ☐ yes ☐ no ☐ not sure If yes, how did your estimation change?
   free text

A.3.3.2 Classification of Subtasks

7. How easy was it for you to understand the classification of subtasks?
   ☐ very easy ☐ easy ☐ difficult ☐ not at all
8. How easy was it for you to comment on the subtasks (subtask type)?
   ☐ very easy ☐ easy ☐ difficult ☐ not at all

9. How important is the subtask classification according task & support to give your requirements?

   ☐ very important ☐ important ☐ less important ☐ not important at all

10. Do you miss any decision relevant information in this artifact?

    ☐ yes ☐ no If yes, which information is missing?

    free text

11. Did your estimate of the ease of use change between the first and second phase?

    ☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

    free text

12. Did your estimate of the importance change between the first and second phase?

    ☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

    free text

A.3.3.3 Decision Specific Subtask Description

13. How easy was it for you to understand the decision specific task description of subtasks?

    ☐ very easy ☐ easy ☐ difficult ☐ not at all

14. How easy was it for you to comment on the decision specific task description of subtasks?

    ☐ very easy ☐ easy ☐ difficult ☐ not at all

15. How important is the decision specific task description of subtasks to give your requirements?

    ☐ very important ☐ important ☐ less important ☐ not important at all

16. Do you miss any decision relevant information in this artifact?

    ☐ yes ☐ no If yes, which information is missing?

    free text

17. Did your estimate of the ease of use change between the first and second phase?

    ☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

    free text

18. Did your estimate of the importance change between the first and second phase?

☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

free text

A.3.3.4 Data Sources

19. How easy was it for you to understand the data source model?
    ☐ very easy ☐ easy ☐ difficult ☐ not at all
20. How easy was it for you to comment on the data source model?
    ☐ very easy ☐ easy ☐ difficult ☐ not at all
21. How important is the data source model to give your requirements?
    ☐ very important ☐ important ☐ less important ☐ not important at all
22. Do you miss any decision relevant information in this artifact?
    ☐ yes ☐ no If yes, which information is missing?

    free text
23. Did your estimate of the ease of use change between the first and second phase?
    ☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

    free text
24. Did your estimate of the importance change between the first and second phase?
    ☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

    free text

A.3.3.5 Workspaces

25. How easy was it for you to understand the data source model?
    ☐ very easy ☐ easy ☐ difficult ☐ not at all
26. How easy was it for you to comment on the data source model?
    ☐ very easy ☐ easy ☐ difficult ☐ not at all
27. How important is the data source model to give your requirements?
    ☐ very important ☐ important ☐ less important ☐ not important at all
28. Did your estimate of the ease of use change between the first and second phase?
    ☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

free text

29. Did your estimate of the importance change between the first and second phase?
☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

free text

A.3.3.6 Virtual Windows

30. How easy was it for you to understand the virtual windows?
☐ very easy ☐ easy ☐ difficult ☐ not at all

31. How easy was it for you to comment on the virtual windows?
☐ very easy ☐ easy ☐ difficult ☐ not at all

32. How important are the virtual windows to give your requirements?
☐ very important ☐ important ☐ less important ☐ not important at all

33. Did your estimate of the ease of use change between the first and second phase?
☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

free text

34. Did your estimate of the importance change between the first and second phase?
☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

free text

A.3.3.7 UI Prototype

35. How easy was it for you to understand the UI prototype?
☐ very easy ☐ easy ☐ difficult ☐ not at all

36. How easy was it for you to comment on the UI prototype?
☐ very easy ☐ easy ☐ difficult ☐ not at all

37. How important are the UI prototype to give your requirements?
☐ very important ☐ important ☐ less important ☐ not important at all

38. Did your estimate of the ease of use change between the first and second phase?
☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

free text

39. Did your estimate of the importance change between the first and second phase?

☐ yes ☐ no ☐ not sure If yes, how did your estimation change?

free text

Do you have any further comments?

free text

A.4 Questionnaire for System Prototype Evaluation

This questionnaire helped us to evaluate the existing software prototype of CION. We are interested in your opinion about the tested browser based CION.

1. What do you particularly like about the current software prototype?

   free text

2. What are you missing in the current software prototype?

   free text

3. How do you rate the user friendliness of the software prototype?

   ☐ very good ☐ good ☐ neither..nor ☐ suboptimal ☐ bad

   free text for reasons

4. How do you rate the utility of the software prototype (5 item Lickert scale, additional free text for reason).

   ☐ very useful ☐ useful ☐ neither..nor ☐ less useful ☐ hardly useful

   free text for reasons

5. How do you like the software prototype overall?

   ☐ very well ☐ well ☐ neither..nor ☐ partly ☐ not at all

   free text for reasons

6. To what extent are your expectations formed by the requirements documents (task descriptions, UI prototype) fulfilled by the software prototype?

   ☐ very well ☐ well ☐ neither..nor ☐ partially ☐ not at all

   free text for reasons

7. Would you use the software prototype in the future?

   ☐ in any case ☐ maybe ☐ not sure ☐ rather not ☐ under no circumstances

   free text for reasons

8. How would you change or improve the prototype? Any further comments?

   free text

A.5 Questionnaire for Resulting Activities Triggered by RE

Resulting activities in the CIO's division:

1. We are interested in the activities and ideas you initiated after the interviews: Which activities did you initiate to change or improve the situation in the information management department based on the interviews?

   free text

2. We are interested in changes on tasks and subtasks you initiated after the interviews:

   a) Which documents and spreadsheets were changed?

      free text

   b) Have tasks (also decisions) or subtasks been changed?

      free text

   c) Has the information needed to make decisions changed?

      free text

A.6 Questionnaire for the Evaluation of TOREOnto

The questionnaire for the *method evaluation* presented in subsection A.6.1 is filled out by the requirements engineer only, immediately after the evaluation has been performed. The questionnaire for the *artifact evaluation* presented in subsection A.6.2 is filled out by both, the domain experts and the requirements engineer twice: once for the original artifacts and once for the improved artifacts.

A.6.1 Questionnaire for Method Evaluation (Requirements engineer only)

QMeth.1) Is the effort to revise the requirements with the use of domain knowledge acceptable? (yes – neutral – no)

QMeth.2) Which difficulties did you have during the method execution?
– free text –

QMeth.3) How easy was it for you to apply the DO usage patterns to the DPs? (very easy – easy – difficult – very difficult)

QMeth.4) Which difficulties did you have during the access of domain knowledge?
– free text –

QMeth.5) How do you rate the access to domain knowledge during the improvement? (very easy – easy – difficult – very difficult)

A.6.2 Questionnaire for Artifact Evaluation

QArtif.1) Which conflicts to other requirements do you find?
– free text –

QArtif.2) How do you rate the severity of the conflicts found, and why?
(unproblematic – neutral – problematic) – reason for rating –

QArtif.3) Is the requirement sufficiently described? If not, which details do you miss in the description? (yes — no) – details of insufficient description —

QArtif.4) How do you rate the severity of missing details, and why?
(unproblematic – neutral – problematic) – reason for rating –

QArtif.5) Is there more than one way to interpret the requirement differently, if yes, which are these?
(yes - no) – free text, details to ways of interpretation. –

QArtif.6) How do you rate the severity of the possible different interpretations, and why?
(unproblematic – neutral – problematic) – reason for rating –

QArtif.7) How easy is it for you to understand the requirement?
(very easy – easy – difficult – very difficult)

QArtif.8) Are there any synonyms used, and if yes, which?
(yes – no), – free text, details of synonyms in requirements. –

QArtif.9) How do you rate the severity of the used synonyms, and why?
(unproblematic – neutral – problematic) – reason for rating –

QArtif.10) Is the requirement expressed correctly? If not, what are the details of the incorrectness
(yes - no) – free text, details of the incorrectness –

QArtif.11) How do you rate the severity of the incorrect requirement, and why?
(unproblematic – neutral – problematic) – reason for rating –

QArtif.12) For all artifacts except *system function*: How do you rate the improvement effect for the requirement, and why?
(greatly improved – slightly improved – not improved – slightly worsen – greatly worsen – not sure) – reason for rating –

QArtif.13) Do you have any further comments to the artifact or its quality?
– free text –

B TORE Artifact Types

	(T1) Task	(D1) As-Is	(D2) To-Be	(D3) System Respons.	(D4) Domain Data	(I1) System Function	(I2) Interaction	(I3) Interaction Data	(I4) UI-Structure	(C2) Int. System Actions	(C3) Internal Data	(C1) Architecture	(G1) Nav./Supp.Functions	(G2) Dialog	(G3) UI-Data	(G4) Screen Structure
Business Process Diagram	X															
Role Description	X															
Task Description	X	X														
Activity Diagram and Activity Description		X	X	X												
Use Case Diagram				X												
Function Description						X										
Use Case Text							X									
ERD					X											
Glossary					X											
Workspace Model									X							
Class Model								X		X	X					
Architecture Description												X				
Dialog Text														X		
Dialog Statechart														X		
Prototype													X		X	X

Figure B.1: Notation elements of the TORE Decision Points [PK04]

C SNIK Metamodel

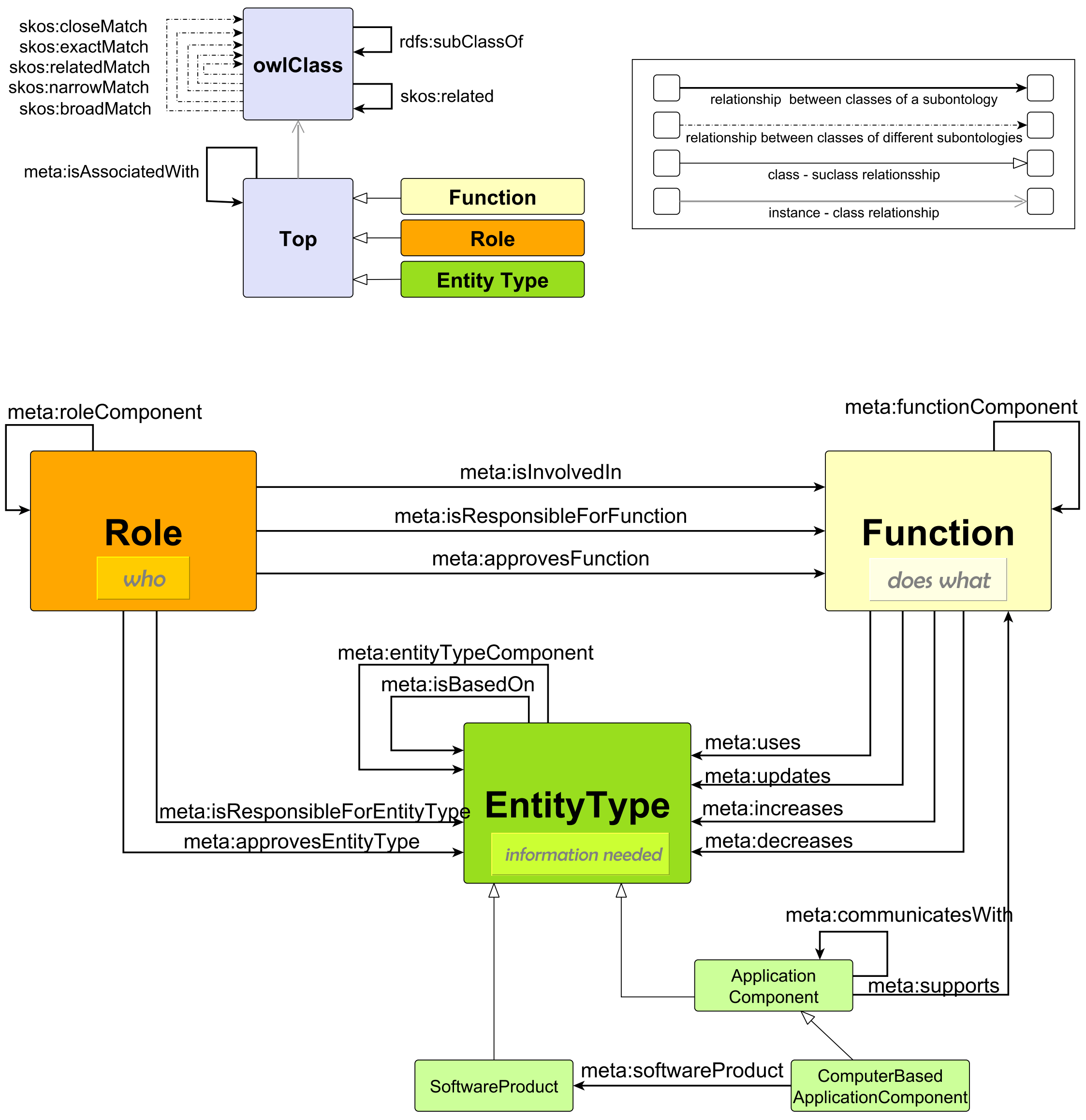


Figure C.1: The SNIK Ontology metamodel: relations and subclass hierarchy. (source: http://www.snik.eu/de/Ergebnisse/index.jsp)

D Artifacts of Tasks Project Management, Change Management, and IT strategy

D.1 The Task Project Management

D.1.1 Goal & Task level

Table D.1 shows the task description of CIO's task project management in Lauesen's Task & Support template. DsTORE's DP *Categorization of Subtasks* is denoted in 2nd column *subtask name* in brackets for reasons of space by, e.g. (prioritization).

D.1.1.1 Decision Point Stakeholders' Tasks

Table D.1: The task project management in Task & Support template

Task	Projektmanagement (Multiprojektmanagement bzw. Projektportfoliomanagement)
Purpose	Planung, Durchführung und Beaufsichtigung von Projekten
Source	Interview with CIO Mr. X, 28.5.2015, 12.10.2015
Responsible	Mr. X

ID	Subtask	As is solution	Example solution
1	Unterstützung der Erstellung von Projektdefinition aus der Change-Bewertung. (Support)	Der CIO wirkt in der Rolle als CAB-Mitglied bei der Erstellung einer Projektdefinition aus einer Change-Bewertung durch einen Mitarbeiter aus dem IM Bereich mit. Dazu gehört auch die Ressourcen-Einteilung im Projekt	—
2	Hinzufügen eines neues Projekts zur Projektwarteliste (Documentation)	Über verschiedene Kanäle wie z.B. E-Mail, Change-Anträge, Besprechungen, Vorstandsitzungen erreichen den CIO Anfragen zu neuen Projekten. Projekte werden von dem Leiter Projektmanagement und dem CIO zur Projektwarteliste hinzugefügt.	
3	Projektpriorisierung		
3a	Priorisierung der Projekte aus Projektwarteliste. **Ziel**: Identifikation der wichtigsten Projekte. Festlegung der Ausführungsreihenfolge, Budgetplanung. **Problem**: Die Planung der Reihenfolge ist aktuell nicht Werkzeuggestützt. Ressourcenmanagement findet aktuell nicht statt (technische Gründe). (Prioritization)	Die Freigabe eines Projekts wird implizit durch die Vergabe einer Priorität erteilt. Verwendung der Projektliste als Teilmenge der Projektwarteliste. Voraussetzung: Budget und Projektdefinition liegen vor. VK-Aufwand und Kosten wird für die Folgejahre unter Berücksichtigung des Teamplans und verfügbaren Budgets geplant. Pro Projekt wird das Risiko bewertet: 1(high): Betriebssicherung 2(medium): Notwendig für Betrieb 3(low): Wünschenswert für Betrieb oder Innovation. Die Umsetzung umfangreicher genehmigter Change-Anträge werden als Projekt realisiert. Wenn der Change dringlich ist, kommt das Projekt direkt in die Projektliste und muss ggf. mit einem Budget versehen werden.	Mit Hilfe des neuen CIO Navigators werden alle Informationen für Planung von Projekten dargestellt. Die dargestellten Daten ermöglichen eine einfache Festlegung der Reihenfolge von Projekten. Dazu zeigt eine Übersicht eine schönere und bessere Darstellung der Projektliste und Projektwarteliste, die über Sortiermöglichkeiten und Filtermöglichkeiten verfügt. Die Unterstützung von Ressourcenmanagement innerhalb der Projekte wäre hilfreich. Einsatz eines Werkzeuges wie MS Project wäre wünschenswert
3b	Vorstellung der Projektwarteliste als Halbjahresplanung beim Vorstand (Communication)	Die vorgefertigte Projektwarteliste als Halbjahresplanung wird dem Vorstand präsentiert. Als Vorbereitung werden die bisher noch nicht gestarteten Projekte aus der aktuellen Projektwarteliste mit einem Filter in eine neue Projektwarteliste übertragen. Die neue Projektwarteliste wird dem Vorstand präsentiert.	Die Erstellung des Berichts an den Vorstand in Form einer gefilterten Projektwarteliste kann in Verbindung mit der Entscheidung neue Projekte auf die Projektwarteliste zu nehmen, IT-gestützt erstellt werden.

3c	Repriorisierung der Projektwarteliste mit dem Vorstand (Prioritization)	Überarbeitung der Projektwarteliste mit Auswahl der zu bearbeitenden Projekte nach Votum des Vorstands, bis Vorstand Einverständnis gibt. Bei Notwendigkeit der Überarbeitung ergibt sich Schleife zu Subtask 3a. Auswahl der Projekte erfolgt durch IM Abteilung, nicht durch Vorstand.	CIONx stellt alle Informationen für die Repriorisierung der Projektwarteliste mit dem Vorstand auf einen Blick dar. Änderungen von Projekten können direkt durchgeführt werden. Alle beschlossenen Änderungen werden in einem Änderungsprotokoll dokumentiert, z.B. Priorisierungsänderungen und Budgetänderungen oder Umwidmungen zur nachträglichen Änderungen in den Quellsystemen.
3d	Diskutieren, Erhalten, Verhandeln von Projektfreigaben (Communication)	–	–
4	Einsetzen einer ProjektleiterIn in Projekt aus der Projektwarteliste. **Problem**: Übersicht zu Qualifikationen der MitarbeiterInnen fehlt, auch im SAP-HR. (Approval)	Sobald ein Projekt eine Priorität hat und damit kurz vor dem Start steht, benötigt es eineN ProjektleiterIn. DieseR ist für eine Budget- und Ressourcenplanung zuständig, die Voraussetzung für den Start des Projektes ist. Kenntnis der eingesetzten ProjektleiterInnen und Ressourcenverbrauch sowie Kapazität. Informationen über Erfahrung (hat PL ähnliche Projekte gemacht?) und welcher IT Bereich (Infrastruktur, Applikation, Organisation). Übersicht über PL gibt es bereits in der Projektliste.	Qualifikationen sollten Bestandteil der elektronischen Personalakte sein. (Wird nicht durch CION realisiert)
5	Projektmonitoring		
5a	Abfrage des Projektstatus aller Projekte 14 tägig im Jour Fixe der ProjektleiterInnen. **Ziel**: Rechtfertigung zum aktuellen Projektstand. **Problem**: Übersicht über aktuellen Projektstatus fehlt. (Monitoring)	Projektleiter Jour fixe zur Besprechung laufender Projekte (BereichsleiterIn, AbteilungsleiterIn, 1 MitarbeiterIn vom Vertragsmanagement, Projektleiter) Projektauslastung, Personalstärke und Ausfälle.	Der CIO hätte gerne als Gesprächsvorbereitung den aktuellen Projektstatus als Einschätzung desR ProjektleiterIns, damit das Gespräch effizienter durchgeführt werden kann. Status laufender Projekte als Teaser für die Projekt-Webseite.

5b	Bewertung des Status, Enddatum, Zieldatum und aktuellem inhaltlichen Stand. (Evaluation)	Entscheidung basiert auf Information von PL in Teamsitzung.	CION zeigt den Projektstatus von Webseite und relevante Projektinformationen aus dem Projektplan als Ort der Statusverletzung, den aktuellen Stand des Budgets und den/die ProjektauftraggeberIn als Adressat für Eskalation. Nach Transparenzgebot muss die Startseite des Projekts öffentlich sein, darf aber keine Budgetinformation und Status zeigen. Notwendigkeit der Konfigurierbarkeit für mehrere Adressaten. Die Darstellung eines Gantt-Diagramms mit einem datumsbezogenen Ausschnitt ist hilfreich. Darin sollen die laufenden Vorgänge des Projekts bezogen auf den Tag als Tabelle sichtbar sein. Das ganze Gantt Diagramm darzustellen geht aufgrund der Komplexität der Projekte nicht.
5c	Eskalation intern durch Bereichsleiter (Communication)	–	–
5d	Eskalation extern durch Auftraggeber (Communication)	–	–
5e	Durchführung Halbjahresreviews. **Ziel**: Erfassung des Erfüllungsgrad der Planung, Stand der Budgetplanung für Kommunikation zum Vorstand. (Monitoring)	Treffen auf Leitungsebene des IM Bereichs, insbesondere mit dem Abteilungsleiter Projektmanagement. Vergleich Ist/Sollstand vs. Projektplan. Betrachten des Erfüllungsstandes zusammen mit Projekt und Prozessmanagement.	–
5f	Evaluation des Halbjahresreviews (Evaluation)	Entscheidungen über weitere Projektstarts oder Budgetnachbesserungen auf Basis der Ergebnisse der Halbjahresreviews	Aufgabe wird unterstützt durch die generelle Projektübersicht

5g	Kommunikation Ergebnis der Halbjahresreviews an den Vorstand (Communication)	Kommunikation mit Hilfe von PowerPoint. Enthält inhaltlich extrahierten Projekt-Stand, Erfüllungsgrad der Planung. Stand der Budgetverwendung. Präsentation der Projekteliste mit Aussage zu Projektstand auf Basis der Ergebnisse aus den Halbjahresreviews. Keine Infos aus Projektliste. Kommunikation kann zu Entscheidungen seitens des Vorstands führen, z.B. Repriorisierung von Projekten.	–
5h	Evaluation des Kostenbudget/ Dienstleistungsbudget. **Problem**: Parallele Buchführung für Dienstleistungsbudget. Finanzcontrolling durch SAP IM, das offizielles Investmentmanagement ist. (Evaluation)	Budget schmilzt über die Projektlaufzeit ab. Assistenz des CIOs macht Teil des Projektcontrollings. Akteur: Mitarbeiter aus IM Bereich, Info an CIO. Ggf. Eskalation oder Umbuchung. Finanzielle Entscheidung über Bedarf, Budget für Projekt aufzufüllen, wenn mehr Budget verbraucht wurde als geplant, oder über Umwidmung wenn weniger Budget verbraucht wurde. Information: Controlling Liste mit Projektausgaben, auch in SAP IM, das offizielles Investitionsmanagement ist. Aufgabe von CIO: Abwägung welche Projekte unterstützt werden oder welche nicht.	Andere Aufbereitung der Budgets notwendig. Aktuelle Lösung auf Basis von Excel ausreichend.
6	Kommunikation der Projektergebnisse an Vorstand u. Auftraggeber (Communication)	Im Projekt: Lenkungsausschuss, im IM Bereich: Bereichsmeeting, gegenüber Auftraggebern: Vorstand	–
7	Monitoring der Transition Phase am Projektende (Monitoring)	Betrachten ob die Phase läuft. Informierungsaufgaben. Streuen in Gremien über die baldige Projektbeendigung, z.B. Vorstrand, klinischen Zentren und administrative Abteilungsleitungen während Jour Fixe. Bei großen Projekten Veröffentlichung in Mitarbeiterzeitung. Bei großen Projekten macht der CIO Präsentation in Leitungsgremien.	Aus jedem Projekt entsteht neuer oder veränderter IT Service. Die Verbindung gibt es noch nicht. Wunsch: Servicebeschreibung mit Parametern, Dokumentation. Service Katalog. Verfolgbarkeit zwischen Projekt, betroffenen Services und Change Requests sieht der CIO nicht als notwendig an.

D.1.2 Domain Level

This section contains the domain data model and the decision specific subtasks of task project management in more detail.

D.1.2.1 Decision Point: To-be decision specific subtask

In the following the decision specific subtasks of the CIO are shown. Descriptions of conventional subtasks that are no decision (also shown in Table D.1) are omitted as they do not contribute the goal of this thesis. The decision specific subtask project management 3a is shown in Figure 6.3 on page 71.

ID: Subtask	PM 3c: Re-prioritization of `project waiting list` with board of directors		
Category	Prioritization		
Actor	CIO	**Supporting Actors:** Board of directors, IM department	
Contribution	Board of directors take note of the yearly planning of the IM department.		
Cause	Yearly planning is finished		
Description	The result of the yearly planning is presented to the hospital board of directors. The CIO considers objections and different ratings of important projects.		
Pre condition	Project planning is finished		
Combined Data	**Entity**	**Attribute**	
	Project List Table Entry	project.priority	
	Project List Table Entry	*	
Computed Data	**Entity**	**Attribute**	**Computation**
	Project	Estimation of invest budget	
	Project	Estimation of service budget	
Post condition	Finalized project waiting list		
Info out	**Entity**	**Attribute**	
	Project List Table Entry	project.priority	
Rules	–		

Figure D.1: Decision subtask project management 3c: Re-prioritization of `project waiting list` with board of directors

ID: Subtask	PM 4: Nominate a project leader for a project	
Category	Approval	
Actor	CIO	**Supporting Actors:** –
Contribution		
Cause	Project has priority and attempts to start	
Description	A project leader needs to be nominated with an appropriate experience to the projects focus.	
Pre condition	The project priority was assigned and the project can start soon	
Combined Data	**Entity**	**Attribute**
	Project List	Projektleitungs-Erfahrung
	Project Definition	proposed project leader
	Project Definition	sub-division (infrastructure, application, organization)
Computed Data	–	
Post condition	Project can start.	
Info out	**Entity**	**Attribute**
	Project	project leader
Rules	Project leader should be associated to the sub-division where the project will be executed. Project leader should have a sound background and experience.	

Figure D.2: Decision subtask project management 4: Nominate a project leader for a project

ID: Subtask	PM 5b: Evaluation of project status, end date, target date and project progress		
Category	Evaluation		
Actor	CIO | **Supporting Actors:** project leader		
Contribution	Gain overview of ongoing projects status		
Cause	Biannual project status update		
Description	Gain overview of project status and project progress. Identification of projects that require attention, support or discussion		
Pre condition	Project is ongoing		
Combined Data	**Entity**	**Attribute**	
	Project	status	
	Project Specific Controlling List Service	remaining budget	
	Controlling List Invest	remaining budget	
	Project Specific Controlling List Service	budget on beginning of year	
	Overview Controlling List Invest	project.budget on beginning of the year	
	Project	time scale	
	Project Definition	milestones	
	Project	responsible sponsor	
Computed Data	**Entity**	**Attribute**	**Computation**
	Project Specific Controlling List Table Entry	budget.remaining	budget on beginning of year - sum(expense.contract volume)
	Project Specific Controlling List Invest Table Entry	budget.remaining	budget on beginning of year - sum(expense.contract volume)
Post condition	Possible escalation with actions in IM division, escalation to board of directors or other divisions		
Info out	**Entity**	**Attribute**	
	project	status	
Rules	If project status is not in plan anymore, then the project will be escalated		

Figure D.3: Decision subtask project management 5b: Evaluation of project status, end date, target date and project progress

ID: Subtask	PM 5f: Evaluation of project's half-year review	
Category	Evaluation	
Actor	CIO | **Supporting Actors:** hospital board of directors, project management department manager	
Contribution	Decision about measures from the half-year review	
Cause	Half-year period is over.	
Description	Each project is evaluated in a half years period	
Pre condition	Project is still ongoing	
Combined Data	**Entity**	**Attribute**
	Project Definition	problem description
	Project Definition	milestones
	Project	status
	Project leader	reasons for problems
Computed Data	–	
Post condition	Defined order, assigned priority of new projects. Budget is transferred to new project.	
Info out	**Entity**	**Attribute**
	Project	status
	Project Specific Controlling List Service	budget transfers
	Project Specific Controlling List Invest	budget transfers
	Project	priority
Rules		

Figure D.4: Decision subtask project management 5f: Evaluation of project's half-year review

ID: Subtask	PM 5h: Evaluation of project budget		
Category	Evaluation		
Actor	CIO | **Supporting Actors:** —		
Contribution	Assign unused budget to other projects.		
Cause	Periodical evaluation		
Description	Reassigning budget to other projects		
Pre condition	Project controlling is up to date		
Combined Data	**Entity**	**Attribute**	
	Project Specific Controlling List Service	budget on beginning of the year	
	Project Specific Controlling List Invest	budget on beginning of the year	
Computed Data	**Entity**	**Attribute**	**Computation**
	Project Summary Controlling List Service Table Entry	remaining budget	budget on beginning of the year - sum(expense.contract volume)
	Project Summary Controlling List Invest Table Entry	remaining budget	budget on beginning of the year - sum(expense.contract volume)
	Project Specific Controlling List Service Table Entry	contract volume	sum(expense.contract volume)
	Project Specific Controlling List Invest Table Entry	contract volume	sum(expense.contract volume)
Post condition	Budget adjustment.		
Info out	**Entity**	**Attribute**	
	Project Specific Controlling List Service	budget transfers	
	Project Specific Controlling List Invest	budget transfers	
	Service Budget	available budget	
	Invest Budget	available budget	
Rules	Increase budget for a project if it requires more; Re-assign budget from other projects if a project has required less budget as planned.		

Figure D.5: Decision subtask project management 5h: Evaluation of project budget

D.1.2.2 Decision Point: Domain Data

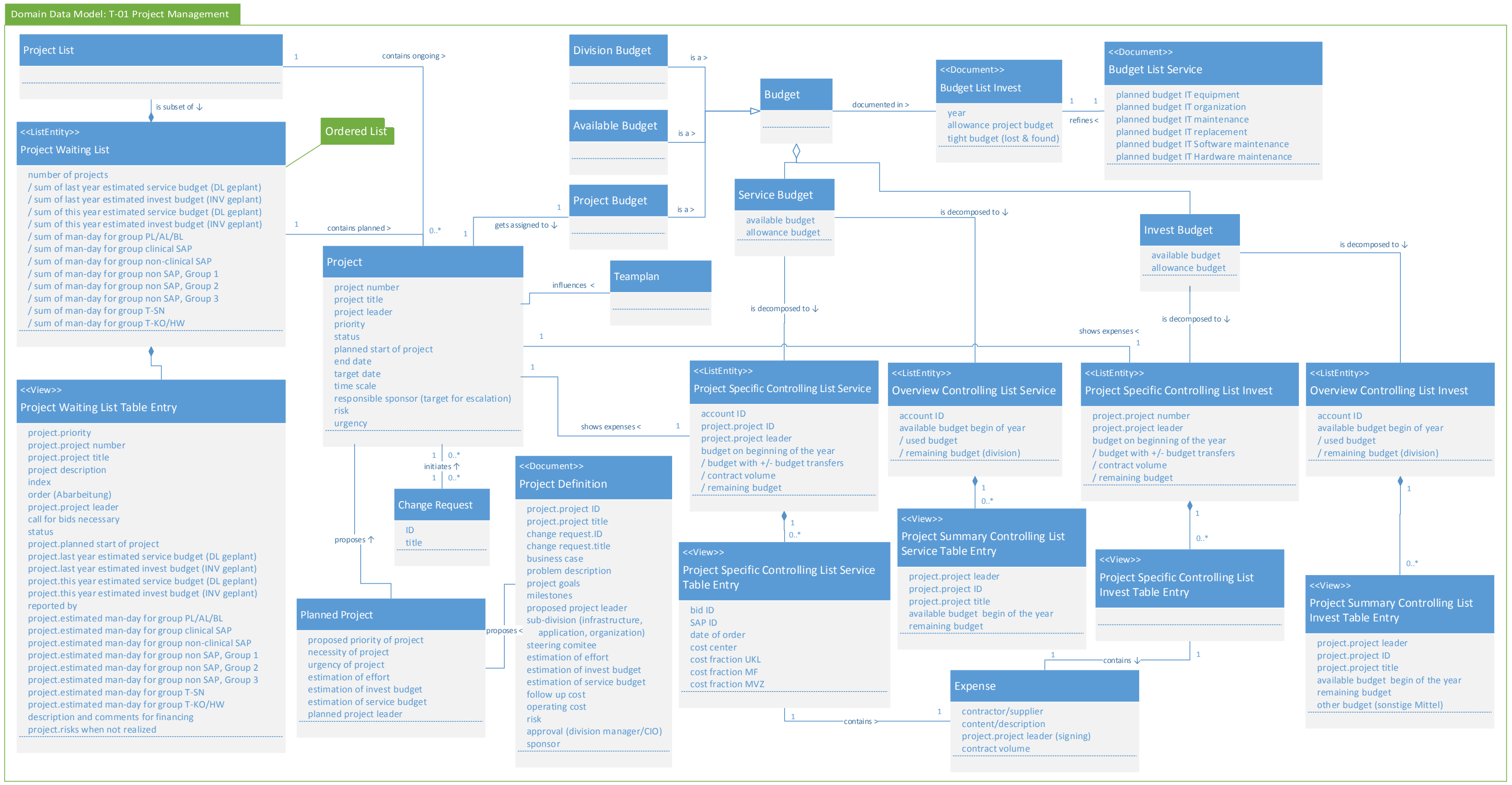

Domain Data Model: T-01 Project Management
Project List
contains ongoing >
is subset of ↓
<<ListEntity>>
Project Waiting List
number of projects
/ sum of last year estimated service budget (DL geplant)
/ sum of last year estimated invest budget (INV geplant)
/ sum of this year estimated service budget (DL geplant)
/ sum of this year estimated invest budget (INV geplant)
/ sum of man-day for group PL/AL/BL
/ sum of man-day for group clinical SAP
/ sum of man-day for group non-clinical SAP
/ sum of man-day for group non SAP, Group 1
/ sum of man-day for group non SAP, Group 2
/ sum of man-day for group non SAP, Group 3
/ sum of man-day for group T-SN
/ sum of man-day for group T-KO/HW
Ordered List
contains planned >
<<View>>
Project Waiting List Table Entry
project.priority
project.project number
project.project title
project description
index
order (Abarbeitung)
project.project leader
call for bids necessary
status
project.planned start of project
project.last year estimated service budget (DL geplant)
project.last year estimated invest budget (INV geplant)
project.this year estimated service budget (DL geplant)
project.this year estimated invest budget (INV geplant)
reported by
project.estimated man-day for group PL/AL/BL
project.estimated man-day for group clinical SAP
project.estimated man-day for group non-clinical SAP
project.estimated man-day for group non SAP, Group 1
project.estimated man-day for group non SAP, Group 2
project.estimated man-day for group non SAP, Group 3
project.estimated man-day for group T-SN
project.estimated man-day for group T-KO/HW
description and comments for financing
project.risks when not realized
Division Budget
Available Budget
Project Budget
is a >
gets assigned to ↓
Project
project number
project title
project leader
priority
status
planned start of project
end date
target date
time scale
responsible sponsor (target for escalation)
risk
urgency
influences <
Teamplan
initiates ↑
Change Request
ID
title
proposes ↑
Planned Project
proposed priority of project
necessity of project
urgency of project
estimation of effort
estimation of invest budget
estimation of service budget
planned project leader
proposes <
<<Document>>
Project Definition
project.project ID
project.project title
change request.ID
change request.title
business case
problem description
project goals
milestones
proposed project leader
sub-division (infrastructure, application, organization)
steering comitee
estimation of effort
estimation of invest budget
estimation of service budget
follow up cost
operating cost
risk
approval (division manager/CIO)
sponsor
shows expenses <
Budget
documented in >
<<Document>>
Budget List Invest
year
allowance project budget
tight budget (lost & found)
refines <
<<Document>>
Budget List Service
planned budget IT equipment
planned budget IT organization
planned budget IT maintenance
planned budget IT replacement
planned budget IT Software maintenance
planned budget IT Hardware maintenance
Service Budget
available budget
allowance budget
is decomposed to ↓
Invest Budget
available budget
allowance budget
shows expenses <
<<ListEntity>>
Project Specific Controlling List Service
account ID
project.project ID
project.project leader
budget on beginning of the year
/ budget with +/- budget transfers
/ contract volume
/ remaining budget
<<ListEntity>>
Overview Controlling List Service
account ID
available budget begin of year
/ used budget
/ remaining budget (division)
<<ListEntity>>
Project Specific Controlling List Invest
project.project number
project.project leader
budget on beginning of the year
/ budget with +/- budget transfers
/ contract volume
/ remaining budget
<<ListEntity>>
Overview Controlling List Invest
account ID
available budget begin of year
/ used budget
/ remaining budget (division)
<<View>>
Project Specific Controlling List Service Table Entry
bid ID
SAP ID
date of order
cost center
cost fraction UKL
cost fraction MF
cost fraction MVZ
<<View>>
Project Summary Controlling List Service Table Entry
project.project leader
project.project ID
project.project title
available budget begin of the year
remaining budget
<<View>>
Project Specific Controlling List Invest Table Entry
<<View>>
Project Summary Controlling List Invest Table Entry
project.project leader
project.project ID
project.project title
available budget begin of the year
remaining budget
other budget (sonstige Mittel)
contains ↓
contains >
Expense
contractor/supplier
content/description
project.project leader (signing)
contract volume
1
0..*

Figure D.6: Part of the domain data model for the task project management

D.1.3 Interaction Level

D.1.3.1 Decision Point: UI Structure

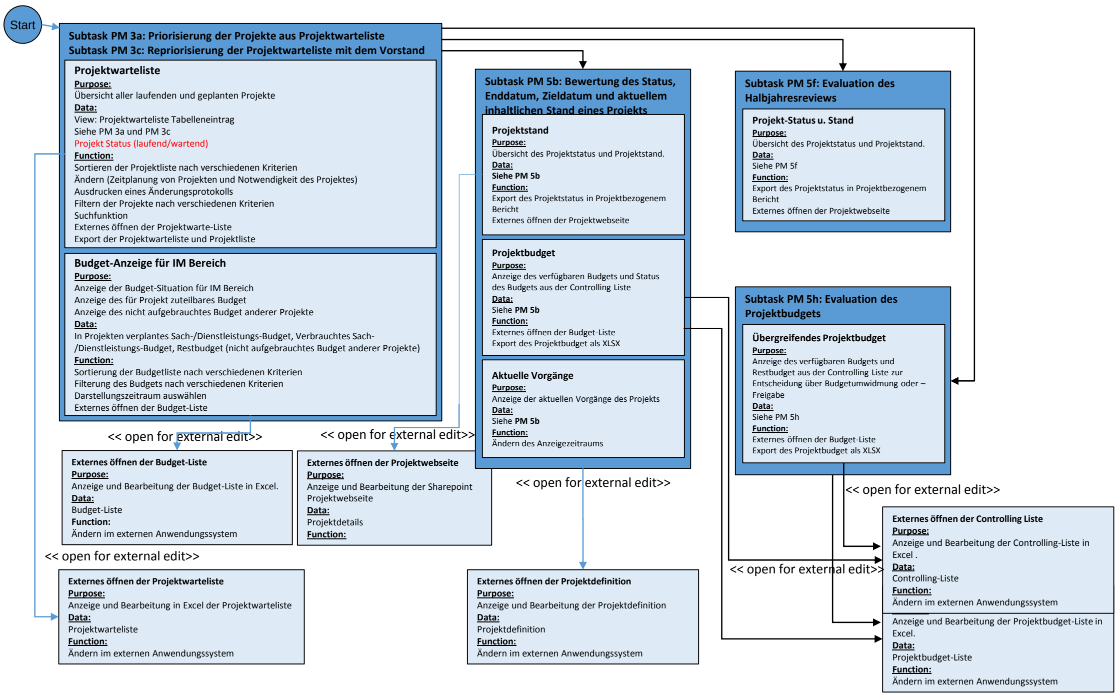
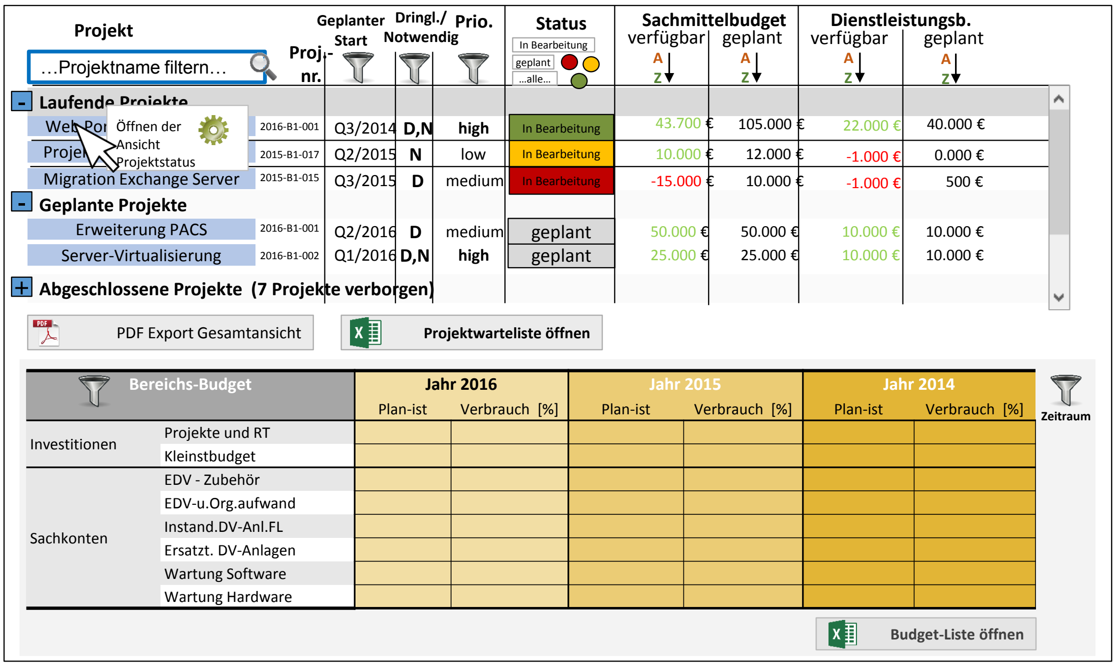

Start
Subtask PM 3a: Priorisierung der Projekte aus Projektwarteliste
Subtask PM 3c: Repriorisierung der Projektwarteliste mit dem Vorstand
Projektwarteliste
Purpose:
Übersicht aller laufenden und geplanten Projekte
Data:
View: Projektwarteliste Tabelleneintrag
Siehe PM 3a und PM 3c
Projekt Status (laufend/wartend)
Function:
Sortieren der Projektliste nach verschiedenen Kriterien
Ändern (Zeitplanung von Projekten und Notwendigkeit des Projektes)
Ausdrucken eines Änderungsprotokolls
Filtern der Projekte nach verschiedenen Kriterien
Suchfunktion
Externes öffnen der Projektwarte-Liste
Export der Projektwarteliste und Projektliste
Budget-Anzeige für IM Bereich
Purpose:
Anzeige der Budget-Situation für IM Bereich
Anzeige des für Projekt zuteilbares Budget
Anzeige des nicht aufgebrauchtes Budget anderer Projekte
Data:
In Projekten verplantes Sach-/Dienstleistungs-Budget, Verbrauchtes Sach-/Dienstleistungs-Budget, Restbudget (nicht aufgebrauchtes Budget anderer Projekte)
Function:
Sortierung der Budgetliste nach verschiedenen Kriterien
Filterung des Budgets nach verschiedenen Kriterien
Darstellungszeitraum auswählen
Externes öffnen der Budget-Liste
Subtask PM 5b: Bewertung des Status, Enddatum, Zieldatum und aktuellem inhaltlichen Stand eines Projekts
Projektstand
Purpose:
Übersicht des Projektstatus und Projektstand.
Data:
Siehe PM 5b
Function:
Export des Projektstatus in Projektbezogenem Bericht
Externes öffnen der Projektwebseite
Projektbudget
Purpose:
Anzeige des verfügbaren Budgets und Status des Budgets aus der Controlling Liste
Data:
Siehe PM 5b
Function:
Externes öffnen der Budget-Liste
Export des Projektbudget als XLSX
Aktuelle Vorgänge
Purpose:
Anzeige der aktuellen Vorgänge des Projekts
Data:
Siehe PM 5b
Function:
Ändern des Anzeigezeitraums
Subtask PM 5f: Evaluation des Halbjahresreviews
Projekt-Status u. Stand
Purpose:
Übersicht des Projektstatus und Projektstand.
Data:
Siehe PM 5f
Function:
Export des Projektstatus in Projektbezogenem Bericht
Externes öffnen der Projektwebseite
Subtask PM 5h: Evaluation des Projektbudgets
Übergreifendes Projektbudget
Purpose:
Anzeige des verfügbaren Budgets und Restbudget aus der Controlling Liste zur Entscheidung über Budgetumwidmung oder –Freigabe
Data:
Siehe PM 5h
Function:
Externes öffnen der Budget-Liste
Export des Projektbudget als XLSX
<< open for external edit>>
Externes öffnen der Budget-Liste
Purpose:
Anzeige und Bearbeitung der Budget-Liste in Excel.
Data:
Budget-Liste
Function:
Ändern im externen Anwendungssystem
<< open for external edit>>
Externes öffnen der Projektwebseite
Purpose:
Anzeige und Bearbeitung der Sharepoint Projektwebseite
Data:
Projektdetails
Function:
<< open for external edit>>
<< open for external edit>>
Externes öffnen der Projektwarteliste
Purpose:
Anzeige und Bearbeitung in Excel der Projektwarteliste
Data:
Projektwarteliste
Function:
Ändern im externen Anwendungssystem
Externes öffnen der Projektdefinition
Purpose:
Anzeige und Bearbeitung der Projektdefinition
Data:
Projektdefinition
Function:
Ändern im externen Anwendungssystem
<< open for external edit>>
<< open for external edit>>
Externes öffnen der Controlling Liste
Purpose:
Anzeige und Bearbeitung der Controlling-Liste in Excel .
Data:
Controlling-Liste
Function:
Ändern im externen Anwendungssystem
Anzeige und Bearbeitung der Projektbudget-Liste in Excel.
Data:
Projektbudget-Liste
Function:
Ändern im externen Anwendungssystem

Figure D.7: The workspace model of task project management



Projekt	Proj.-nr.	Geplanter Start	Dringl./ Notwendig	Prio.	Status	Sachmittelbudget verfügbar	Sachmittelbudget geplant	Dienstleistungsb. verfügbar	Dienstleistungsb. geplant
Laufende Projekte									
Web Por	2016-B1-001	Q3/2014	**D,N**	**high**	In Bearbeitung	43.700 €	105.000 €	22.000 €	40.000 €
Proje	2015-B1-017	Q2/2015	**N**	low	In Bearbeitung	10.000 €	12.000 €	-1.000 €	0.000 €
Migration Exchange Server	2015-B1-015	Q3/2015	**D**	medium	In Bearbeitung	-15.000 €	10.000 €	-1.000 €	500 €
Geplante Projekte									
Erweiterung PACS	2016-B1-001	Q2/2016	**D**	medium	geplant	50.000 €	50.000 €	10.000 €	10.000 €
Server-Virtualisierung	2016-B1-002	Q1/2016	**D,N**	**high**	geplant	25.000 €	25.000 €	10.000 €	10.000 €
Abgeschlossene Projekte (7 Projekte verborgen)									

PDF Export Gesamtansicht

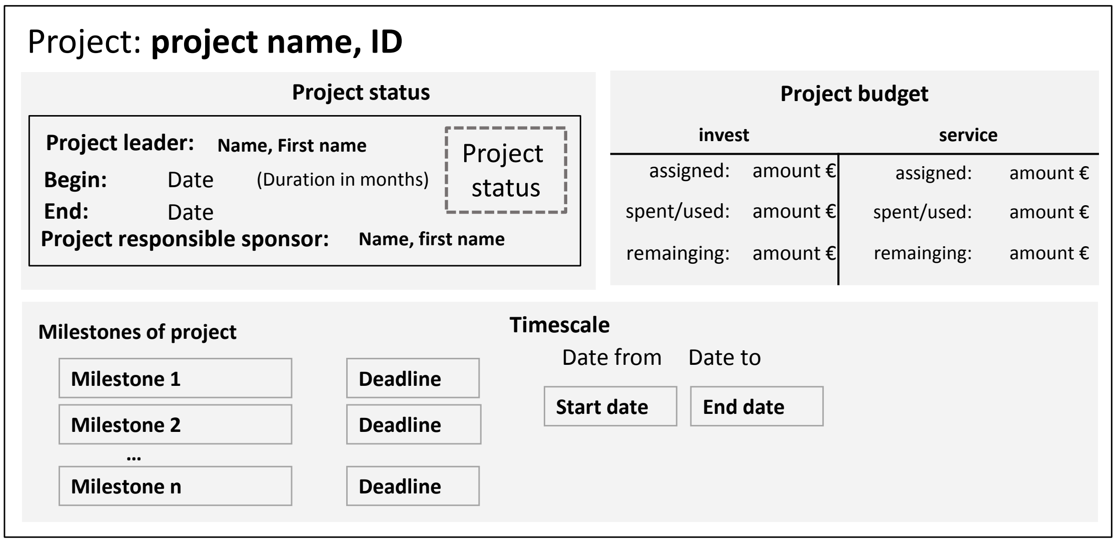
Projektwarteliste öffnen

Bereichs-Budget		**Jahr 2016**		**Jahr 2015**		**Jahr 2014**	
		Plan-ist	Verbrauch [%]	Plan-ist	Verbrauch [%]	Plan-ist	Verbrauch [%]
Investitionen	Projekte und RT						
	Kleinstbudget						
Sachkonten	EDV - Zubehör						
	EDV-u.Org.aufwand						
	Instand.DV-Anl.FL						
	Ersatzt. DV-Anlagen						
	Wartung Software						
	Wartung Hardware						

Zeitraum

Budget-Liste öffnen

Figure D.8: The UI prototype for the subtasks project management 3a, 3c, 5h



Project: **project name, ID**

Project status

Project leader: **Name, First name**
Begin: Date (Duration in months)
End: Date
Project responsible sponsor: **Name, first name**

Project status

Project budget

	invest		**service**
assigned:	amount €	assigned:	amount €
spent/used:	amount €	spent/used:	amount €
remainging:	amount €	remainging:	amount €

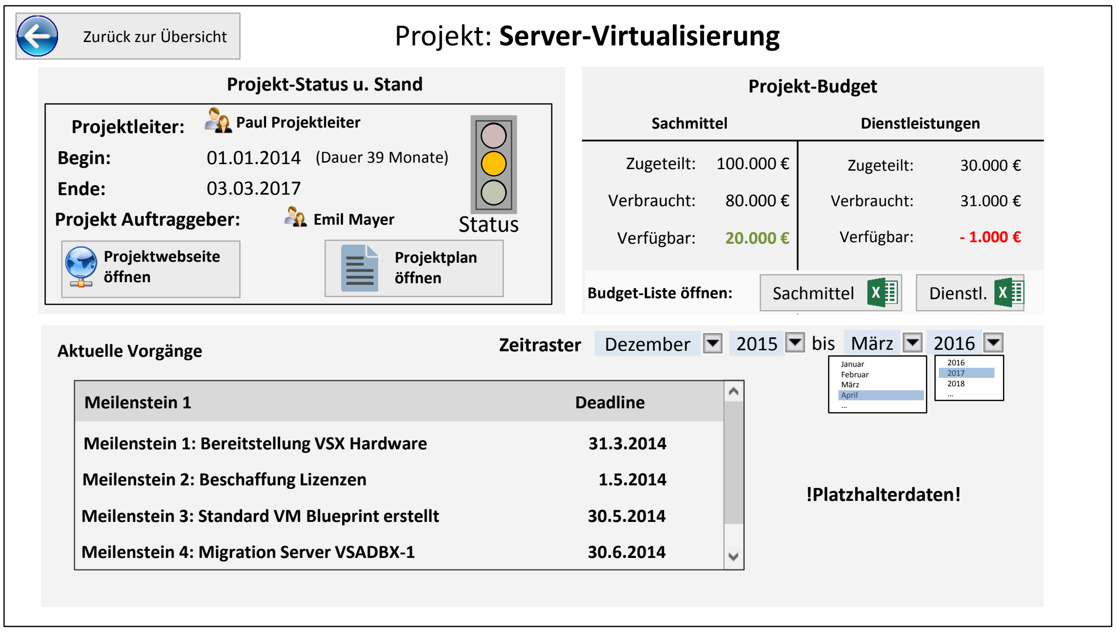
Milestones of project

Milestone 1	**Deadline**
Milestone 2	**Deadline**
...	
Milestone n	**Deadline**

Timescale

Date from	Date to
Start date	**End date**

Figure D.9: The virtual window for project details as used in subtasks project management 5b, 5f



Zurück zur Übersicht

Projekt: **Server-Virtualisierung**

Projekt-Status u. Stand

Projektleiter: **Paul Projektleiter**
Begin: 01.01.2014 (Dauer 39 Monate)
Ende: 03.03.2017
Projekt Auftraggeber: **Emil Mayer**

Status

Projektwebseite öffnen **Projektplan öffnen**

Projekt-Budget

	Sachmittel		**Dienstleistungen**
Zugeteilt:	100.000 €	Zugeteilt:	30.000 €
Verbraucht:	80.000 €	Verbraucht:	31.000 €
Verfügbar:	**20.000 €**	Verfügbar:	**- 1.000 €**

Budget-Liste öffnen: Sachmittel Dienstl.

Aktuelle Vorgänge

Zeitraster Dezember 2015 bis März 2016

Meilenstein 1	**Deadline**
Meilenstein 1: Bereitstellung VSX Hardware	**31.3.2014**
Meilenstein 2: Beschaffung Lizenzen	**1.5.2014**
Meilenstein 3: Standard VM Blueprint erstellt	**30.5.2014**
Meilenstein 4: Migration Server VSADBX-1	**30.6.2014**

!Platzhalterdaten!

Figure D.10: The UI prototype for the subtasks project management 5b, 5f

D.2 The Task Change Management

D.2.1 Goal & Task level

D.2.1.1 Decision Point Stakeholders' Tasks

Table D.2: The task *change management* in Task & Support template

Task	Change Management
Purpose	Planung und Durchführung von Änderungen an IT Systemen
Source	Interview with CIO Mr. X, 28.5.2015, 13.10.2015
Responsible	CIO

ID	**Subtask**	**As is solution**	**Example solution**
1	Erfassung Change Antrag (Documentation)	Change Antrag per Hauspost (wg. notwendigen Unterschriften) in den IM Bereich. Selten kommen Changeanträge als eingescannte PDFs per E-Mail. Papierbasierte Changes werden manuell in den SharePoint gestellt. Assistenz des CIOs ist Mitglied im CAB und ist dafür verantwortlich	–
2	Prüfung der formalen Vollständigkeit des Changes (Evaluation)	Rückweisung bei Unvollständigkeit (per E-Mail durch Assistenz des CIOs).	Die neue Ansicht im CIO-Navigator zeigt den Change-Antrag an und markiert fehlende Information.
3	Besprechung neuer Change-Anträge (Communication)	Regelmäßiger Termin mit CAB bestehend aus: Bereichsleitung, Abteilungsleitung (AL) Projektmanagement, Bereichsassistentin. Bei Bedarf betroffene AL (in der Regel Applikation, System, selten Service) und weitere Bearbeiter.	Besprechung neuer Changes. Entscheidung Bewertung oder sofortige Ablehnung. Bei Besprechung laufender Change-Anträge („In Prüfung“) auch mehr Bearbeiter ins CAB (Eskalation, Status,. . .)
4	Einholen Change Bewertung		
4a	Aufforderung zur Change-bewertung an Mitarbeiter des IM Bereichs (Communication)	Beurteilung (Entscheidung) über Sinnhaftigkeit eines Change Antrags Ziel der Change-Bewertung: Abschätzung und Kalkulation der Kosten sowie Identifikationen der Abhängigkeiten durch Mitarbeiter des IM Bereichs. Assistenz des CIOs schickt Delegation an Mitarbeiter mit Link auf PDF des Change Antrags im Sharepoint und Link zu Template für Change Bewertung in Word) Evtl. Einladung für nächste CAB Sitzung.	–
4b	Aufforderung der ALs zur Change Bewertung(Communication)	Alle Abteilungsleiter müssen eine Change-Bewertung abgeben und werden formal dazu aufgefordert.	–

4c	Aufforderung der ALs zur Abgabe von Statements zu Change Bewertung (Communication)	Nach Erstellung der Change-Bewertung (ChB) sind die Abteilungsleiter aufgefordert, binnen 1 Woche ein Statement abzugeben	–
4d	Statement zu Change Bewertung erstellen (Documentation)	Nach Abgabe der Statements zu einer Change-Bewertung (ChB) durch die Abteilungsleiter ist der CIO aufgefordert, binnen 1 Woche ein Statement abzugeben.	–
4e	Prüfen des Vorhandenseins der Change Bewertung und der AL-Statements (Evaluation)	Entscheidung über Wiedervorlage, wenn Statements oder Change Bewertungen nicht vorliegen	–
5	Bearbeiten Change Bewertung		
5a	Evaluierung der Change Bewertung. **Problem**: Es existieren Medienbrüche durch die fehlende Integration zwischen Service Management und SharePoint. Ausdruck von Changes durch Assistenz des CIOs. Entscheidungen des CAB werden auf Papier notiert und nachträglich in SharePoint erfasst. Lösung für CAB Sitzungen jedoch angemessen und schnell. (Evaluation)	Ergebnisse der Evaluierung der Change-Bewertung: 1. Ablehnung (5b) 2. Überarbeitung durch Antragsteller (5c) 3. operativ zu lösen (5d) 4. Projektwarteliste (5e) 5. aktuelles Projekt (5f). Die Dokumentation der Entscheidung findet am Change-Antrag in SharePoint statt. Falls Änderung sofort umgesetzt werden muss, findet eine Re-Priorisierung der Projekte und damit eine Änderung der Projektliste statt. Aus der Change-Bewertung kann sich die genehmigung des Change-Antrag ergeben. Wenn das Thema groß ist, wird es als Projekt realisiert. Wenn es darüber hinaus noch dringlich ist, kommt es unmittelbar in die Projektliste. Dann muss das neue Projekt ggf. mit einem Budget versehen werden (siehe Subtasks Project Management 4).	Eine neue Ansicht in CION zeigt alle zu einem Change-Antrag vorhandenen Change-Bewertungen und Statements an. Evtl. notwendige Projektinformation werden aus der Projekt-Übersicht mit Budget-Information entnommen. Anzeige des Bereichsbudgets mit Darstellung des Restbudgets des laufenden Jahres und allokierte Kosten der Folgejahre von heute+4 Jahre aus der Controllingliste. Die Berechnung des Restbudgets läuft über die die Addition der verfügbaren Sach- und Dienstleistungs-Budgets aus den Subtasks PM 5b, 5h. Die Anzeige des Bereichsbudgets mit allokierten Kosten der Folgejahre ermöglicht die Entscheidung über verschieben von Changes in das nächstes Jahr.

5b	Dokumentation der Ablehnung des Changes (Documentation)	Dokumentation mit Begründung der Ablehnung.	
5c	Beauftragen der Überarbeitung des Changes durch Antragsteller (Communication)	Falls noch Informationen zum Change Antrag fehlen oder unklare Aufgabenstellung.	
5d	Übergabe an Abteilungsleiter zur Bearbeitung des Changes im Operativen Betrieb. **Problem**: Bei in den operativen Bereich delegierten Changes besteht die große Gefahr des Untergehens. SCSM kann grundsätzlich noch nicht mit Changes umgehen Monitoring durch CAB mit Hilfe von Deadlines. (Communication)	Die Übergabe eines Changes an einen an Abteilungsleiter zur Bearbeitung des Changes im Operativen Betrieb wird in der Dokumentation des CAB zu den Change-Anträgen vermerkt und mit einer Deadline für die Wiedervorlage versehen. Assistenz des CIOs erstellt daraus manuell Wiedervorlagen für das CAB, um sicher zu stellen, dass der Change auch wirklich bearbeitet wird. Das CAB fungiert als Controllinginstanz für die operative Bearbeitung des Changes (Siehe Aufgabe Change Management 6).	Controlling der Abarbeitung von Changes im operativen Bereich. Changes müssten als ServiceRequests oder technische Changes im SCSM weitergeführt werden.
5e	Initiieren des Changes als Projekt (Documentation)	Dokumentation in noch unverbindliche Projektwarteliste für Folgejahr. Wenn der Change als Projekt durchgeführt werden soll, wird Subtask Projektmanagement 2 ausgeführt.	
5f	Beauftragen der Änderung in bestehendem Projekt (Communication)	Der Change wird in bestehende Projekte mit aufgenommen und dort gelöst.	–
5g	Freigabe und Korrektur des Mitteilungstextes an Antragsteller über die Entscheidung des Change-Antrags. (Approval)	Assistenz des CIOs erstellt einen Mitteilungstext zur Kommunikation der Entscheidung über einen Change-Antrag an den Antragsteller. Die Mitgleider des CABs bekommen diesen Text zur Kenntnis, ggf. Korrektur und Freigabe. Die Kommunikation übernimmt Assistenz des CIOs	Der Mitteilungstext wird von CION als Template angezeigt und mit den Informationen über den Urheber und Titel/Bezeichnung des Change Antrages versehen. In CION hat der CIO die Möglichkeit, den Text per E-Mail zu versenden.

6	Monitoring der operativen Abarbeitung von Changes. **Problem:** Fehlende einheitliche Betrachtung von Change/Service Request/Projekt. **Problem:** Einige Changes mit geringem Umfang werden direkt in den Operativen Bereich delegiert. Aber es gibt kein Tracking/Monitoring über deren Status der Umsetzung im Rahmen von Wartungsfenstern. Es fehlt der Bezug zwischen Changemanagement und Operativem Management zur Identifikation des Status der Umsetzung solcher kleinen Changes. (Monitoring)	Terminmanagement. Changes werden mit Status „Operativ“ alle 8 Wochen betrachtet. Der Bezug zwischen Projekt und Wartungsfenster gibt es. Dieser wird jedoch manuell Erfasst. Dadurch existiert auch kein offensichtlicher Bezug der genutzt werden würde. Frage der Durchgängigkeit und durchgehender Dokumentation.	–

D.2.2 Domain Level

D.2.2.1 Decision Point: To-be decision specific subtask

ID: Subtask	CM 2: Evaluation of change request completeness		
Category	Evaluation		
Actor	CIO | **Supporting Actors:** Assistant of CIO		
Contribution	...		
Cause	Change request (CR) was submitted		
Description	Reject if change request is incomplete. Assistant notifies CR issuer by e-mail to revise the CR. If CR is complete it is further processed.		
Pre condition	CR is available.		
Combined Data	**Entity**	**Attribute**	
	Change request	* (all)	
Computed Data	**Entity**	**Attribute**	**Computation**
	—	—	
Post condition	Incomplete CR is to be revised by issuer.		
Info out	**Entity**	**Attribute**	
	Change request	completeness	
	Re-submission	Change request ID, date	
Rules	–		

Figure D.11: Decision subtask CM 2: Evaluation of change request completeness

ID: Subtask	PM 5g: Approval of message text for issuer about the decision of change request		
Category	Approval		
Actor	CIO | **Supporting Actors:** Assistant of CIO, CAB		
Contribution	...		
Cause	Decision about change request (CR) has been made by CAB.		
Description	Taking note, correction and release of the message text to issuer of CR		
Pre condition	Decision about CR is documented		
Combined Data	**Entity**	**Attribute**	
	change request	message text of rejection or approval	
	change request	issuer	
Computed Data	**Entity**	**Attribute**	**Computation**
	—	—	
Post condition	Approval to send message text to CR-issuer has been given		
Info out	**Entity**	**Attribute**	
	change request	message text of rejection or approval	
Rules	–		

Figure D.12: Decision subtask CM 5g: Approval of message text about the decision of change request

ID: Subtask	CM 4e: Evaluation of existence of change request rating and change request statements		
Category	Evaluation		
Actor	CIO	**Supporting Actors:** Department managers (responsible staff members in IM department).	
Contribution	...		
Cause	Next change advisory board (CAB) meeting is close; re-submission date is reached		
Description	In advance to the CAB meeting the CIO ensures that all change request (CR) ratings and CR statements are available.		
Pre condition	CR is on the agenda for next CAB meeting		
Combined Data	**Entity**	**Attribute**	
	Change request	title, deadline of change, date received, status, revision by issuer	
	Change request statement	*	
	Change request rating	*	
	Re-submission	date	
Computed Data	**Entity**	**Attribute**	**Computation**
	—	–	
Post condition	Possible adjournment of evaluation of change request		
Info out	**Entity**	**Attribute**	
	Re-submission	change request ID, date	
Rules	If CAB reaches no consensus about the CR or not all statements and ratings are available, the CR evaluation is scheduled to a future CAB meeting.		

Figure D.13: Decision subtask change management 4e: Evaluation of existence of change request rating and change request statements

ID: Subtask	CM 5a: Evaluation change request rating		
Category	Evaluation		
Actor	CIO	**Supporting Actors:** —	
Contribution	...		
Cause	Decision about to implement the change request (CR) has to be made.		
Description	Reject CR or revision by issuer; implement CR by (1) operative change, (2) within an ongoing project (3) in a new project.		
Pre condition	Change request (CR) is complete, all CR statements and ratings are available		
Combined Data	**Entity**	**Attribute**	
	Change request	*	
	Change request statement	*	
	Change request rating	*	
Computed Data	**Entity**	**Attribute**	**Computation**
	Division budget	remaining budget	annual budget - expenses
	Division budget	known fixed cost for the next 4 years	multiply(fixed cost per year, 4)
Post condition	Rejection: start subtask CM 5b; revision by issuer: subtask CM 5c; solve CR by operative change: start subtask 5d; solve CR in current project: start subtask 5g; solve CR by new project: start subtask CM 5f;		
Info out	**Entity**	**Attribute**	
	Change request	message text of rejection or approval	
	Change request status	1..5	
Rules	–		

Figure D.14: Decision subtask change management 5a: Evaluation change request rating

D.2.2.2 Decision Point: Domain Data

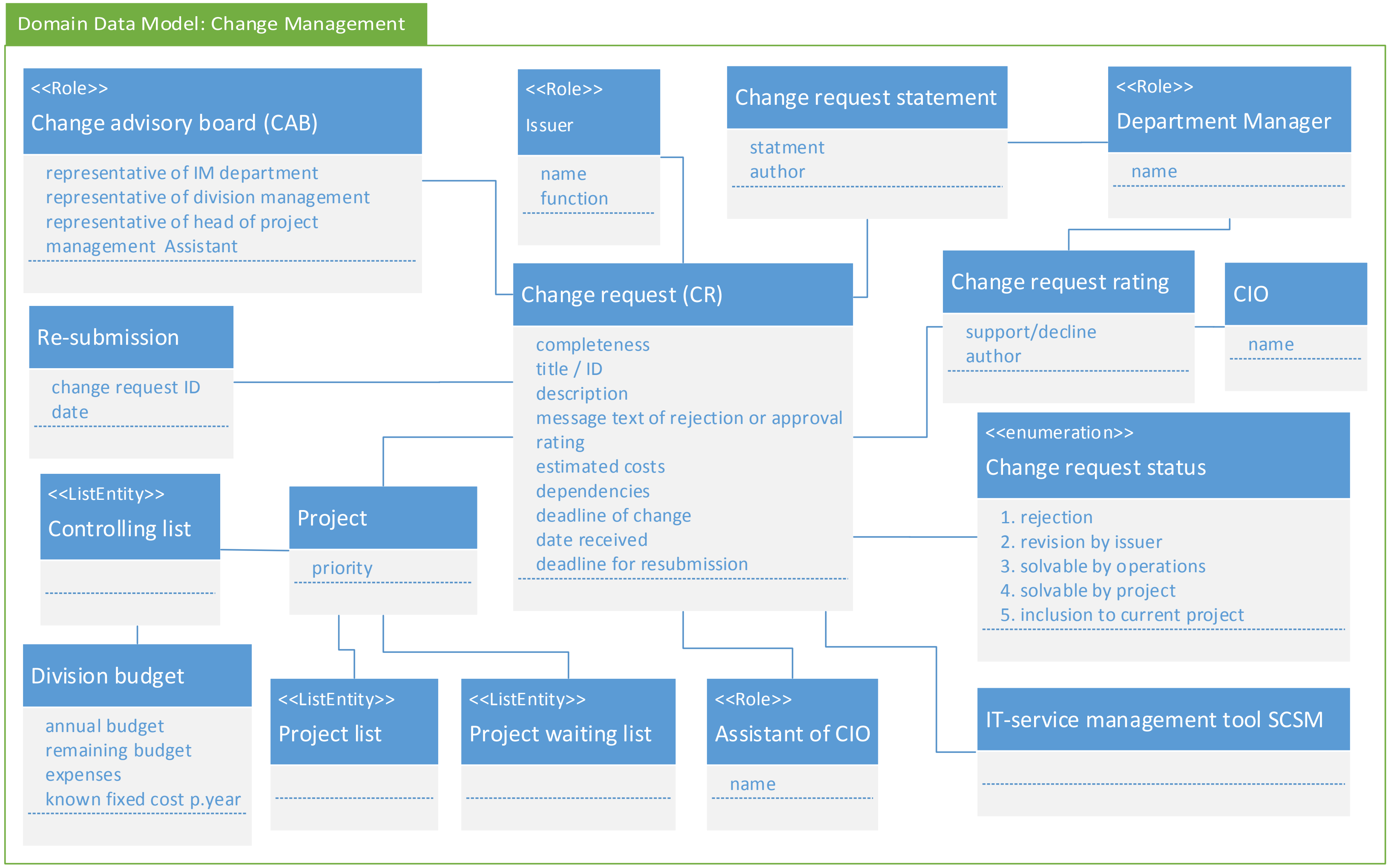


Figure D.15: The domain data model for the task change management

D.2.3 Interaction Level

D.2.3.1 Decision Point: UI Structure

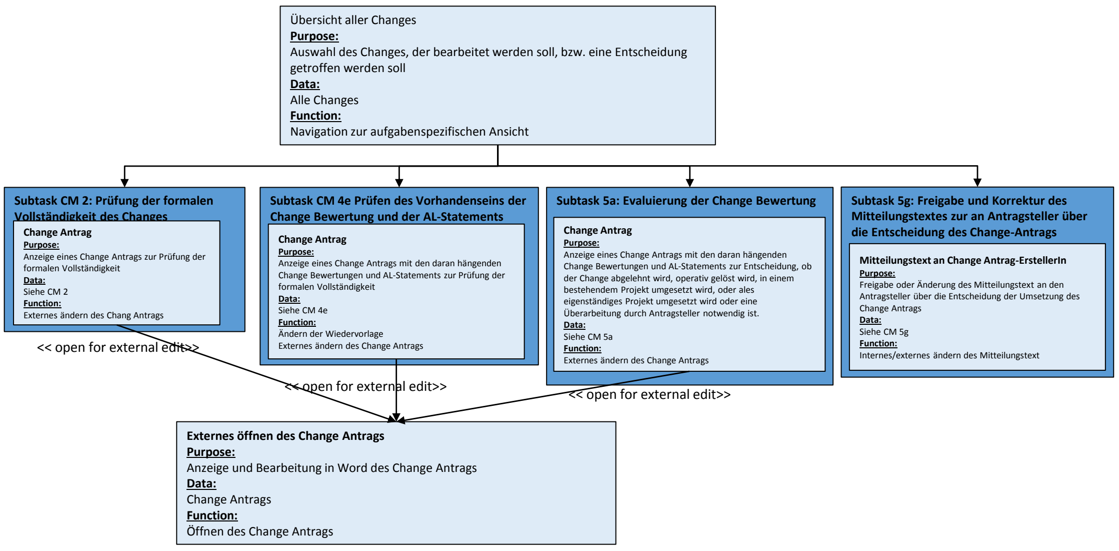

Übersicht aller Changes
Purpose:
Auswahl des Changes, der bearbeitet werden soll, bzw. eine Entscheidung getroffen werden soll
Data:
Alle Changes
Function:
Navigation zur aufgabenspezifischen Ansicht
Subtask CM 2: Prüfung der formalen Vollständigkeit des Changes
Change Antrag
Purpose:
Anzeige eines Change Antrags zur Prüfung der formalen Vollständigkeit
Data:
Siehe CM 2
Function:
Externes ändern des Chang Antrags
Subtask CM 4e Prüfen des Vorhandenseins der Change Bewertung und der AL-Statements
Change Antrag
Purpose:
Anzeige eines Change Antrags mit den daran hängenden Change Bewertungen und AL-Statements zur Prüfung der formalen Vollständigkeit
Data:
Siehe CM 4e
Function:
Ändern der Wiedervorlage
Externes ändern des Change Antrags
Subtask 5a: Evaluierung der Change Bewertung
Change Antrag
Purpose:
Anzeige eines Change Antrags mit den daran hängenden Change Bewertungen und AL-Statements zur Entscheidung, ob der Change abgelehnt wird, operativ gelöst wird, in einem bestehendem Projekt umgesetzt wird, oder ales eigenständiges Projekt umgesetzt wird oder eine Überarbeitung durch Antragsteller notwendig ist.
Data:
Siehe CM 5a
Function:
Externes ändern des Change Antrags
Subtask 5g: Freigabe und Korrektur des Mitteilungstextes zur an Antragsteller über die Entscheidung des Change-Antrags
Mitteilungstext an Change Antrag-ErstellerIn
Purpose:
Freigabe oder Änderung des Mitteilungstext an den Antragsteller über die Entscheidung der Umsetzung des Change Antrags
Data:
Siehe CM 5g
Function:
Internes/externes ändern des Mitteilungstext
<< open for external edit>>
<< open for external edit>>
<< open for external edit>>
Externes öffnen des Change Antrags
Purpose:
Anzeige und Bearbeitung in Word des Change Antrags
Data:
Change Antrags
Function:
Öffnen des Change Antrags

Figure D.16: The workspace model of task change management

D.2.4 System Level

D.2.4.1 Decision Point: UI Data

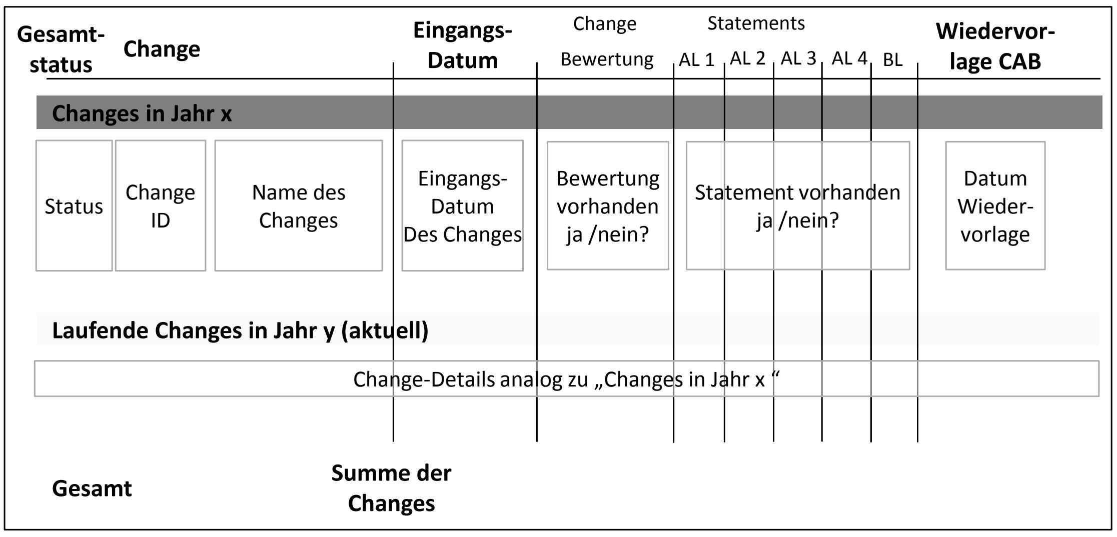
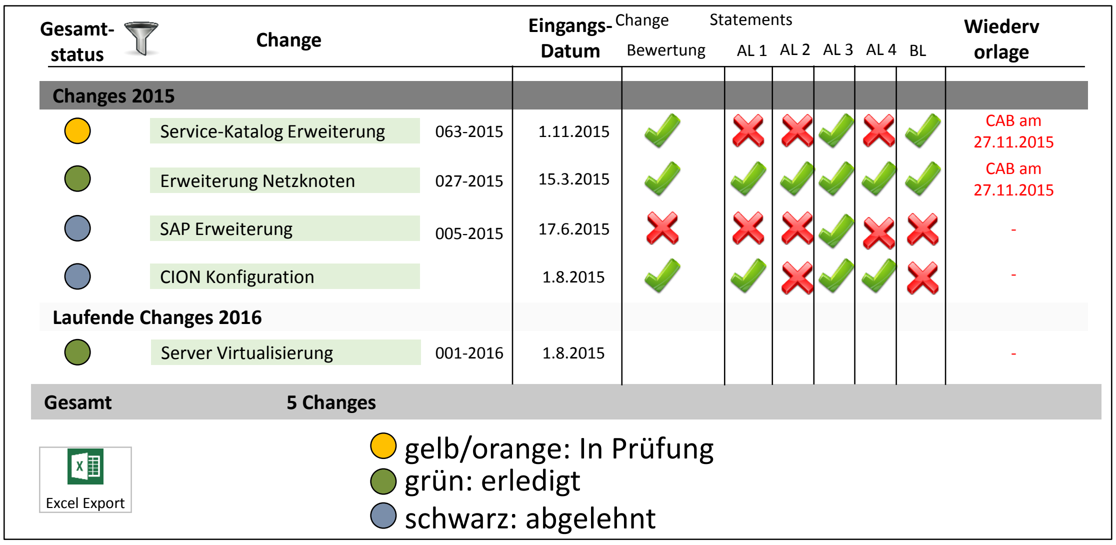


Figure D.17: The virtual window of the change request overview for task change management 4e

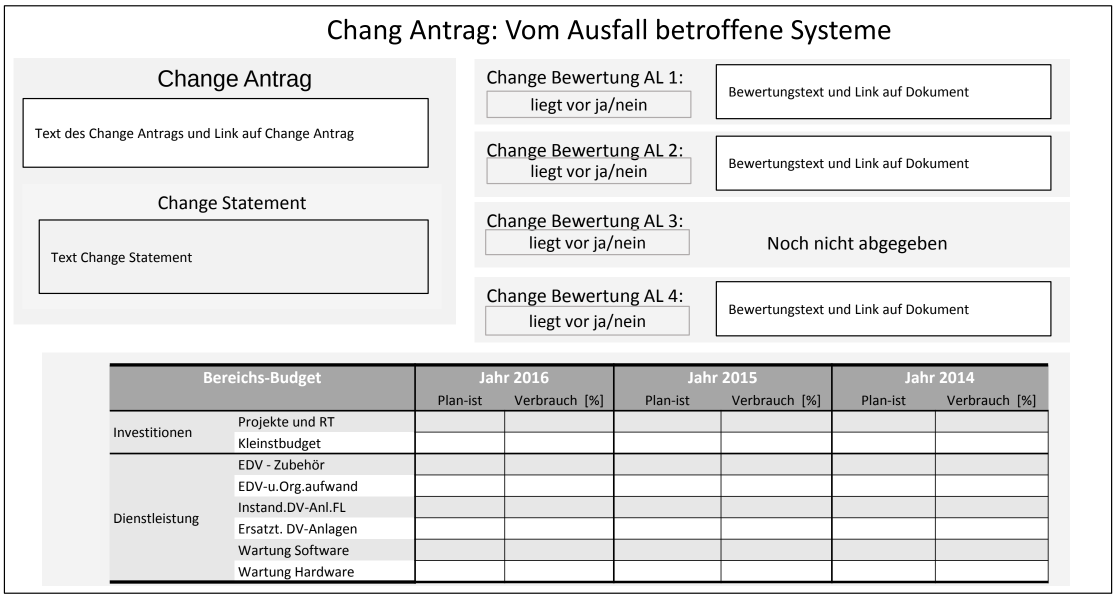


Gesamt-status	Change		Eingangs-Datum	Change Bewertung	Statements AL 1	AL 2	AL 3	AL 4	BL	Wiedervorlage
Changes 2015										
🟡	Service-Katalog Erweiterung	063-2015	1.11.2015	✓	✗	✗	✓	✗	✓	CAB am 27.11.2015
🟢	Erweiterung Netzknoten	027-2015	15.3.2015	✓	✓	✓	✓	✓	✓	CAB am 27.11.2015
🔵	SAP Erweiterung	005-2015	17.6.2015	✗	✗	✗	✓	✗	✗	-
🔵	CION Konfiguration		1.8.2015	✓	✓	✗	✓	✓	✗	-
Laufende Changes 2016										
🟢	Server Virtualisierung	001-2016	1.8.2015							-
Gesamt	**5 Changes**									

Excel Export

🟡 gelb/orange: In Prüfung
🟢 grün: erledigt
🔵 schwarz: abgelehnt

Figure D.18: The UI prototype of change request overview for subtask change management 4e



Chang Antrag: Vom Ausfall betroffene Systeme

Change Antrag

Text des Change Antrags und Link auf Change Antrag

Change Statement

Text Change Statement

Change Bewertung AL 1: liegt vor ja/nein — Bewertungstext und Link auf Dokument

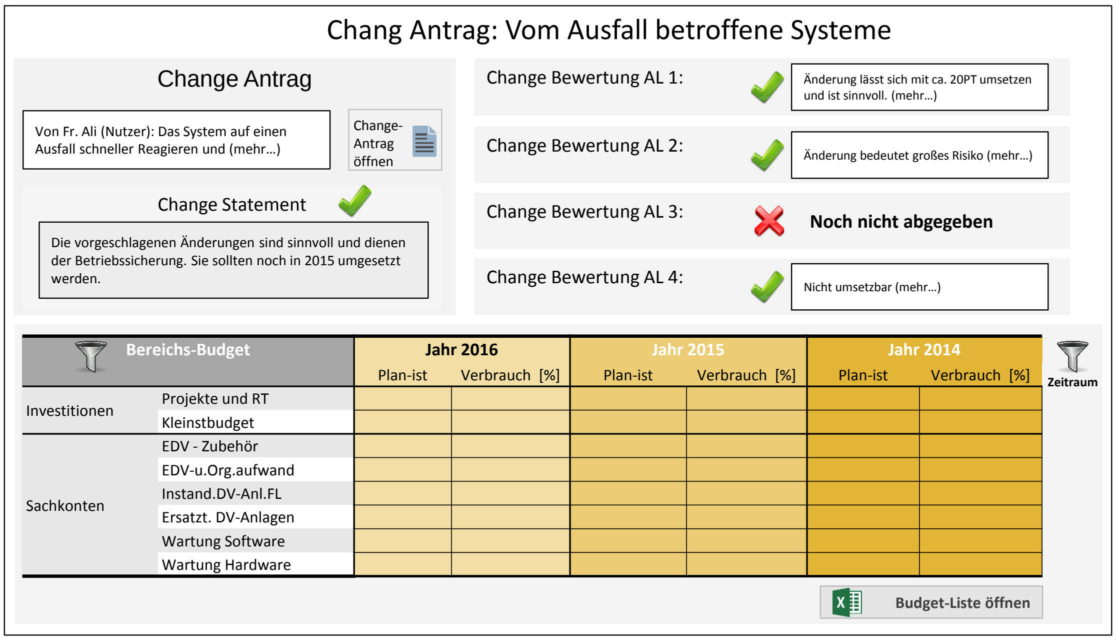
Change Bewertung AL 2: liegt vor ja/nein — Bewertungstext und Link auf Dokument

Change Bewertung AL 3: liegt vor ja/nein — Noch nicht abgegeben

Change Bewertung AL 4: liegt vor ja/nein — Bewertungstext und Link auf Dokument

Bereichs-Budget		Jahr 2016		Jahr 2015		Jahr 2014	
		Plan-ist	Verbrauch [%]	Plan-ist	Verbrauch [%]	Plan-ist	Verbrauch [%]
Investitionen	Projekte und RT						
	Kleinstbudget						
Dienstleistung	EDV - Zubehör						
	EDV-u.Org.aufwand						
	Instand.DV-Anl.FL						
	Ersatzt. DV-Anlagen						
	Wartung Software						
	Wartung Hardware						

Figure D.19: The virtual window of the change appraisal for task change management 4e and 5a



Chang Antrag: Vom Ausfall betroffene Systeme

Change Antrag

Von Fr. Ali (Nutzer): Das System auf einen Ausfall schneller Reagieren und (mehr...)

Change-Antrag öffnen

Change Statement ✓

Die vorgeschlagenen Änderungen sind sinnvoll und dienen der Betriebssicherung. Sie sollten noch in 2015 umgesetzt werden.

Change Bewertung AL 1:	✓	Änderung lässt sich mit ca. 20PT umsetzen und ist sinnvoll. (mehr...)
Change Bewertung AL 2:	✓	Änderung bedeutet großes Risiko (mehr...)
Change Bewertung AL 3:	✗	Noch nicht abgegeben
Change Bewertung AL 4:	✓	Nicht umsetzbar (mehr...)

Bereichs-Budget		Jahr 2016		Jahr 2015		Jahr 2014	
		Plan-ist	Verbrauch [%]	Plan-ist	Verbrauch [%]	Plan-ist	Verbrauch [%]
Investitionen	Projekte und RT						
	Kleinstbudget						
Sachkonten	EDV - Zubehör						
	EDV-u.Org.aufwand						
	Instand.DV-Anl.FL						
	Ersatzt. DV-Anlagen						
	Wartung Software						
	Wartung Hardware						

Zeitraum

Budget-Liste öffnen

Figure D.20: The UI prototype of change request appraisal for subtask change management 4e and 5a

D.3 The Task IT Strategy

D.3.1 Goal & Task level

D.3.1.1 Decision Point: Categorization of Subtasks

We explain the subtasks of Table D.3 in brief to comprehend the categorization and the respective decisions. The task IT strategy consists of two groups: (i) subtasks 1a-1g are related to the creation of the strategic IM plan and (ii) subtasks 2a-2f are related to the use of the strategic IM plan. Before the creation of the strategic IM plan starts, the involved stakeholders (subtask 1a) need to be selected and approved. The CIO nominates the persons of the IT strategy stakeholders, which is finally approved by the hospitals board of directors. An initial set of actions that will be considered in the next strategic IM plan is prioritized by the CIO. This *prioritization* in subtask 1b is the decision about the activities' importance. Then workshops are held with the IT strategy stakeholders and the CIO in subtask 1c where the participants discuss ideas, how the future fields of actions can be realized in the IM department. From the workshop results, the CIO decides a set of IT actions in subtask 1d which fulfill the strategic hospital goals. Afterwards, the CIO *documents* the identified IT

Table D.3: The subtasks of task IT strategy

ID	sub task	category
1	Creation of strategic information management plan	
1a	Approve IT strategy stakeholder	Approval
1b	Prioritize future fields of actions for IT strategy	Prioritization
1c	Perform workshops	Communication
1d	Evaluate workshop results	Evaluation
1e	Document IT actions	Documentation
1f	Prioritize IT actions	Prioritization
1g	Present strategic information management plan	Communication
2	Use of strategic information management plan	
2a	Prioritize projects	Prioritization
2b	Evaluate change request	Evaluation
2c	Justify IM department finances prospectively and retrospectively	Evaluation
2d	Evaluate the security-criticality of projects	Evaluation
2e	Use strategic information management plan for funding applications	Support
2f	Evaluate strategic information management plan adherence	Evaluation

actions in subtask 1e into the *strategic IM plan* document, which is a conventional subtask. In subtask 1f, the CIO *prioritizes* the identified IT actions of the strategic IM plan, which involves the decision about their importance. Finally, the *strategic IM plan* is presented (communication) to the hospitals boards of directors. Projects that contribute to defined IT actions are prioritized in subtask 2a, which is a decision about the preference of some projects to other. The CIO evaluates the change requests' contribution or possible conflicts to IT actions in subtask 2b. The decision results in the preference of change requests which contribute to IT actions. The CIO uses the strategic IM plan to justify the departments' budgets prospectively or retrospectively in subtask 2c. Thus, the CIO needs to decide on a case by case basis, to what extent the expenses of projects or the department at whole contribute to IT actions. As security is in general one of the high priority IT actions, all projects are evaluated regarding their contribution to security. The decision in subtask 2d is about the security criticality of projects. In subtask 2e, the strategic IM plan are used for the writing of funding applications or research project proposals to ensure that all important strategic IM aspects are considered. Finally, the adherence to the strategic IM plan is evaluated in subtask 2f to identify achieved or neglected IT actions. The adherence of the strategic IM plan involves the decision to what extent an IT action is fulfilled or not.

D.3.1.2 Decision Point Stakeholders' Tasks

Table D.4: The task IT strategy in Task & Support template

Task	IT strategy
Purpose	Creation and use of IT strategy
Source	Interview with CIO Mr. X, 28. June 2016;Translated by C.Kücherer, 2017-10-24
Responsible	CIO Mr. X

ID	**Subtask**	**As is solution**	**Example solution**
1	Creation of IT strategy		
1a	Approve IT strategy stakeholder. **Problem:** Important stakeholders for the IT strategy must not be forgotten. (Approval)	The IT strategy steering committee has to be formed prior to the creation of a new IT strategy. Participating stakeholders must be associated to the divisions (but not limited to) *patient care*, *research and teaching*, *information management*. Stakeholder state their requirements and expectations regarding the IT strategy, representing the opinion of their associated divisions. The proposal of IT strategy steering committee participants is presented to the hospital boards of directors for approval.	Task will not be supported by CIONx.
1b	Prioritize future fields of actions for IT strategy. (Prioritization)	Possible areas and their priority that will be addressed in the IT strategy are gathered in workshops with the IT strategy steering committee. For the last IT strategy, 14 workshops have been carried out. Main questions are "which IT related problems currently exist in the organization?" and "which important topics exist in the IM department?". The identified topics are then clustered to coarse-grained topics from which larger IT actions are synthesized. Each IT action can be assigned one or more corporate objectives.	The identification of relevant IT actions can be supported by current Gartner studies and CIRS Reporting (crititical incident reporting) for patient and medical care, which delivers indications for IT weaknesses. Further sources for IT actions could be HIMSS and EMRAM.
1c	Perform workshops (Communication)	The IT strategy project team discuss the future fields of action in workshops, to gather IT actions.	Task will not be supported by CIONx.

1d	Evaluate workshop results (Evaluation)	The analysis and synthesis of workshop results leads to the list of IT actions. The CIO and head of departments are involved in the synthesis.	Task will not be supported by CIONx
1e	Document IT actions (Documentation)	Based on the workshop results (subtask 1d) the project team prepares the catalog of IT actions. IT actions are large and broad topics that are textually described in the IT strategy.	Task will not be supported by CIONx.
1f	Prioritize IT actions. **Problem:** 1) missing consent about IT actions 2) political dimension of IT actions. (Prioritization)	Each IT action is prioritized. Less important IT actions are neglected. As the stakeholder have different goals and requirements, there is not always a consent about IT actions. Some stakeholders have concerns about the documented IT actions.	Consensus can be reached by conflict resolution talks or social voting.
1g	Present strategic information management plan (Communication)	The CIO keeps the whole hospital informed about the existence and binding character of the IT strategy.	Task will not be supported by CIONx.
2	Use of IT strategy		
2a	Prioritize projects (Prioritization)	New projects from proposals or change requests are assessed against the existing IT strategy. Projects that contribute to high priority IT actions are promoted and receive a high priority. This assessment is implicitly made by the CIO.	CIONx could show the IT actions a project contributes to during the prioritization.
2b	Evaluate change requests (Evaluation)	Evaluation of a change request's contribution to IT actions. Change requests that contribute to IT actions will be accepted. This assessment is implicitly made by the CIO.	Task will not be supported by CIONx.
2c	Justify IM department finances prospectively and retrospectively. **Problem:** 1) planned investments must be justified. 2) no explicit relation between projects and IT actions. 3) Additional personnel must be justified. (Evaluation)	The hospitals board of directors is questioning planned investments. As the directors have committed to the IT strategy, investments or additional personnel that contributes to IT actions can be justified.	CIONx should show the contribution of projects to IT actions for the use in presentations. Appropriate metrics should show the total cost and effort of IT actions.

2d	Evaluate the security-criticality of projects (Evaluation)	Projects with impact on IT security are evaluated in more detail.	A Chief Information Security Officer (CISO) requires the project contribution to security specific IT actions. The result of this subtask is an overview of security relevant projects used for the project prioritization.
2e	Use strategic information management plan for funding applications (Support)	The IT strategy is used as the basis for funding applications.	Task will not be supported by CIONx.
2f	Evaluate strategic information management plan adherence. **Problem:** Arguments for non-fulfilled IT actions are missing. (Evaluation)	Whenever a new IT strategy is created, the fulfillment of the current IT strategy is evaluated. Ideally, this evaluation would take place on a yearly base.	An IT board should be established who meet each half year to evaluate the fulfillment of the IT actions documented in the IT strategy. CIONx shows the relation of IT action to its planned budget, number of projects and total costs. This can provide argument for a non-fulfilled IT action (too high costs).

D.3.2 Domain Level

D.3.2.1 Decision Point: To-be decision specific subtasks

Decision subtask IT strategy 2d: Evaluation of security-criticality of projects as shown in Figure D.29 is presented on page 242 for reasons of space.

ID: Subtask	1b: Prioritize future fields of actions for IT strategy	
Category	Prioritization	
Actor | Supp.Act.	CIO | IT-strategy project team	
Contribution	Support of organization's business processes	
Cause	IT-Strategy needs to be created for the next period	
Description	The topics to be covered in workshops are prioritized in advance to ensure that the most important topics for the IM department are taken into account.	
Pre condition	Topics proposals exist	
Combined Data	**Entity**	**Attribute**
	Strategic future fields	topic
	Strategic hospital goal	Description
	IT-Strategy stakeholder	Name, AssignedDivision
	StrategicRessource	Gartner-Study, CIRS Reporting, HIMS, N-RAM Score
Computed Data	–	
Post condition	Prioritized list of topics proposal.	
Info out	**Entity**	**Attribute**
	Topic proposals	priority
Rules	– none given –	

Figure D.21: Decision subtask IT strategy 1b: Prioritization of future fields of activities for IT strategy

ID: Subtask	IT strategy 1f: Prioritize IT actions	
Category	Prioritization	
Actor | Supp.Act.	CIO | IT-Strategy stakeholder	
Contribution		
Cause	Biannual evaluation; creation of new strategic information management plan; review or evaluation meeting with IT strategy board	
Description	Choice of relevant and manageable number of IT actions for strategic IM plan	
Pre condition	List of prioritized IT actions is available	
Combined Data	**Entity**	**Attribute**
	IT action	all
Computed Data	–	
Post condition	List of selected IT actions is available.	
Info out	**Entity**	**Attribute**
	IT action	priority
Rules	– none given –	

Figure D.22: Decision subtask IT strategy 1f: Prioritization of IT actions

ID: Subtask	IT strategy 2c: Justify IM departments finances prospectively/retrospectively
Category	Evaluation
Actor | Supp.Act.	CIO, finance manager, investment management | department manager, CIO's assistant
Contribution	Definition of a budget for the next year.
Cause	Budget planing starts, initiated by finance manager
Description	Planned investments are called into question and must be justified. The hospitals board of directors have commitment to the strategic IM plan, which is a strong argument. The explicit relation between projects and the strategic IM plan supports the justification of additional human resources in projects that are important to reach the strategic IM plan.
Pre condition	Project list exists, containing estimated effort of projects.
Combined Data	**Entity**: **Attribute** Project: IT action Project List: sum of planned budget, sum of assigned budget Controlling List Project: all IT action: estimated cost IT action: complexity
Computed Data	**Entity**: **Attribute**: **Computation** Project List: sum of planned budget: sum of planned budget for selected projects Project List: sum of assigned budget: sum of assigned budget for selected projects
Post condition	The estimation of the IM department's budget for next year regarding the strategic IM plan is available.
Info out	**Entity**: **Attribute** Project List: – Controlling list: –
Rules	– none given –

Combined Data:

Entity	Attribute
Project	IT action
Project List	sum of planned budget, sum of assigned budget
Controlling List Project	all
IT action	estimated cost
IT action	complexity

Computed Data:

Entity	Attribute	Computation
Project List	sum of planned budget	sum of planned budget for selected projects
Project List	sum of assigned budget	sum of assigned budget for selected projects

Info out:

Entity	Attribute
Project List	–
Controlling list	–

Figure D.23: Decision subtask IT strategy 2c: Prospective/Retrospective justification for departments finances

ID: Subtask	IT strategy 2f: Evaluate strategic IM plan adherence		
Category	Evaluation		
Actor | Supp.Act.	CIO | Information management board, project manager		
Contribution	Exoneration from fulfillment of strategic information management plan, collection of lessons learned		
Cause	Biannual evaluation; creation of new strategic information management plan; review or evaluation meeting with IT strategy board		
Description	The IT strategy board evaluates the achievement of planned IT actions. A recommendation of action is given, if postulated IT actions are neglected.		
Pre condition	Existing project status reports with relation to IT actions.		
Combined Data	**Entity**	**Attribute**	
	IT action	project, degree of fulfillment	
	Project	budget, status	
	Project list	(all)	
Computed Data	**Entity**	**Attribute**	**Computation**
	Project	overall cost	sum of single expenses
	IT action	overall cost of projects	sum of all contributing projects overall costs
Post condition	Evaluation of strategic IM plan is done		
Info out	**Entity**	**Attribute**	
	IT action	degree of fulfillment	
	exoneration protocol	exoneration of CIO	
	exoneration protocol	recommendation for neglected IT actions	
Rules	– none given –		

Figure D.24: Decision subtask IT strategy 2f: Evaluation of strategic IM plan adherence

D.3.3 Decision Point: Domain Data

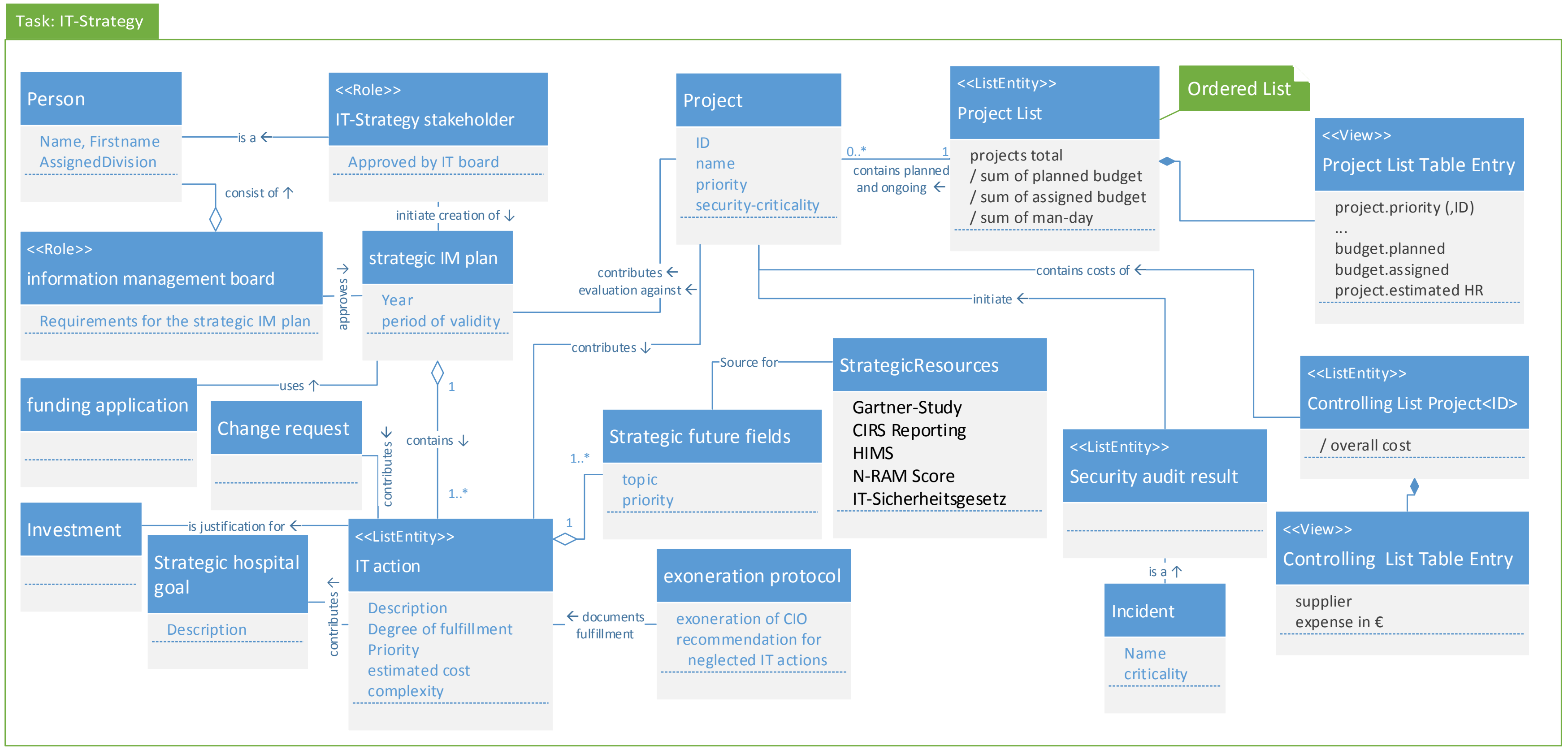


Figure D.25: The domain data model for IT strategy

D.3.4 System Level

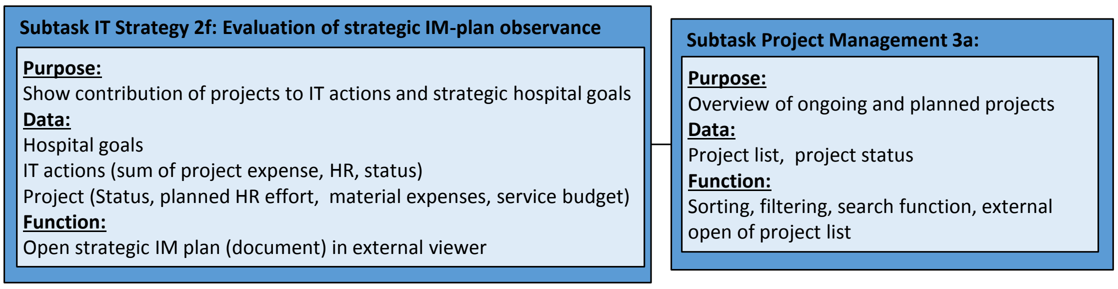

Subtask IT Strategy 2f: Evaluation of strategic IM-plan observance
Purpose:
Show contribution of projects to IT actions and strategic hospital goals
Data:
Hospital goals
IT actions (sum of project expense, HR, status)
Project (Status, planned HR effort, material expenses, service budget)
Function:
Open strategic IM plan (document) in external viewer
Subtask Project Management 3a:
Purpose:
Overview of ongoing and planned projects
Data:
Project list, project status
Function:
Sorting, filtering, search function, external open of project list

Figure D.26: The workspace model of subtask IT strategy 2f

D.3.4.1 Decision Point: UI Structure, UI-Data and Screen Structure

Figure D.26 shows the workspace model for subtask IT strategy 2f. The DP *UI-Data* and *Screen Structure* is documented with the virtual window as shown Figure D.27.

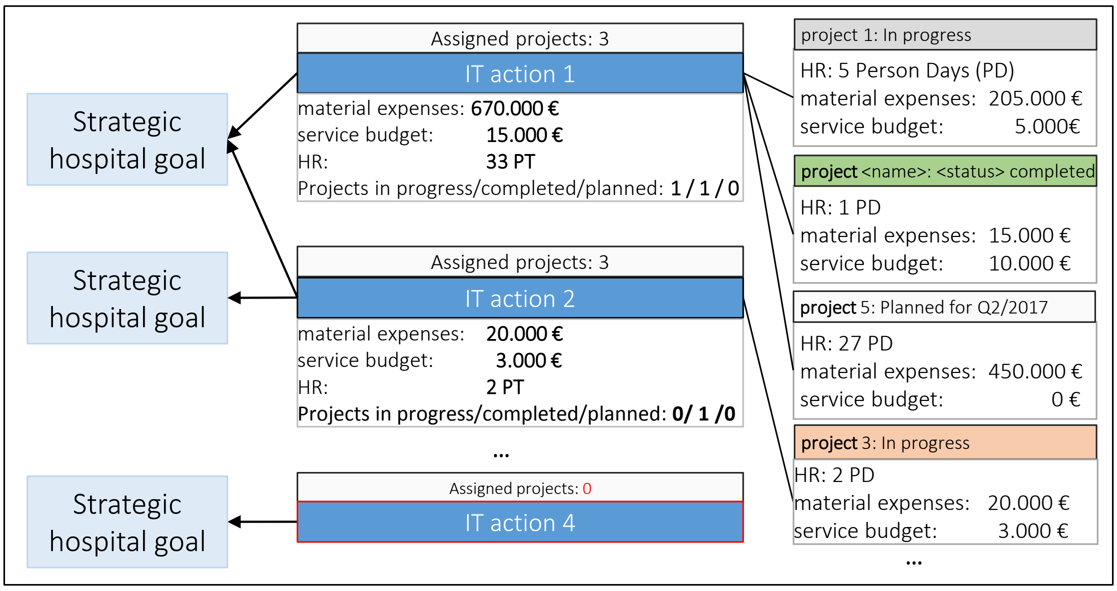


Figure D.27: The virtual window for subtask Evaluation of strategic IM plan adherence

For the subtask 2f (see Table D.3 and Figure D.24), it is essential for the CIO to see an overview of the IT actions defined in the last strategic IM plan along with the details of contributing projects. In particular, the financial aspects of the project are interesting, which relates to subtask 2c: retrospective justification of departments finances. The IT actions are related visually to the strategic hospital goals. With this information presented in one view, the CIO is able to evaluate the degree of fulfillment of IT.

D.3.4.2 Decision Point: Data Sources

Figure D.28 shows the data source model for the task IT strategy. It shows that the strategic IM plan is a (text-)document located on the specific URL on the SharePoint server. Both, the `project list` and the project specific `controlling list project` are spreadsheets, available as comma separated values on the specified URLs on the intranet SharePoint server.

entity	**format**	**location**
strategic IM plan	document	http://intranet.sharepoint/2015/strategicPlan.docx
project list	spreadsheet	http://filesrv1/2016/pwl.xlsx
controlling list project	spreadsheet	http://intranet.sharepointServer/controllingList.xlsx

Figure D.28: The data source model task IT strategy

ID: Subtask	IT strategy 2d: Evaluate the security-criticality of projects	
Category	Evaluation	
Actor | Supp.Act.	CIO, CISO, security panel | project manager	
Contribution	1: ensure to consider security during project planning, and 2: argumentation for difficult projects depends not only on CIO	
Cause	Budget planing starts.	
Description	–	
Pre condition	Project list with security-criticality of projects is available	
Combined Data	**Entity**	**Attribute**
	Project	Name, ID, security-criticality, estimation of resources (assigned budget, man-day)
	strategic IM plan	IT action
	Strategic Resources	IT-Sicherheitsgesetz
	Incident	Name, criticality
Computed Data	–	
Post condition	Estimation of security panel / CIOs rating of a projects' security-criticality	
Info out	**Entity**	**Attribute**
	SecurityAuditResults	–
Rules	– none given –	

Figure D.29: Decision subtask IT strategy 2d: Evaluation of security-criticality of projects

E Additional Material for SLR Domain Ontology Usage

E.1 Ontology Types

Table E.1: The primary studies assigned to the three ontology types.

Ontology type	Studies	Count	%
Domain ontology	S01, S03, S05, S07, S09, S12, S13, S14, S15, S16, S17, S19, S20, S22, S26, S29, S33, S36, S37, S38, S39, S41, S43, S44, S45, S50, S52, S53, S54, S55, S57, S58, S59, S61, S62, S67, SNB01, SNB02, SNB03, SNB06, SNB07, SNB08, SNB10, SNB11, SNB13, SNB14	46	44 %
RE ontology	S01, S02, S04, S05, S06, S08, S09, S10, S11, S12, S13, S18, S19, S20, S21, S22, S24, S25, S27, S28, S29, S31, S34, S35, S38, S40, S42, S47, S51, S56, S58, S60, S62, S64, S65, SNB04, SNB05, SNB09, SNB12	39	37 %
Requirements ontology	S23, S30, S32, S34, S40, S41, S42, S43, S45, S46, S48, S49, S50, S54, S56, S58, S60, S63, S66, SNB10	20	19 %
total	81 studies	105	100 %

Table E.2: The knowledge contained in requirements ontologies (19 approaches in total, multiple mentions possible)

Knowledge in ontology	Studies	Count	%
extracted (formalized) requirements	S23, S30, S32, S34, S41, S42, S43, S46, S49, S50, S54, S56, S58, S60, S63, S66, SNB10	17	77 %
reusable requirements	S40	1	5 %
NFRs	S45, S46	2	9 %
business rules	S46	1	5 %
components, relations, interactions	S48	1	5 %
total	19 studies, rounding error = 1 %	22	100 %

Table E.3: The knowledge contained in RE-ontologies (39 approaches in total, multiple mentions possible).

Knowledge in ontology	Studies	Count	%
RE method or -process concepts	S02, S10, S22, S38, S24, S28, S31, S34, S38, S65, SNB12	11	22 %
semantics, categories, guidelines of requirements	S04, S05, S06, S08, S20, S24, S25, S42, S47, S56, S58, S60, S64, SNB05, SNB09	15	30 %
RE artifacts content, meta data, qualities	S04, S08, S09, S11, S25, S29, S19	7	14 %
UML model semantics	S27, S31, SNB05	3	6 %
software architecture semantics	SNB05	1	2 %
software quality characteristics, metrics	S01, S10, S51, SNB04	4	8 %
security related NFR standards, security threats and defense actions	S06, S12, S13, S18, S21, S35, S40, S62	8	16 %
cross cutting concerns	SNB09	1	2 %
total	39 studies	50	100 %

Table E.4: The domains of used ontologies. (33 studies, 38 mentions, multiple mentions possible, rounding error = 4 %)

Domain	Studies	Count	%
university management	S01, S07, S16, S37, S67	5	13 %
conference organization	S33	1	3 %
news	S44, S61	2	5 %
library	S17, S33, S36, S39	4	11 %
media player	S61	1	3 %
e-health	S45, S54	2	5 %
crisis management	S15	1	3 %
social media	SNB01	1	3 %
meeting scheduler	S09	1	3 %
aircraft, airport	S19, S38	2	5 %
information management	S58	1	3 %
maritime	S62	1	3 %
steam boiler	S13	1	3 %
financial investment	S15, S33	2	5 %
insurance	S22	1	3 %
taxation	S41	1	3 %
e-commerce	S20, S43, S44, S50	4	11 %
general knowledge	SNB07 (encyclopedia)	1	3 %
none given	S03, S05, S12, S14, S53, S55	6	16 %
total	18 domains, 37 studies	38	100 %

E.2 Ontology Metamodel Terms

AbstractRequirement (1) AbstractSource (1) action (2) actions (2) ActiveEntities (1) activity (2) actor (4) AdjectivalProperties (1) AdverbialProperties (1) Algorithms (1) ambiguity (2) Assumption (1) atomic (1) attribute (1) behaviour (1) BooleanCombination (1) Business (1) BusinessEntities (1) BusinessGoals (1) BusinessProcesses (1) cause (1) class (2) classification (1) Complement (1) component (1) composit (1) concept (5) consequence (1) constrained (1) constraint (2) Constraints (1) context (1) CountrySpecificLaws (1) data (4) DataProperty (1) DataPropertyValue (1) DataType (1) DataValue (1) Dependency (1) deprecated (1) description (1) deviation (1) Directed (1) Document (1) domain (1) DomainClass (1) EdurentEntities (1) entities (1) EnumeratedClass (1) environment (2) event (2) exclude (1) extra-system (1) FeatureModelDescriptionLanguage (1) features (1) FMDL (1) function (4) FunctionalProperties (1) FunctionalRequirements (1) functions (1) Goal (2) GraphProductLine (1) GraphType (1) guide (1) hardgoal (1) How (1) Impact (1) InactiveEntities (1) Individual (1) InstanceType (1) interface (1) Intersection (1) ISOModel (1) material (1) MeasurementUnit (1) message (1) MorphologicalInformation (1) NonfunctionalRequirement (1) object (6) ObjectPropertyValue (1) OMG-ODM (2) OntologySchemaSWORE (1) operation (1) organization (1) OrganizationalPolicies (1) parts (1) PerdurentEntities (1) person (1) possession (1) PrimitiveType (1) process (1) ProductEntities (1) properties (1) property (4) PropertyValue (1) proptery (1) QualitativeProperties (1) quality (3) QualityMetric (1) recommendation (1) Require (1) Requirement (1) resource (1) Restriction (1) Result (1) risklevel (1) Role (1) safeguard (1) Scenario (1) Search (1) SemanticInformation (1) softgoal (1) Stakeholder (1) state (2) StudyNode (1) Subject (1) SubProcesses (1) Superclass (1) Symbol (1) SyntacticInformation (1) system (1) table (1) Targetclass (1) task (1) taxonomy (1) top-level (1) Undirected (1) Union (1) UniqueID (1) Unweighted (1) UseCases (1) VariableAttributesOfBoilerplates (1) Verb (1) Weighted (1) When (2) Where (2) Who (2) Why (2) word (2)

Figure E.1: Overview of metamodel terms. (Term frequency in parenthesis)

achieve (1) aggregation (3) anonym (1) apply (1) association (1) complement (2) ComplyWith (1) conflicts (1) contains (2) contradict (1) contradiction (1) DependsOn (1) disjoint (2) entails (1) equivalence (2) equivalent (2) generalization (1) HasA (1) inconsistent (1) inheritance (2) invalidates (1) IsA (2) IsComposedOf (1) IsDetailedBy (1) IsFollowedBy (1) IsInputtedTo (1) IsInvalidFor (1) IsKindOf (1) IsPartOf (2) IsPerformedBy (1) IsSimilarTo (1) LeadsTo (1) manipulate (1) mutual (1) OutPut (1) partial (1) Participate (1) perform (1) redundancy (1) redundant (1) refines (1) reflective (2) relationships (1) require (1) requires (1) subclass (1) SubClassOf (2) support (1) symmetric (2) transitive (3) use (1)

Figure E.2: Overview of metamodel relations. (Term frequency in parenthesis)

E.3 Extraction Form

Table E.5: The columns of the extraction form. Col. refers to the column identifier (A–ZZ) in the extraction-spreadsheet. attrib.=attribute, overv.=overview

Column	Study data	Description	RQ	Metric
A	Study ID	S_i refers to Dermeval studies and SNB_i to the studies found by snowball search	Overv.	–
B	Author	The authors of the article	Overv.	–
C	Title	The title of the article	Overv.	–
D	type of ontology	RE-, domain- or requirements ontology	RQ1.1	m1.1.1
E–G	*reserved* (3 columns)	for ontology types coded in single columns	RQ1.1	m.1.1.1
H	number of ontology types	number of ontology types	RQ1.1	–
I	name in text	ontology name or description in text	RQ1	–
J	type of ontology	ontology type according Dermeval Table 12	RQ1.1	–
K	knowledge in ontology	the knowledge contained in the ontology	RQ1.2	m1.2.1
L	postulated RE quality attrib.	postulated improved qual. attribute of article	RQ2.1	–
M	addressed RE quality attrib.	identified improved quality attributes according ISO/IEC 29148	RQ2.1	m2.1.1
N–AA	*reserved* (14 columns) for	quality attributes coding in single columns	RQ2	m2.1.1
AB	number of RE quality attrib.	number of improved quality attributes	RQ2.1	m2.1.2
AC	specification of quality attrib.	explicit or implicit specification	RQ2.1	–
AD	details of quality attribute	details of quality attribute	RQ2.1	–
AE	type of text based artifacts	type of text based artifacts in focus	RQ2.2	m2.2.1
AF	type of model based artifacts	type of model based artifacts in focus	RQ2.2	m2.2.2
AG	specification level	if explicitly mentioned	–	–
AH–AI	*reserved* (2 columns)	related TORE DPs and specification level	–	–
AJ	domain of ontology	the domain of ontology	RQ3.1	m3.3.1
AK	ontology metamodel	details of the ontologies metamodel	RQ3.2	m3.2.1
AL	type of instances of ontology	individuals present in the ontology	RQ3.2	m3.2.2
AM	modification of ontology	how is the ontology modified during RE?	RQ3.3	m3.3.1
AN	querying of ontology	how is the ontology queried during RE?	RQ3.3	m3.3.2
AO	reasoning of ontology	how is reasoning used during RE?	RQ3.3	m3.3.3
AP	other actions	any other actions performed with ontology	RQ3.3	–
AQ	output during modification	kind of output generated during modification	RQ3.4	m3.4.1
AR	input used for querying	information used to create queries	RQ3.4	m3.4.2
AS	input used for reasoning	information used as precondition for reasoner	RQ3.4	m3.4.3
AT	kind of rules used	kind of used rules or axioms	RQ3.4	m3.4.4
AU	artifact modification	kind of artifact modification with ontology	RQ3.5	m3.5.1
AV	tool exists	yes if a tool is presented or mentioned	RQ3.6	m3.6.1
AW	features described	tool features described or shown	RQ3.6	m3.6.2
AX	artifact related preconditions	type of preconditions of artifacts	RQ4.1	m4.1.1
AY	ontology related preconditions	type of preconditions of ontology	RQ4.1	m4.1.2
AZ	preconditions for requirements	details of preconditions	RQ4.1	m4.1.3
BA	type of approach evaluation	none, (quasi-)experiment, case study	RQ4.2	m4.2.1
BB	Dermevals evaluation	evaluation as in Dermeval's Table 13	RQ4.2	–

Column	Study data	Description	RQ	Metric
BC	type of example	none, simple example, or real world example	–	–
BD	comments	such as name of example in text	RQ4.2	–
BE	short description of approach	summary of the approach in own words	–	–
BF	pattern	set of identified pattern	RQ3	–
BG–BO	*reserved* (9 columns)	for patterns coded in single columns	RQ3	–
BP	relevant	contains *no* for articles that use DO not for requirements quality improvement.	Incl. criterion 10	
BQ	reason for relevance	Reason and details for non relevant articles	Incl. criterion 10	
BR	tools and frameworks used	Details of used technology, tools, framewrks.	RQ4	–

E.4 Tool Features

Table E.6: Identified tool features of the presented tools.

Ontology learning and population	
Ontology learning (domain knowledge)	creation and maintenance of DOs (add, modify, delete the concepts or relationships of domain knowledge); mining and presentation of general domain concepts; ontology editor; wiki-like user interface to contribute domain knowledge; manual inclusion of concepts into DO; discovery of new relationships in the ontology with reasoner; specification of new rules in context free grammar; split of DO into subdomain-ontologies; merging and validating ontologies.
Ontology population (with requirements)	Parse initial requirements, similarity calculation between two strings based on thesaurus; search of requirements terms in ontology; formalize requirements into OWL; specify relationships between requirements; automatic maintenance of the ontology to represent the requirement content; transforming CNL descriptions into ontologies; semi-automatic transformation of natural language specifications into ontology;
Annotation and linking	
Annotation & link usage	link requirements to ontology classes; annotation of feature models with non functional concepts; find ontological concepts for requirements terms using a synonym dictionary; show textual definitions of concepts as tooltips;
Knowledge reuse	
Knowledge management and reuse	modify (add, edit, remove) knowledge elements; wiki-like user interface to contribute domain knowledge; access the domain-knowledge by searching with parameters; find past solutions in a specific domain or for NFR to reuse; social media information sharing and collaboration among the RE stakeholders;
Complete requirements templates with attributes	
Template rel.	provide boilerplate repository; selection of boilerplates;
Attribute related	offering metrics for a given quality requirement; provide list of fully formulated boilerplates; manual fill of boilerplate parts to complete the formulation of the requirement; show attribute values of the requirements; suggestions of verb-object pairs and subject-verbs;

Review ontology	
Ontology visualization and review	browse in ontology; graph visualization; ontology representation of feature models; display the ontology thesaurus part; check ontology class hierarchy; select concepts from ontologies for composing model of existing system;
Defects detection	
Checking requirements	Check the requirement using relationship and rules of ontology; check consistency and realism of requirements; detect potential conflicts between requirements; show missing terms from the DO; present possible interactions with other domains; detection of missing concepts by inference (require-relationship); check ontology consistency with rules; conflict detection mechanism showing conflicting situations; detect inconsistency, redundancies and overlaps; detect conflicts in activity diagram evolution; inconsistency detection and resolution; validate use cases against a DO; validate requirements by checking the ontologies consistency with a reasoner.
Reporting and visualization	produce a report of recommendations; report on inconsistencies; highlight conflicting goals in the goal graph; validation view to check the consistency of the ontology, showing conflicting concepts and their classification; automated generation of explanations for detected requirements problems.
Improvement	suggestion of missing requirements (considering domain concepts without respective requirements); Domain specific recommendation of complementary and conflicting of features based on business rules; give advice to the requirements engineers to add certain objects to specification; filtering of ambiguous text in requirements; inference engine to deduce goal candidates;
Supporting features, Import, export, tool integration	
Import and export	import of RE documents and DOs; list of all requirements; parse and create documents or models; transform requirements in ontology into UML model; convert UML model to ontology; building UML class models from ontology.
RE functionality	create, manage, modify early textual requirements; enter requirements; access existing UML activity diagrams; modeling of use cases following a use case metamodel; enable users to enter artifact specifications in natural language.
Other tool integration	mind mapping tool; export from ontology in Protégé to IBM Doors; RETool integrated in Protégé; integration of ontology editor; goal graph editor;

E.5 Identified Frameworks and Integrated Tools

Table E.7: Frameworks, formal languages or tools used or integrated in the approaches.

Framework or tool	Description	Studies
OWL (Web ontology language)	W3C specification to describe ontology on a formal description language.	S01, S05, S09, S13, S16, S22, S38, S39, S45, S50, S54, S67
Protégé	Ontology editor, API	S01, S09, S38, S39, S53, S54, SNB01

Framework or tool	Description	Studies
Pellet Reasoner	external reasoner for OWL/RDF	S09, S39, S50, S54, SNB01
SWRL rules	W3C submission: a Semantic Web Rule Language Combining OWL and RuleML	S22, S39, S67, SNB01
openNLP	Stanford NLP Parser	S12, S13 S16, S22
JessTab	Java expert system shell, plug in for Protégé	S39, S67, S01
WordNet	Lexical database of English terms.	S12, S22, S62
Eclipse Framework and plug-ins	Rich client platform, feature model, rBPMN, SAFDML editor, modeling framework (EMF)	S05, S39, S53
RDF	Resource description framework	S22, S05
(X)SPARQL	Query language for OWL/RDF ontologies	S05, S09
SROIQ	Description Logic	S50
OntoLancs	Ontology learning framework	S37
OntoViz, OWLViz	Protégé plugin for visualization	S09
JUNG library	Java Universal Network Graph Framework	S50
ACEView Plugin	Ontology and rule editor plugin for Protégé	S54
Fact++	OWL-DL reasoner	S54
Racer	knowledge representation system	S67
SHOIN	Description Logic	S67
PROMPT	Protégé plugin for management of multiple ontologies	SNB01
Text2Onto	ontology learning framework using textual documents	SNB01
OOPS!	OntOlogy Pitfall Scanner	SNB01
XQuery	XML query language for XML-databases by W3C	S05
Jena	semantic framework	S12
COPPEER	Peer-to-peer framework	S14
SubVersion API	software versioning and revision control system	S14
MindManager	Tool to create hierarchical graphs	S38
IBM Doors	Requirement management tool	S38
ATL	UML metamodel to OWL metamodel converter	S39
UML2OWL	Use Case converter based on ATL Use Case - UML ODM Implementation to convert UML to OWL	S39
	programming and query languages (except for ontology query languages)	
C#	programming language	S44, S43
.NET	application platform	S44, S43
SQL	database server (Microsoft, MySQL)	S01, S13, S44, S43
Java	programming language	S12, S36, S39, S50, S01
Prolog	Logical programming language	S16

E.6 Input and Output Types of Domain Ontology Usage Patterns

Table E.8: Input, output types, and intermediate results of patterns

input / output type	ontology learning p	ontology population p	consolidate req. p	review DO p	glossary terms p	knowledge base p	complete template p	reasoning p	filter req. text p
Input types									
domain ontology	✗	✗	✗	✗	✗	✗			✗
populated domain ontology		✗						✗	✗
consistency rules	✗	✗						✗	
completeness rules	✗	✗						✗	
individual requirement	✗	✗		✗	✗	✗		✗	✗
software requirements specification	✗	✗	✗	✗	✗	✗		✗	✗
Model			✗						
Boilerplates, templates							✗		
Search question						✗			
Output types									
Report of requirements problems			✗					✗	
Report of requirements candidates								✗	
Improved individual requirement				✗	✗	✗	✗	✗	✗
Improved software requirements specification				✗	✗	✗	✗	✗	✗
Modified domain ontology	✗	✗		✗					
Knowledge elements						✗			
Mapping		✗					✗		
domain ontology individuals						✗			

E.7 TOREOnto Evaluation - Rating Table

Table E.9: Rating of the requirements quality attributes of the artifacts. Metrics: **o**riginal / **i**mproved, n/a not applicable, – none given or found; **severity**: 👍 unproblematic, 👉 neutral, 👎 problematic; **completeness rating**: + sufficiently descried, ○ partially sufficiently described, – insufficiently described; **understandability** (underst.): ○○ very easy, ○ easy, ● difficult, ●● very difficult; **correctness**: ✗ incorrect, ✓ correct; **details**: ∅ none, ↑major, ↓ minor.

DP	conflicts				completeness				interpretations				underst.		synonyms				correctness			
	details		severity		rating		severity		rating		severity		degree		details		severity		rating		severity	
	m4.1.1		m4.1.2		m4.2.1		m4.2.2		m4.3.1		m4.3.2		m4.3.3		m4.3.4		m4.3.5		m4.4.1		m4.4.2	
	o	i	o	i	o	i	o	i	o	i	o	i	o	i	o	i	o	i	o	i	o	i
Supp.stakeh.	↓	n/a	👍	n/a	–	+	👍	n/a	∅	∅	n/a	n/a	○○	○○	yes	no	👍	n/a	✗	✗	👍	👍
Stakeh.tasks	∅	↓	n/a	👍	○	+	👉	n/a	∅	∅	n/a	n/a	○○	○○	yes	no	👉	n/a	✗	✓	👍	n/a
As-is to-be act.	↑	↓	👍	👍	○	+	👍	n/a	yes	∅	👎	n/a	○	○○	yes	no	👎	n/a	✗	✗	👉	👍
Dom.data	↑	∅	👍	👍	+	+	👍	👍	∅	∅	n/a	n/a	●	○	yes	no	👉	n/a	✗	✓	👍	n/a
Sys.funcn/a		↓		👍		+		👉		∅		n/a		○○		yes		👍		✓		n/a
	m6.1.1		m6.1.2		m6.2.1		m6.2.2		m6.3.1		m6.3.2		m6.3.3		m6.3.4		m6.3.5		m6.4.1		m6.4.2	
Supp.stakeh.	↑	n/a	👎	n/a	–	–	👎	👉	yes	∅	👎	n/a	●	○○	yes	no	👎	n/a	✗	✓	👎	n/a
Stakeh.tasks	↑	n/a	👎	n/a	–	–	👎	👉	yes	∅	👎	n/a	●	○	yes	no	👎	n/a	✓	✓	n/a	n/a
As-is to-be act.	n/a	↑	.	👉	+	+	n/a	n/a	yes	∅	👎	n/a	●	○	yes	no	👉	n/a	✗	✓	👉	n/a
Dom.data	↓	↓	👉	👉	–	–	👉	👉	yes	∅	👎	n/a	●	○	no	no	n/a	n/a	✗	✓	👎	n/a
Sys.func.		–		n/a		+		n/a		∅		n/a		○		no		n/a		✗		👎
Supp.stakeh.	↓	–	👉	n/a	–	+	👎	n/a	yes	yes	👍	👍	○	○○	yes	no	👎	n/a	✗	✓	👉	n/a
Stakeh.tasks	–	–	n/a	n/a	–	–	👍	👎	no	yes	n/a	👎	○	●	yes	no	👍	n/a	✗	✗	👍	👎
As-is to-be act.	↓	↓	👍	👎	–	–	👎	👎	no	no	n/a	n/a	○	●	no	no	n/a	n/a	✓	✓	n/a	n/a
Dom.data	↓	–	👍	n/a	–	–	👍	👉	no	no	n/a	n/a	○	○○	yes	no	👍	n/a	✗	✓	👍	n/a
Sys.func.		–		–		–		n/a		no		n/a		●		no		n/a		✓		n/a

F Abbreviations

3LGM^2	three layer graph based model
AI	artificial intelligence
BPMN	business process model and notation
CIO	Chief Information Officer
CION	CIO navigator
CIONx	CIO navigator for the specific hospital x
CISO	Chief Information Security Officer
CNL	controlled natural language
DDM	domain data model
DO	domain ontology
DP	Decision Point
DSL	domain-specific language
DSML	domain-specific modeling language
DSS	decision support system
DsTORE	decision specific TORE, as presented in Chapter 6
DW	data warehouse
EMF	eclipse modeling framework
GORE	Goal Oriented Requirements Engineering

GQM	Goal Question Metric
GUI	graphical user interface, synonymous to UI
HIS	hospital information system
HR	Human Resources
HTTP	Hypertext Transfer Protocol
ID	Any kind of identifier
IDM	interaction data model
IEEE	Institute of Electrical and Electronics Engineers
IM	information management
InR	individual requirement
IR	information retrieval
IS	information system
IT	information technology
ML	machine learning
NFR	non functional requirement
NLP	natural language processing
OLAP	online analytical processing
OMG	Object Management Group
OMG-ODM	Object Management Group - Ontology Definition Metamodel
OWL	Web Ontology Language
PDSS	personal decision support system
POS	part of speech
RDF	resource description framework
RE	Requirements Engingeering

RQ	research question
RQA	Requirements quality attribute as defined by ISO/IEC 29148 [ISO11].
SE	Software Engineering
SLR	systematic literature review
SNB	The studies found by snowballing
SNIK	semantic network of information management in hospitals
SPARQL	A graph based query language for RDF ontologies, specified by W3C.
SRS	software requirements specification
SysML	systems modeling language
TORE	task-oriented requirements engineering as described in Section 2.2
TOREOnto	TORE extension to use domain ontologies, as presented in Chapter 10
UI	user interface
UML	unified modeling language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VW	Virtual Window

G Glossary

Application Domain Anything to which computing may be applied [Bjø06]. Example of an application domain is information management in hospitals.

Domain The term **domain** is used in literature in differently. Overall a *domain* is a sphere of knowledge, influence or activity [Eva04]. Some authors refer to a domain as a universe of discourse that can be spoken about [Bjø06]. Further, a *domain* denotes the application area [Lau02], what we call system domain. In most cases the term *domain* and *application domain* can be used synonymous, which also shows the definition that a domain is a specific problem or business area where software systems are used [Som09].

Domain Knowledge is valid knowledge about a universe of discourse, typically the application domain or any other specialized discipline. Domain experts use and develop their own domain knowledge, based on a defined terminology. Domain knowledge is generated by a domain analysis, often related to a high effort. The communication between users, software engineers, requirements engineers, stakeholder and domain expert is based on domain knowledge. Domain knowledge can be formalized into a domain ontology that provides reuse and semantic relations inside the domain knowledge.

Domain Ontology (DO) A *domain ontology* is a semantic description of real world concepts of a specific domain. The domain ontology might be specific to an certain application, software or IS. Examples for domains are aerospace, e-business, or information management. A DO may comprise, among other things, typical roles in the domain, functions or tasks, common application components, or data entities. An example of a DO is the SNIK ontology [JSPW14], showing concepts and their relations of the application domain IM in hospitals.

Domain-specific language (DSL) is a formal language for a certain problem area. DSLs comprise different types of languages, that are *domain-specific markup languages*, *domain-specific modeling languages*, and *domain-specific programming languages*. For this thesis, we focus on DSL in context of domain-specific modeling. A **domain-specific modeling language** – also called conceptual modeling language – needs a definition. The language of a model is called a metamodel. The language for defining a modeling language is thus a meta-metamodel. Meta-metamodels can be derived or be a customization of existing languages. Examples of derived meta-metamodels are entity relationship diagrams, formal languages, or ontology languages. RE artifacts (such as tasks or domain data models) can be expressed in a general-purpose conceptual modeling language such as UML, ER diagram, object relational mapping [GFG12].

Entity Relationship Diagram (ER diagram) An entity–relationship model (ER model) describes inter-related things of interest in a specific domain of knowledge. Originally this notation was developed by Chen[Som09] for database modeling. The entity types of an ER model classify the things of interest and are associated with relationships in between. In the SE domain ER models are used to represent the main domain entities, that is to say tasks or business processes that a person, business or organization needs to perform. Consequently, the ER model is an abstract data model that defines a data or information structure that can be implemented in relational databases.

Goal Oriented Requirements Engineering (GORE) are RE methods that try to capture the rationale for the software systems by a focus on user goals [Lap05]. Goal models are used to capture, represent and align software requirements with goals by using AND/OR goal graphs. Individual requirements must trace to goals of the goal graph and thus provide the rationale for the requirement [EB13, SKS07].

Information Management (IM) Information management is about the management of information systems. It encompasses the management activities related to information processing in an organization. The goal of information management is to structure and provide a systematic information processing following the organizational goals [WHA$^+$11].

Information Technology Infrastructure Library (ITIL) Tasks of operational IM have adopted best practices. Such best practices are documented in the Information Technology Infrastructure Library (ITIL) [Arr13]. ITIL is one of the major approaches to structure and standardize IT operations. It contains a set of best practices for service design, service transition and service operation, governed by a service strategy, allowing a continuous service improvement. Several hospitals have adopted ITIL within their operational IM [HHBA11]. However, the tasks, entity types, tools and adoption of ITIL depend on the hospital needs, organizational structure and complexity of IT used.

Ontology conceptualize real world knowledge as formal concepts, their relations to each other, and their rules [HP09]. In this work we follow the definition of a *computational* *ontology* from Studer et al.: “An ontology is a formal, explicit specification of a shared conceptualization” [SBF98]. Conceptualization signifies an abstract and simplified view in an area of interest of the real world, for example the domain. The conceptualization contains objects, concepts, other entities, and their relations to each other. Formal means that the expressions of conceptualization (the concepts) are given in a formal language. The explicit specification means the intentional use of axioms that define the semantics of the vocabulary in a formal language, for example the relation *cooperates-with* between two persons is symmetric, irreflexive and intransitive. An ontology can be *queried* by a formal language such as SPARQL [HBS09] to access its concepts, relations, and individuals. With *automated reasoning*, logical inference is supported based on rules (the axioms) that are contained in an ontology. We distinguish three different types of ontologies: requirements ontology, RE ontology, and domain ontology.

Operational information management has the goal to successfully operate the components of ISs and thus to provide services for users. In context of the application domain of this thesis, the IS is more specifically a hospital information system. The task of operational IM comprise the planning, directing and monitoring of services for the HIS [WHA+11].

Personal Decision Support System (PDSS) are specialized IS that are a subset of DSS, developed for a single or a small group of managers with the goal to improve the process and outcome of decision-making. The scope of a PDSS is the support of one or a small number of decision tasks, where often semi-structured data, based on spreadsheets, are used. In a similar way to DW, the primary *users* are executives, who access heterogeneous and isolated *data sources* at runtime. In contrast to DW, this usage is repetitive and predefined and the amount of accessed data is not extensive. In summary, the data of PDSSs are complex, while their usage is simple.

Requirements Engineer A requirements engineer is a role in which a person acts with the aim to elicit, document, and manage (gather) requirements from stakeholders. The requirements engineers usually has a sound understanding of RE processes and activities, and should have a basic understanding of the stakeholders problem domain and software engineering domain. Synonyms: analyst, business analyst, requirements analyst.

Requirements Engineering (RE) is an iterative process comprising the activities to systematically elicit, elaborate, structure, document and manage requirements from stakeholders with the goal to specify a software intense system [Som09, vL01]. The aspect of specifying requirements focus on *building the right system*, whereas testing and verification focus on *building the system right*. This incorporates an understanding of the environment and context of the intended system so that requirements reflects what stakeholders want. RE is part of the software engineering discipline and is typically performed within projects.

Software Engineer A software engineer is a person who applies the principles of software engineering to the design, development, maintenance, testing, and evaluation of the software that make computers or other devices containing software work.

Software Engineering (SE) The Software Engineering Body of Knowledge defines Software Engineering as the "application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software" [BF14].

Strategic information management comprise the tasks to establish strategies and principles for the evolution of the hospital information system. The strategic goals of the IM department are the result of strategic management activities and are documented in a strategic information management plan [WHA$^+$11].

System Domain We define a *system domain* as a class of ISs that have a set of common requirements, terminology, functionality to solve similar problems in any domain.

Tactical information management has the goal to introduce, remove or change application components or functions of ISs in form of projects. In context of the application domain of this thesis, the IS is more specifically a hospital information system. Tactical IM projects are initiated to implement strategic IM goals, defined in the strategic IM plan. The result of tactical IM projects is the HIS [WHA$^+$11].

Bibliography

[ADEG09] Sebastian Adam, Joerg Doerr, Michael Eisenbarth, and Anne Gross. Using Task-oriented Requirements Engineering in Different Domains – Experience of Application in Research and Industry. *Proceedings of 17th IEEE International Requirements Engineering Conference (RE'09)*, pages 267–272, 2009. 4, 5, 20, 21, 22, 277

[AHKGW15] Elske Ammenwerth, Reinhold Haux, Petra Knaup-Gregori, and Alfred Winter. *IT-Projektmanagement im Gesundheitswesen*. Schattauer, Stuttgart, 2., voll. überarb. und erw. Aufl. edition, 2015. 48, 49

[AP08] David Arnott and Graham Pervan. Eight key issues for the decision support systems discipline. *Decision Support Systems*, 44:657–672, 2008. 42

[Arn08] David Arnott. Personal Decision Support Systems. In Frada Burstein and Clyde Holsapple, editors, *Handbook of Decision Support Systems 2*, chapter 43, pages 127–150. Springer-Verlag, 2008. 37, 43

[Arn10] David Arnott. Senior executive information behaviors and decision support. *Journal of Decision Systems*, 19(4):465–480, 2010. 37

[Arr13] Valerie Arraj. *ITIL®: the basics*. The APM Group and The Stationary Office, 2013. In: . Date visited 18. Dec. 2014. 258

[AvH09] Grigoris Antoniou and Frank van Harmelen. Web Ontology Language: OWL. In *Handbook on Ontologies*, pages 91–110. Springer Berlin / Heidelberg, 2009. 28

[BBJ12] Ilyès Boukhari, Ladjel Bellatreche, and Stéphane Jean. An ontological pivot model to interoperate heterogeneous user requirements. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7610 LNCS(PART 2):344–358, 2012. 186

[BCR94] Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. The Goal Question Metric Approach. *Encyclopedia of Software Engineering*, 2:528—-532, 1994. 10, 79, 99, 154

[BF14] Pierre Bourque and Richard E. Fairley, editors. *Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)*. IEEE Computer Society, Los Alamitos, CA, USA, 3.0 edition, 2014. 4, 260

[Bjø06] Dines Bjørner. *Software Engineering 3. Domains, requirements, and software design*. Texts in Theoretical Computer Science: an EATCS Series. Springer, Berlin; Heidelberg, 2006. 3, 4, 192, 257

[Boe01] Barry W. Boehm. *Software Engineering Economics*, pages 99–150. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. 3, 17

[BTM12] Daniele Barone, Thodoros Topaloglou, and John Mylopoulos. Business Intelligence Modeling in Action: A Hospital Case Study. In Jolita Ralyte, Xavier Franch, Sjaak Brinkkemper, and Stansilaw Wrycza, editors, *Advanced information systems engineering*, pages 502–517, 2012. 66

[C$^+$04] Alan Cooper et al. *The inmates are running the asylum:[Why high-tech products drive us crazy and how to restore the sanity]*. Sams Indianapolis, IN, USA:, 2004. 22

[CBCG10] Verónica Castañeda, Luciana Ballejos, María Laura Caliusco, and María Rosa Galli. The Use of Ontologies in Requirements Engineering. *Global Journal of Researches in Engineering*, 10(6):2–8, 2010. 94, 95, 106, 131, 132

[CMSV09] Philipp Cimiano, Alexander Mädche, Steffen Staab, and Johanna Völker. Ontology Learning. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, pages 245–267. Springer-Verlag Berlin Heidelberg, 2 edition, 2009. 28

[dBD10] Fabian de Bruijn and Hans L. Dekkers. Ambiguity in natural language software requirements: A case study. In Roel Wieringa and Anne Persson, editors, *Requirements Engineering: Foundation for Software Quality: 16th International Working Conference, REFSQ 2010, Essen, Germany, June 30–July 2, 2010. Proceedings*, pages 233–247, Berlin, Heidelberg, 2010. Springer. 17

[DVB$^+$15] Diego Dermeval, Jéssyka Vilela, Ig Ibert Bittencourt, Jaelson Castro, Seiji Isotani, Patrick Brito, and Alan Silva. Applications of Ontologies in Requirements Engineering: A Systematic Review of the Literature. *Requirements Engineering*, 21(4):405–437, November 2015. 5, 28, 101, 102, 110, 117, 130, 132, 192

[EB13] Richard Ellis-Braithwaite. Analysing the assumed benefits of software requirements. In Richard et al. Berntsson, editor, *19th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2013). Proceedings of the REFSQ 2013 Workshops CreaRE, IWSPM, and RePriCo, the REFSQ 2013 Empirical Track (Empirical Live Experiment and Empirical Re*, volume ICB 56 of *ICB-Research Reports*, pages 207–214, Essen, Germany, 2013. ICB Universität Duisburg Essen. 22, 258

[EK08] Khaled El Emam and Günes A. Koru. A replicated survey of IT software project failures. *IEEE Software*, 25(5):84–90, 2008. 3, 17

[Eva04] Eric Evans. *Domain-driven design*. Addison-Wesley, Boston MA, 2004. 4, 257

[FLGPSF13] Mariano Fernández-López, Asunción Gómez-Pérez, and Mari Carmen Suárez-Figueroa. Methodological guidelines for reusing general ontologies. *Data and Knowledge Engineering*, 86:242–275, 2013. 186

[Gam08] Erich Gamma. *Design patterns*. Addison-Wesley professional computing series. Addison-Wesley, Boston, MA; Munich, 36. print. edition, 2008. 98, 118

[Gao13] Shijia Gao. Mobile decision support systems research: a literature analysis. *Journal of Decision Systems*, 22(1):10–27, 2013. 37

[GFG12] Renata Guizzardi, Xavier Franch, and Giancarlo Guizzardi. Applying a Foundational Ontology to Analyze Means-end Links in the i* Framework. *Proceedings - International Conference on Research Challenges in Information Science*, 2012. 258

[GH06] Alexandre Gachet and Pius Haettenschwiler. Development processes of intelligent decision-making support systems: review and perspective. In *Intelligent Decision-making Support Systems*, pages 97–121. Springer, 2006. 37, 38

[GOS09] Nicola Guarino, Daniel Oberle, and Steffen Staab. What Is an Ontology? In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, pages 1–17. Springer-Verlag Berlin Heidelberg, 2009. 27, 28

[GPFLC04] Asunción Gómez-Pérez, Mariano Fernández-López, and Oscar Corcho. *Ontological Engineering*. Advanced information and knowledge processing. Springer, London, 2004. 28

[GRG08] Paolo Giorgini, Stefano Rizzi, and Maddalena Garzetti. GRAnD: A goal-oriented approach to requirement analysis in data warehouses. *Decision Support Systems*, 45(1):4–21, 2008. 66

[GRR16] Stephany García, Oscar Romero, and Ruth Raventós. DSS from an RE Perspective: A systematic mapping. *Journal of Systems and Software*, 117:488–507, 2016. 38, 43, 44, 65

[Gua98] Nicola Guarino. Formal Ontology and Information Systems. In *Proceedings of the first international conference on Formal Ontology in Information Systems*, pages 3–15, June 1998. 95

[HBS09] Alice Hertel, Jeen Broekstra, and Heiner Stuckenschmidt. RDF Storage and Retrieval Systems. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, pages 489–508. Springer-Verlag Berlin Heidelberg, 2009. 28, 259

[HHBA11] Alexander Hoerbst, Werner O. Hackl, Roland Blomer, and Elske Ammenwerth. The status of IT service management in health care - ITIL® in selected European countries. *BMC Medical Informatics and Decision Making*, page 76, 2011. 258

[HHPC12] Bryan Hosack, Dianne Hall, Daivd Paradice, and James Courtney. A Look Toward the Future: Decision Support Systems Research is Alive and Well. *Journal of the Association for Information*, 13(Special Issue):315–340, 2012. 37

[HJK$^+$17] K. Höffner, F. Jahn, C. Kücherer, B. Paech, B. Schneider, M. Schöbel, S. Stäubert, and A. Winter. Technical environment for developing the SNIK ontology of information management in hospitals. In *Studies in Health Technology and Informatics*, volume 243, 2017. 32

[HMPR04] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. Design Science in Information Systems Research. *MIS Quarterly*, 28(1):75–105, 2004. 8

[Hol08a] Clyde W. Holsapple. Decisions and Knowledge. In Frada Burstein, editor, *Handbook on Decision Support Systems 1*, chapter 2, pages 21–53. Springer, 2008. 39, 40, 41, 277

[Hol08b] Clyde W. Holsapple. DSS Architecture and Types. In Frada Burstein, editor, *Handbook on Decision Support Systems 1*, chapter 9, page 854. Springer, 2008. 41, 42

[HP09] Pascal Hitzler and Bijan Parsia. Ontologies and Rules. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, pages 111–132. Springer-Verlag Berlin Heidelberg, 2009. 27, 258

[HRS14] Lutz J. Heinrich, René Riedl, and Dirk Stelzer. *Informationsmanagement*. De Gruyter Oldenbourg, Berlin [u.a.], 11., vollst. überarb. Aufl. edition, 2014. 29

[HS06] Hans-Jörg Happel and Stefan Seedorf. Applications of Ontologies in Software Engineering. In *International workshop on semantic web enabled software engineering (SWESE'06)*, pages 1–14, 2006. 94

[HW01] Clyde W Holsapple and Andrew B Whinston. Decision support systems: a knowledge-based approach. *Studies in Informatics and Control*, 10(1):73–76, 2001. 40

[Int11] International Organization for Standardization/ International Electrotechnical Commission. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models ISO/IEC 25010:2011(E), 2011. 27

[IRE15] IREB. Certified Professional for Requirements Engineering – Foundation Level. Technical report, International Requirements Engineering Board (IREB) e.V., Karlsruhe, Germany, 2015. 16

[ISO11] ISO; IEC; IEEE. Systems and software engineering — Life cycle processes — Requirements engineering (ISO/IEC/IEEE 29148), 2011. 15, 16, 17, 18, 93, 99, 154, 155, 186, 255

[JSK$^+$16] Franziska Jahn, Michael Schaaf, Christian Kahmann, Kais Tahar, Christian Kücherer, Barbara Paech, and Alfred Winter. An ontology-based scenario for teaching the management of health information systems. In *HEC Health Exploring Complexity: An Interdisciplinariy Systems Approach*, 2016. 32

[JSPW14] Franziska Jahn, Michael Schaaf, Barbara Paech, and Alfred Winter. Ein semantisches Netz des Informationsmanagements im Krankenhaus. In *Informatik 2014*, volume LNI P-232, pages 1491–1498, 2014. 31, 32, 95, 257

[Jun15] Manuel Jung. *Requirements Engineering von Informationsmanagement-Informationssystemen im Krankenhaus*. Master thesis, Heidelberg University, 2015. 48, 51

[KC07] Barbara A. Kitchenham and Stuart Charters. Guidelines for Performing Systematic Literature Reviews in Software Engineering (Version 2.3). Technical Report EBSE 2007-001, Keele University; University of Durham, Keele, Staffs, UK; Durham, UK, 2007. 7, 97, 131

[KCH$^+$90] Kyo Chul Kang, Sholom G. Cohen, James A. Hess, William E. Novak, and A. Spencer Peterson. Feasibility Study Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report November, Software Engineering Institute, Carnegie Mellon University, 1990. 114

[KJJ+15] Christian Kücherer, Manuel Jung, Franziska Jahn, Michael Schaaf, Kais Tahar, Barbara Paech, and Alfred Winter. System Analysis of Information Management. In Douglas Cunningham, Petra Hofstedt, Klaus Meer, and Ingo Schmitt, editors, *INFORMATIK 2015*, Bonn, 2015. Gesellschaft für Informatik. 11, 48

[KLS+16] Christian Kücherer, Jan David Liebe, Michael Schaaf, Johannes Thye, Alfred Winter, and Franziska Jahn. The Status Quo of Information Management in Hospitals - Results of an Online Survey. In H.C. Mayr and M. Pinzger, editors, *INFORMATIK 2016*, Bonn, 2016. 11, 54

[KP17] Christian Kücherer and Barbara Paech. A Task-oriented Requirements Engineering Method for Personal Decision Support Systems - A Case Study. In *19th International Conference on Enterprise Information Systems (ICEIS 2017)*, 2017. 11

[Krc15] Helmut Krcmar. *Informationsmanagement*. Springer Gabler, Berlin; Heidelberg, 6., überarb. Aufl. edition, 2015. 29

[Küc17] Christian Kücherer. Use of Domain Ontologies to Improve Requirements Quality. In *Joint Proceedings of REFSQ-2017 Workshops, Doctoral Symposium, Research Method Track, and Poster Track co-located with the 23nd International Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2017)*, Essen, Germany, 2017. CEUR-WS.org. 11

[Lap05] Alexei Lapouchnian. Goal-Oriented Requirements Engineering : An Overview of the Current Research by. *Requirements Engineering*, 8(3):32, 2005. 258

[Lau02] Soren Lauesen. *Software requirements*. Addison-Wesley, London ; Munich, 2002. 23, 257

[Lau03] Soren Lauesen. Task descriptions as functional requirements. *Software, IEEE*, 20(2):58–65, 2003. 23

[Lau05] Soren Lauesen. *User interface design*. Pearson/Addison-Wesley, Harlow; Munich, 2005. 21, 23, 26, 277

[LH01] Soren Lauesen and M B Harning. Virtual Windows: Linking User Tasks, Data Models, and Interface Design. *IEEE Software*, 18(4):67–75, 2001. 26

[Liu10] Chi Lun Liu. CDADE: Conflict Detector in Activity Diagram Evolution Based on Speech Act and Ontology. *Knowledge-based Systems*, 23(6):536–546, 2010. 103

[LKH08] Jan Marco Leimeister, Sebastian Klapdor, and Christian Hörmann. *IT-Management in deutschen Krankenhäusern: eine empirische Studie unter IT-Entscheidungsträgern*. BoD–Books on Demand, 2008. 54

[MHS05] Marjan Mernik, Jan Heering, and Anthony M. Sloane. When and How to Develop Domain-Specific Languages. *ACM Computing Surveys (CSUR)*, 37(4):316–344, 2005. 5

[MLP08] Diana Maynard, Yaoyong Li, and Wim Peters. NLP Techniques for Term Extraction and Ontology Population. In *Ontology Learning and Population*. IOS Press, 2008. 29

[MSM08] Marcus Messerschmidt, Peter Schülein, and Martin Murnleitner. Der Wertbeitrag der IT zum Unternehmenserfolg. Technical report, PricewaterhouseCoopers AG WPG, Stuttgart, Stuttgart, Germany, 2008. 54

[NE00] Bashar Nuseibeh and Steve Easterbrook. Requirements engineering: a roadmap. In *22nd International Conference on on Software Engineering, Future of Software Engineering Track, ICSE 2000*, pages 35–46, Limerick, Ireland, 2000. ACM Press. 183

[Obj15] Object Management Group. OMG Unified Modeling Language TM (OMG UML), v.2.5. Technical Report 2.5, Object Management Group, 2015. 16

[OSS+10] Inah Omoronyia, Guttorm Sindre, Tor Stålhane, Stefan Biffl, Thomas Moser, and Wikan Sunindyo. A domain ontology building process for guiding requirements elicitation. In Roel Wieringa and Anne Persson, editors, *Requirements Engineering: Foundation for Software Quality. REFSQ 2010.*, volume 6182 LNCS, pages 188–202. Springer, 2010. 186

[PA05] Graham Pervan and David Arnott. A critical analysis of Decision Support Systems research. *Journal of Information Technology*, 20(2):67–87, 2005. 41, 42

[Pae00] Barbara Paech. *Aufgabenorientierte Softwareentwicklung*. Springer, Berlin; Heidelberg, 2000. 22

[Pan09] Jeff Z. Pan. Resource Description Framework. In *Handbook on Ontologies*, pages 71–90. Springer Berlin / Heidelberg, 2009. 28

[PHK+13] Erik Jan Philippo, Werner Heijstek, Bas Kruiswijk, Michel R. V. Chaudron, and Daniel M. Berry. Requirement Ambiguity not as Important as Expected - Results of an Empirical Evaluation. In J. Doerr and A. L. Opdahl, editors, *19th*

International Conference on Requirements Engineering: Foundation for Software Quality (REFSQ' 13), volume LNCS 7830 of *Lecture Notes in Computer Science*, pages 65–79, Essen, Germany, 2013. Springer. 17

[PJR+14] Alireza Pourshahid, Iman Johari, Gregory Richards, Daniel Amyot, and Okhaide S. Akhigbe. A goal-oriented, business intelligence-supported decision-making methodology. *Decision Analytics Journal*, 9(1):1–36, 2014. 66

[PK04] Barbara Paech and Kirstin Kohler. Task-Driven Requirements in Object-Oriented Development. In Julio Cesar Sampaio do Prado Leite and Jorge Horacio Doorn, editors, *Perspectives on Software Requirements*, volume 753, chapter 3, pages 45–67. Springer US, Boston, MA, 2004. 5, 20, 21, 22, 24, 51, 206, 277, 279

[Poh16] Klaus Pohl. *Requirements engineering fundamentals: a study guide for the certified professional for requirements engineering exam-foundation level-IREB compliant*. Rocky Nook, Inc., 2016. 3, 16, 17

[Pow02] Daniel J. Power. *Decision Support Systems: Concepts and Resources for Managers*. Quorum Books division Greenwood Publishing, 2002. 42

[Pow07] Daniel J. Power. What are the features of a data-driven dss? Technical report, DSS News, February 2007. Modified October 8, 2010. 44

[Pow08a] Daniel J. Power. Decision Support Systems: A Historical Overview. In Frada Burstein, editor, *Handbook on Decision Support Systems 1*, chapter 7, page 854. Springer, 2008. 42

[Pow08b] Daniel J. Power. Understanding Data-Driven Decision Support Systems. *Information Systems Management*, 25(2):149–154, 2008. 42, 44

[PR15] Klaus Pohl and Chris Rupp. *Requirements engineering fundamentals*. Rocky Nook, Santa Barbara, CA, 2nd edition (online-ausg.) edition, 2015. 16, 99

[RBJ04] James Rumbaugh, Grady Booch, and Ivar Jacobson. *The Unified Modeling Language Reference Manual*. Addison-Wesley, Boston, MA, 2nd edition, 2004. 119

[RH09] Per Runeson and Martin Höst. Guidelines for Conducting and Reporting Case Study Research in Software Engineering. *Empirical Softw. Eng.*, 14(2):131–164, December 2009. 63, 87, 131, 132, 151, 171

[Rod17] Ewald Rode. *Ontologieunterstützung für Tasks und Subtasks in JIRA*. Bachelor thesis, Heidelberg University, 2017. 173, 187

[Sax91] KBC Saxena. Decision support engineering: a DSS development methodology. In *Proceedings of the 24th Annual Hawaii International Conference on System Sciences (HICSS '91)*, volume 3, pages 98–107, 1991. 37

[SBF98] Rudi Studer, V. Richard Benjamins, and Dieter Fensel. Knowledge Engineering: Principles and Methods. *Data & Knowledge Engineering*, 25(1-2):161–197, 1998. 5, 27, 28, 94, 95, 259

[SG09] Camille Salinesi and Ines Gam. How specific should requirements engineering be in the context of decision information systems? *Proceedings of the 2009 3rd International Conference on Research Challenges in Information Science, RCIS 2009*, pages 247–254, 2009. 38, 43, 44, 65

[SJT$^+$15] Michael Schaaf, Franziska Jahn, Kais Tahar, Christian Kücherer, Alfred Winter, and Barbara Paech. Entwicklung und Einsatz einer Domänenontologie des Informationsmanagements im Krankenhaus. In *Informatik 2015*, volume LNI P-246 of *Lecture Notes in Informatics*, Cottbus, Germany, 2015. 32

[SJT$^+$16] Michael Schaaf, Franziska Jahn, Kais Tahara, Christian Kücherer, Alfred Winter, and Barbara Paech. The visualization of large ontologies from a tool point of view. In *HEC Health Exploring Complexity: An Interdisciplinary Systems Approach*, 2016. 32

[SKS07] Masayuki Shibaoka, Haruhiko Kaiya, and Motoshi Saeki. GOORE : Goal-Oriented and Ontology Driven Requirements Elicitation Method. *Advances in Conceptual Modeling – Foundations and Applications*, 4802:225–234, 2007. 22, 258

[Som09] Ian Sommerville. *Software Engineering*. Pearson Education, ninth edit edition, 2009. 4, 15, 27, 257, 258, 259

[Sta87] Charles B. Stabell. Decision support systems: Alternative perspectives and schools. *Decision Support Systems*, 3(3):243–251, 1987. 41

[TB15] Thodoros Topaloglou and Daniele Barone. Lessons from a Hospital Business Intelligence Implementation. *Proceeding from CAiSE 2015*, pages 19–33, 2015. 66

[TSD10] Efraim Turban, Ramesh Sharda, and Dursun Delen. *Decision Support and Business Intelligence Systems*. Prentice Hall Press, 2010. 44

[vL01] Axel van Lamsweerde. Goal-Oriented Requirements Engineering: A Guided Tour. In Frances M. Titsworth, editor, *5th IEEE International Symposium on*

Requirements Engineeirng (ISRE'01), pages 249–261, Toronto, ON, Canada, 2001. IEEE. 259

[VRM16] Joselaine Valaski, Sheila Reinehr, and Andreia Malucelli. Which Roles Ontologies play on Software Requirements Engineering? A Systematic Review. In *Int'l Conf. Software Eng. Research and Practice - SERP'16*, number July, 2016. 28, 130, 191

[WHA+11] Alfred Winter, Reinhold Haux, Elske Ammenwerth, Birgit Brigl, Nils Hellrung, and Franziska Jahn. *Health Information Systems: Architectures and Strategies*. Health Informatics. Springer London, 2. ed. edition, 2011. 29, 30, 47, 258, 259, 260, 277

[Wie14] Roel J Wieringa. *Design Science Methodology for Information Systems and Software Engineering*. Springer Berlin Heidelberg, Heidelberg, Germany; New York, USA; Dordrecht, The Netherlands; London, UK, 2014. 6, 8, 9, 10, 79, 152, 190, 277

[WM12] Roel Wieringa and Ayse Morali. Technical Action Research as a Validation Method in Information Systems Design Science. In Ken Peffers, Marcus Rothenberger, and Bill Kuechler, editors, *Design Science Research in Information Systems. Advances in Theory and Practice: Proceedings of the 7th International Conference on Design Science Research in Information Systems (DESRIST 2012)*, volume LNCS 7286 of *Lecture Notes in Computer Science*, pages 220–238, Las Vegas, Nevada, USA, 2012. Springer Berlin Heidelberg. 8

[YLM07] Eric Yu, Lin Liu, and John Mylopoulos. A Social Ontology for Integrating Security and Software Engineering. *Integrating Security and Software Engineering: Advances and Future Visions*, pages 70–105, 2007. 22

[Yu97] Eric Yu. Towards modelling and reasoning support for early-phase requirements engineering. In *Proceedings of ISRE '97: 3rd IEEE International Symposium on Requirements Engineering*, pages 226–235, Annapolis, MD, USA, 1997. IEEE Comput. Soc. Press. 22

Systematic Literature Review References

[ASL13] Taiseera Hazeem Al Balushi, Pedro R Falcone Sampaio, and Pericles Loucopoulos. Eliciting and Prioritizing Quality Requirements Supported by Ontologies: A Case Study Using the ElicitO Framework and Tool. *Expert Systems*, 30(2):129–151, 2013.

[AVP09] Gabriela N. Aranda, Aurora Vizcaíno, and Mario Piattini. Analyzing Ontology as a Facilitator during Global Requirements Elicitation. *Proceedings - 2009 4th IEEE International Conference on Global Software Engineering, ICGSE 2009*, pages 309–314, 2009.

[BAE^{+}11] Ebrahim Bagheri, Mohsen Asadi, Faezeh Ensan, Dragan Gašević, and Bardia Mohabbati. Bringing Semantics to Feature Models with SAFMDL. *Proceedings of the 2011 Conference of the Center for Advanced Studies on Collaborative Research*, pages 287–300, 2011.

[BBJ12] Ilyès Boukhari, Ladjel Bellatreche, and Stéphane Jean. An Ontological Pivot Model to Interoperate Heterogeneous User Requirements. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7610 LNCS(PART 2):344–358, 2012.

[CAW^{+}16] Erik Casagrande, Edin Arnautovic, Wei Lee Woon, Hatem H Zeineldin, and Davor Svetinovic. Semiautomatic System Domain Data Analysis : A Smart Grid Feasibility Case Study. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, PP(99):1–11, 2016.

[CBC12] Verónica Castañeda, Luciana Ballejos, and María Laura Caliusco. Improving the Quality of Software Requirements Specifications with Semantic Web Technologies. In Alejandro et al. Oliveros, editor, *Anais do WER12 - Workshop em Engenharia de Requisitos*, number 6, Buenos Aires, Argentina, 2012.

[Chi17] Sridhar Chimalakonda. *A Software Engineering Approach for Design of Educational Technologies*. PhD thesis, International Institute of Information Technology, Hyderabad, India, 2017.

[DNA+08] Mamadou H. Diallo, Leila Naslavsky, Thomas A. Alspaugh, Hadar Ziv, and Debra J. Richardson. Toward architecture evaluation through ontology-based requirements-level scenarios. In Rogério et al. de Lemos, editor, *Architecting Dependable Systems V*, volume 5829 LNCS, pages 225–247. Springer Berlin Heidelberg, 2008.

[DO09] Dang Viet Dzung and Atsushi Ohnishi. Improvement of Quality of Software Requirements with Requirements Ontology. *Proceedings - International Conference on Quality Software*, pages 284–289, 2009.

[DO12] Dang Viet Dzung and Atsushi Ohnishi. A Verification Method of Elicited Software Requirements using Requirements Ontology. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, 1:553–558, 2012.

[DRXD10] Ester J C De Lima, José A. Rodrigues Nt., Geraldo B. Xexéo, and Jano M. De Souza. ARARA - A Collaborative Tool to Requirement Change Awarenes. *Proceedings of the 2010 14th International Conference on Computer Supported Cooperative Work in Design, CSCWD 2010*, pages 134–139, 2010.

[DSM12] Olawande Daramola, Guttorm Sindre, and Thomas Moser. Ontology-Based Support for Security Requirements Specification Process. In Pilar et al. Herrero, editor, *On the Move to Meaningful Internet Systems: OTM 2012, Rome, Italy, September 10-14, 2012. Proceedings*, pages 194–206, Berlin, Heidelberg, 2012. Springer.

[DSOS13] O. Daramola, Tor Stålhane, I Omoronyia, and G. Sindre. Using Ontologies and Machine Learning for Hazard Identification and Safety Analysis. In Walid Maalej and Anil Kumar Thurimella, editors, *Managing Requirements Knowledge*, number I, chapter 6, pages 117–141. Springer, Berlin, Heidelberg, 2013.

[DTR16] Ashish Kumar Dwivedi, Anand Tirkey, and Santanu Kumar Rath. An Ontology Based Approach for Formal Modeling of Structural Design Patterns. In *Ninth International Conference on Contemporary Computing (IC3), 2016*, 2016.

[FMK+11] Stefan Farfeleder, Thomas Moser, Andreas Krall, Tor Ståalhane, Inah Omoronyia, and Herbert Zojer. Ontology-Driven Guidance for Requirements Elicitation. In Grigoris et al. Antoniou, editor, *ESWC 2011: Proceedings of the 8th Extended Semantic Web Conference on The Semanic Web: Research and Applications - Volume Part II*, volume LNCS 6644 of *Lecture Notes in Computer Science*, pages 212–226, Heraklion, Crete, Greece, 2011. Springer Berlin Heidelberg.

[GA13] Smita Ghaisas and Nirav Ajmeri. Knowledge-Assisted Ontology-based Requirements Evolution. In Walid Maalej and Anil Kumar Thurimella, editors, *Managing Requirements Knowledge*, number 1, pages 143–167. Springer Berlin Heidelberg, 2013.

[GEPP08] Frederik Gailly, Sergio España, Geert Poels, and Oscar Pastor. Integrating Business Domain Ontologies with Early Requirements Modelling. In *Advances in Conceptual Modeling - Challenges and Opportunities*, volume 5232, pages 282–291. Springer, Berlin, Heidelberg, 2008.

[HIB$^+$17] Olavo Holanda, Seiji Isotani, Ig I. Bittencourt, Diego Dermeval, and Williams Alcantara. An Object Triple Mapping System Supporting Detached Objects: A Performance and Memory Usage Empirical Comparison. *Engineering Applications of Artificial Intelligence*, 62(May 2016):234–251, 2017.

[KGRS10] Leonid Kof, Ricardo Gacitua, Mark Rouncefield, and Pete Sawyer. Ontology and Model Alignment as a Means for Requirements Validation. *Proceedings - 2010 IEEE 4th International Conference on Semantic Computing, ICSC 2010*, pages 46–51, 2010.

[KHKS08] Motohiro Kitamura, Ryo Hasegawa, Haruhiko Kaiya, and Motoshi Saeki. *A Supporting Tool for Requirements Elicitation Using a Domain Ontology*, chapter A Supporti, pages 128–140. Springer, Berlin, Heidelberg, 2008.

[KJL09] Petr Kroha, Robert Janetzko, and José Emilio Labra. Ontologies in Checking for Inconsistency of Requirements Specification. *3rd International Conference on Advances in Semantic Processing - SEMAPRO 2009*, pages 32–37, 2009.

[KO10] M.a Kossmann and M.b Odeh. Ontology-driven Requirements Engineering - A Case Study of ontoREM in the Aerospace Context. *20th Annual International Symposium of the International Council on Systems Engineering, INCOSE 2010*, 2:1000–1012, 2010.

[KSY$^+$10] Haruhiko Kaiya, Yuutarou Shimizu, Hirotaka Yasui, Kenji Kaijiri, and Motoshi Saeki. Enhancing Domain Knowledge for Requirements Elicitation with Web Mining. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, pages 3–12, 2010.

[LAC08] Claudia Lopez, Hernan Astudillo, and Luiz Cysneiros. Semantic-aided Interactive Identification of Reusable NFR Knowledge Fragments. In Robert Meersman and Zahir Tari, editors, *On the move to meaningful Internet systems: OTM 2008*, pages 324–333, 2008.

[Liu16] Chi Lun Liu. CDNFRE: Conflict Detector in Non-functional Requirement Evolution Based on Ontologies. *Computer Standards and Interfaces*, 47:62–76, 2016.

[LJXL11] Ge Li, Zhi Jin, Yan Xu, and Yangyang Lu. An Engineerable Ontology Based Approach for Requirements Elicitation in Process Centered Problem Domain. In Hui Xiong; and W. B. Lee, editors, *Knowledge science, engineering and management*, pages 208–220. Springer Berlin / Heidelberg, 2011.

[LSH16] Jörg Leukel, Vijayam Sugumaran, and Marvin Hubl. The Role of Application Domain Knowledge in Using OWL DL Diagrams: A Study of Inference and Problem-Solving Tasks. *International Conference on Information Systems*, page 16, 2016.

[LWZX07] Zong-yong Li, Zhi-xu Wang, Ai-hui Zhang, and Yong Xu. The Domain Ontology and Domain Rules based Requirements Model Checking. *International Journal of Software Engineering and Its Applications*, 1(1):89–100, 2007.

[LY12] Chi Lun Liu and Heng Li Yang. Applying Ontology-based Blog to Detect Information System Post-development Change Requests Conflicts. *Information Systems Frontiers*, 14(5):1019–1032, 2012.

[MC15] Anas Mahmoud and Doris Carver. Exploiting Online Human Knowledge in Requirements Engineering. *2015 IEEE 23rd International Requirements Engineering Conference, RE 2015 - Proceedings*, pages 262–267, 2015.

[MRPG15] Itzel Morales-Ramirez, Anna Perini, and Renata S.S. Guizzardi. An Ontology of Online User Feedback in Software Engineering. *Applied Ontology*, 10(3-4):297–330, 2015.

[NGA16] Tuong Huan Nguyen, John C. Grundy, and Mohamed Almorsy. Ontology-based Automated Support for Goal-use Case Model Analysis. *Software Quality Journal*, 24(3):635–673, 2016.

[NVLG14] Tuong Huan Nguyen, Bao Quoc Vo, Markus Lumpe, and John Grundy. KBRE: A Framework for Knowledge-based Requirements Engineering. *Software Quality Journal*, 22(1):87–119, 2014.

[OSJ12] Janis Osis, Armands Slihte, and Asnate Jansone. Using Use Cases for Domain Modeling. In *Proceedings of the 7th International Conference on Evaluation of Novel Approaches to Software Engineering*, pages 224–231, 2012.

[OSJM+16] Karla Olmos-Sánchez, Cd Juárez, Chih Mex, Jorge Rodas-Osollo, Cd Juárez, and Chih Mex. Generation of an OWL Ontology from a Knowledge Domain Extended Lexicon. In *Regional Consortium for Computing Sciences and Foundation*, 2016.

[PDC+11] Paulo F. Pires, Flávia C. Delicato, Raphael Cóbe, Thais Batista, Joseph G. Davis, and Joo Hee Song. Integrating Ontologies, Model Driven, and CNL in a Multi-viewed Approach for Requirements Engineering. *Requirements Engineering*, 16(2):133–160, 2011.

[Pol09] Jantima Polpinij. An Ontology-based Text Processing Approach for Simplifying Ambiguity of Requirement Specifications. *2009 IEEE Asia-Pacific Services Computing Conference, APSCC 2009*, pages 219–226, 2009.

[PP15] Paulo Afonso Parreira and Rosângela Dellosso Penteado. *CrossCutting Concerns Identification Supported by Ontologies: A Preliminary Study*, pages 385–407. Springer International Publishing, Cham, 2015.

[RB09] Thomas Riechert and Thorsten Berger. Leveraging Semantic Data Wikis for Distributed Requirements Elicitation. *Proceedings - International Conference on Software Engineering*, pages 7–13, 2009.

[SKS07] Masayuki Shibaoka, Haruhiko Kaiya, and Motoshi Saeki. GOORE: Goal-Oriented and Ontology Driven Requirements Elicitation Method. *Advances in Conceptual Modeling - Foundations and Applications*, 4802:225–234, 2007.

[SSWM13] Amina Souag, Camille Salinesi, Isabelle Wattiau, and Haris Mouratidis. Using Security and Domain Ontologies for Security Requirements Analysis. *Proceedings - International Computer Software and Applications Conference*, pages 101–107, 2013.

[SVG+16] Monique Soares, Jéssyka Vilela, Gabriela Guedes, Carla Silva, and Jaelson Castro. Core Ontology to Aid the Goal Oriented Specification for Self-Adaptive System. In *New Advances in Information Systems and Technologies*, pages 609–618. Springer, 2016.

[YT15] Xiaobu Yuan and Shubhrendu Tripathi. Combining Ontologies for Requirements Elicitation. *5th International Model-Driven Requirements Engineering Workshop, MoDRE 2015 - Proceedings*, pages 31–35, 2015.

[ZDCS16] Christoforos Zolotas, Themistoklis Diamantopoulos, Kyriakos C. Chatzidimitriou, and Andreas L. Symeonidis. From Requirements to Source Code: a Model-Driven Engineering Approach for RESTful Web Services. *Automated Software Engineering*, 2016.

List of Figures

List of Tables

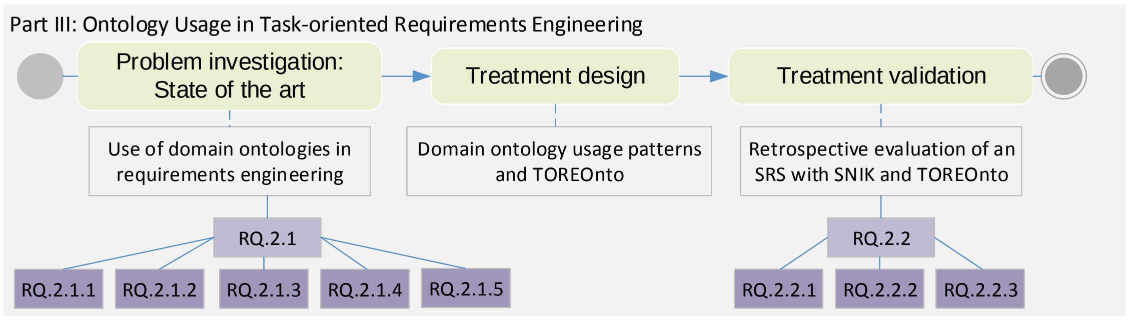
Addendum to Dissertation

Title: Domain-specific Adaptation of Requirements Engineering Methods
Author: Christian Kücherer

Despite careful preparation of the work, some minor mistakes were identified after the dissertation has been printed. Please find below the corrections:

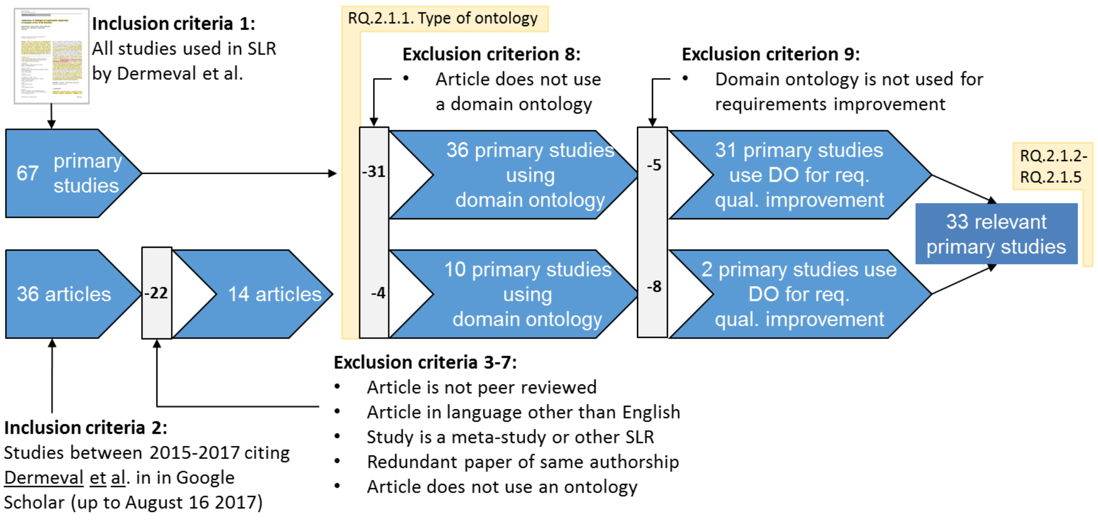
Chapter 8

Figure 8.1: Overview of design science approach and RQs for Part III: In the original image RQ.2.1.5 was missing. The correct figure below contains RQ.2.1.5


Part III: Ontology Usage in Task-oriented Requirements Engineering
Problem investigation: State of the art
Treatment design
Treatment validation
Use of domain ontologies in requirements engineering
Domain ontology usage patterns and TOREOnto
Retrospective evaluation of an SRS with SNIK and TOREOnto
RQ.2.1
RQ.2.2
RQ.2.1.1
RQ.2.1.2
RQ.2.1.3
RQ.2.1.4
RQ.2.1.5
RQ.2.2.1
RQ.2.2.2
RQ.2.2.3

Chapter 9

Figure 9.2 Primary study selection flowchart: In the original picture wrong references to research questions were used. Instead of RQ1-RQ4, the references RQ.2.1.1 – RQ.2.1.5 as is given below:


Inclusion criteria 1:
All studies used in SLR by Dermeval et al.
RQ.2.1.1. Type of ontology
Exclusion criterion 8:
• Article does not use a domain ontology
Exclusion criterion 9:
• Domain ontology is not used for requirements improvement
67 primary studies
-31
36 primary studies using domain ontology
-5
31 primary studies use DO for req. qual. improvement
RQ.2.1.2-RQ.2.1.5
33 relevant primary studies
36 articles
-22
14 articles
-4
10 primary studies using domain ontology
-8
2 primary studies use DO for req. qual. improvement
Exclusion criteria 3-7:
• Article is not peer reviewed
• Article in language other than English
• Study is a meta-study or other SLR
• Redundant paper of same authorship
• Article does not use an ontology
Inclusion criteria 2:
Studies between 2015-2017 citing Dermeval et al. in in Google Scholar (up to August 16 2017)

References

References [Poh16] and [PR15] are double entries, which however does not affect the contents or statements in the thesis. The correct reference is:

[PR15] Klaus Pohl and Chris Rupp. Requirements engineering fundamentals. **A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB compliant**, Rocky Nook, Santa Barbara, CA, 2nd edition (online-edition), 2015.

Heidelberg, 26.2.2018 ________________________________