

Crossing Disciplinary Borders to Improve Requirements Communication



Anne Hess

Abstract Software requirements specifications (SRS) serve as an important source of information for a variety of roles involved in software engineering (SE) projects. This situation poses a challenge to requirements engineers: Different information needs have to be addressed, which are strongly dependent on the particular role(s) that SRS stakeholders have within a project. This chapter summarizes the contributions of a thesis that aimed to address and reduce role-specific defects in SRS that negatively influence the efficient usage and acceptance of these documents. To achieve this goal, we collected empirical data about role-specific information needs in a series of empirical studies that served as a baseline for a secondary analysis toward the definition of role-specific views. Moreover, we realized a proof-of-concept implementation that is capable of generating role-specific views on SRS. The results of a case study revealed that role-specific views have the potential to efficiently support SRS consumers during the analysis of a given SRS. Besides conducting further empirical studies in industry, future work aims to foster cross-disciplinary collaboration and requirements communication, especially in agile teams. Thereby, we are exploring synergy potential with best practices from non-SE disciplines.

1 Introduction

Software development is a cooperative and creative process that requires stakeholders from various software engineering (SE) disciplines to collaborate, exchange information, and coordinate their tasks and efforts [1, 2]. In order to make this collaboration successful, a shared understanding of the functional and nonfunctional requirements of a software system is therefore required [3, 4].

A. Hess (✉)
Fraunhofer IESE, Kaiserslautern, Germany
e-mail: Anne.Hess@iese.fraunhofer.de

© The Author(s) 2022
M. Felderer et al. (eds.), *Ernst Denert Award for Software Engineering 2020*,
https://doi.org/10.1007/978-3-030-83128-8_7

A key discipline that supports the detailed analysis and communication of these requirements is requirements engineering (RE) – a systematic and disciplined approach to the elicitation, analysis, specification, and management of requirements [5]. The results of these RE activities are typically documented in various requirements artifacts, which are consolidated and structured in software requirements specifications (SRS). These SRS finally serve as an important information basis for various people involved in SE disciplines such as architecture design, interaction/user interface design, and software testing.

Despite their importance, we have observed that the acceptance of SRS is often limited and that SRS are often neglected in practice. A more detailed investigation of this observation revealed a number of so-called *role-specific requirements defects* that are not sufficiently addressed by today’s RE approaches (see Table 2, Sect. 2.2). While these defects can basically be traced to insufficient quality of SRS, as well as the requirements artifacts they contain, they are characterized by an interesting fact: Different SRS consumers might experience these defects individually – depending on their particular discipline-related roles, responsibilities, and tasks within an SE project.

In fact, the occurrence of role-specific defects is critical as they negatively influence the efficient usage and analysis of an SRS from the consumer’s viewpoint. This situation can again lead to failures, delays, and frustration in subsequent SE activities and, ultimately, to costly changes and budget or time overruns [6, 10].

Addressing these role-specific defects finally motivated the practical improvement goals of the thesis “Role-Specific Views on SRS: An Empirical Approach” [11], whose contributions are summarized in this chapter. Within the scope of this thesis, we collected empirical data about role-specific information needs in a series of empirical studies, which served as a baseline for a secondary analysis aimed at the definition of role-specific views. Moreover, we realized a proof-of-concept implementation of our envisioned solution that is capable of generating role-specific views on SRS. The results of the case study revealed that role-specific views have the potential to efficiently support SRS consumers during the analysis of a given SRS as well as requirements engineers during specification activities.

Following the thesis work, we continued to further investigate and improve cross-disciplinary collaboration and requirements communication, especially in agile teams. In doing so, we even cross the borders of the field of SE to get inspired by best practices from non-SE disciplines such as criminology, film studies, psychology, and law. The research objectives that we are pursuing in this research context include the identification of synergies with such best practices and their incorporation – as well as adaptation – into innovative methods and techniques to overcome existing challenges.

In the remainder of this chapter, we will first discuss the background and results of the research activities that shaped both the practical and scientific thesis goals in Sect. 2. Then we will provide an overview of the overall solution idea, the research approach, and our thesis contributions in Sect. 3. In Sect. 4, we will share insights and results from our series of empirical studies, which served as a baseline for the

definition of role-specific views on SRS. Section 5 introduces current and future research activities that followed the thesis. The chapter concludes with a summary in Sect. 6.

2 Background and Improvement Goals

This section introduces some background information related to the notion and role of requirements artifacts and shares the results of research activities that were conducted to shape the practical and scientific improvement goals of the thesis.

2.1 Requirements Artifacts

A central activity of RE approaches is the documentation of requirements-related information (covering both functional and nonfunctional system aspects, as well as elements of the usage environment) established and worked out during elicitation, validation, and negotiation activities. This documentation typically results in a set of *requirements artifacts*, which we define as “. . . an object produced or shaped by a human conception or agency in order to externalize requirements-related information” [12].

Requirements artifacts are predominantly written in natural language, both in unstructured common language and in a structured form using templates and forms [13, 14]. Other documentation types include conceptual models (like UML state diagrams or UML activity diagrams) or hybrid documentation combining natural language and conceptual models [5].

In fact, requirements artifacts represent an important factor for the success of a project [15, 16] as they serve as a medium of communication between interdisciplinary project team members with different roles and tasks [17] (see Table 1).

2.2 Practical Improvement Goals

To become a good basis for the various consumers, both the SRS and the requirements artifacts they contain must meet certain quality criteria (such as unambiguity, consistency, or completeness) in order to prevent the occurrence and propagation of requirements defects [7, 17]. Such defects ultimately may lead to major delays, cost overruns, commercial consequences, and even the loss of lives [18].

Despite this major impact that requirements defects have throughout the whole lifecycle of a software product, the data analysis of an industry survey as well as literature reviews revealed that industry is still facing problems and challenges in

Table 1 Consumers of requirements artifacts [5, 15]

Role	Tasks based on requirements artifacts
Project manager	Planning of work packages and milestones for the implementation of the system
Customer	Validation of requirements as part of contract
Software architect	Design of system architecture; analysis of change impacts; validation of requirements as part of quality assurance
Usability expert (Interaction designer)	Design of interactions and user interface; analysis of change impacts; validation of requirements as part of quality assurance
Developer	Implementation of the system; analysis of change impacts
System test engineer	Development of test cases to validate the system; analysis of change impacts; validation of requirements as part of quality assurance
Maintenance engineer	Analysis of defects during system maintenance

addressing quality-related issues when documenting requirements artifacts. As a consequence, the acceptance of SRS is often limited and the documents are often neglected in practice by their consumers.

To better understand and shape the practical problem of the thesis, we conducted expert interviews and performed a literature review with the goal of identifying requirements defects that hinder an efficient use and analysis of SRS from the viewpoint of their readers. This activity resulted in a taxonomy of 14 requirements defect categories [11].

Within the thesis, we focused our research activities on a subset of these defects – so-called *role-specific requirements defects* – which are characterized by an interesting fact: In contrast to “general defects” (such as ambiguity, missing, or lacking traceability), role-specific requirements defects are possibly experienced differently by different readers, depending on the role they have in a project [19]. Hence, we excluded the aforementioned “general defects” from our taxonomy and obtained a list of role-specific defects, which are summarized in Table 2 together with a list of requirements for a suitable solution.

Table 2 Role-specific defects and corresponding solution requirements

Role-specific requirements defect	Requirements for suitable solution
Important information is missing [6–9]	The SRS contains all information that is relevant for a particular role
Superfluous information is specified [6, 8, 9]	The SRS contains only information that is relevant for a particular role
Important information is misplaced [6, 8, 9]	The SRS is appropriately structured so that relevant information can easily be found by a particular role
Requirements artifacts are in an unusable form [6, 8, 9]	Requirements artifacts are specified at the right level of detail using the right notation

As highlighted earlier, the occurrence of role-specific defects negatively influences the efficient usage and analysis of an SRS, as it is time-consuming and/or cognitively difficult for SRS consumers to resolve and reconcile these problems while working with the SRS [6, 8]. This is critical, as it might again lead to failures, delays, and frustration in subsequent SE activities, and ultimately to costly changes and budget or time overruns [6, 10].

Hence, addressing these role-specific problems ultimately constituted the practical improvement goals of the thesis, which are stated as:

Improve the quality of SRS by addressing and reducing role-specific requirements defects in SRS. Resulting from this achievement, we aim to increase the efficiency of SRS analysis and hence the acceptance of an SRS from the viewpoint of its readers.

2.3 Literature Review Activities

To strengthen and isolate the scientific contribution of the thesis, the role-specific defects and corresponding solution requirements in Table 2 were subjected to further literature review activities. The underlying research goal (RG) and related research questions (RQ) of these activities are summarized in Table 3.

Table 3 Research goals and research questions of literature review activities (LRA)

RG _{LRA} : Investigate and analyze the state of the practice/state of the art with regard to existing solutions that particularly address role-specific defects and the corresponding requirements.
RQ _{LRA-1} : To what extent do existing requirements documentation and quality assurance methods, techniques, or guidelines address role-specific defects and corresponding requirements?
RQ _{LRA-2} : To what extent do existing view-based approaches support our overall solution idea of generating role-specific views on requirements documents?

Our literature review activities toward RQ_{LRA-1} revealed several (*normative*) references that highlight and report on good quality characteristics of both individual requirements and SRS [5, 8, 15, 18, 20]. However, we found that all of these references are quite vague when it comes to describing what completeness or minimality means from a role-specific point of view. This issue is also claimed and supported by the work of [17], who argue that some quality criteria (such as completeness) have to be rethought from a quality-in-use perspective. In fact, their ABRE-QM (activity-based RE quality models) concept fits well into the scope of this thesis as our empirical work contributes to the instantiation of their quality models and our tool-based solution represents a concrete implementation of their vision.

Similarly, existing *writing guidelines* [15, 21], *languages, or requirements templates* [5] do not provide any information on the preferred level of detail and notation that is suitable for a particular role. This observation also holds for *standard SRS structures* such as [20, 22], which define an overall structure and corresponding information that should be included in an SRS but do not provide any advice on the suitable level of detail, as well as the right notation that should be used to specify the information in the various sections. Moreover, these references do not address the requirement of “minimality,” i.e., they do not hide irrelevant information from the reader.

Requirements *validation techniques* [5, 23–25] seem to be helpful for identifying and resolving role-specific defects [26] but typically happen after the SRS has been created (rather than during the construction of the SRS). Moreover, these activities easily get tedious and cognitively difficult for the inspectors if they have to search for the relevant information that they have to validate from a dedicated perspective. Also, inspection guidelines and checklists have to explicitly contain and address the role-specific information needs that are to be checked with regard to their fulfillment. And this knowledge is often missing.

There also exist a variety of *formal methods* that are capable of automatically detecting and handling quality-related defects such as ambiguities, inconsistencies, or missing traceability [27–29]. However, to the best of our knowledge, none of these approaches is capable of automatically detecting and resolving the role-specific defects that are the focus of the thesis.

Addressing RQ_{LRA-2}, we identified several *view(point)-based solutions* in the context of RE [30–32]. However, the definition and usage of these viewpoints differ between the various solutions. That is, some viewpoints reflect (orthogonal) viewpoints of the system, or process-specific or organizational viewpoints rather than role-specific views. Moreover, from a technical perspective, existing commercial requirements management tools are capable of automatically generating views on SRS that fit role-specific information needs. This is typically realized by applying filter rules to predefined attributes [5]. However, in order to define and instantiate these attributes, detailed knowledge about role-specific information needs is required, which is not available in the tools but rather has to be obtained individually.

We conclude that many good approaches are available that at least partially address the role-specific defects and related requirements stated in Table 2. However, what all of these approaches are lacking is the explicit provision of detailed knowledge about role-specific information needs. That is, to sufficiently address role-specific defects and corresponding requirements – and hence to ultimately address our practical improvement goals – we claim that detailed empirical knowledge about role-specific information needs is required. It is exactly this empirical knowledge that constitutes the core scientific improvement goal of the thesis, stated as:

Investigate role-specific information needs regarding SRS from the viewpoint of different SRS readers.

3 Solution Idea and Research Approach

The overall solution idea of the thesis is illustrated in Fig. 1.

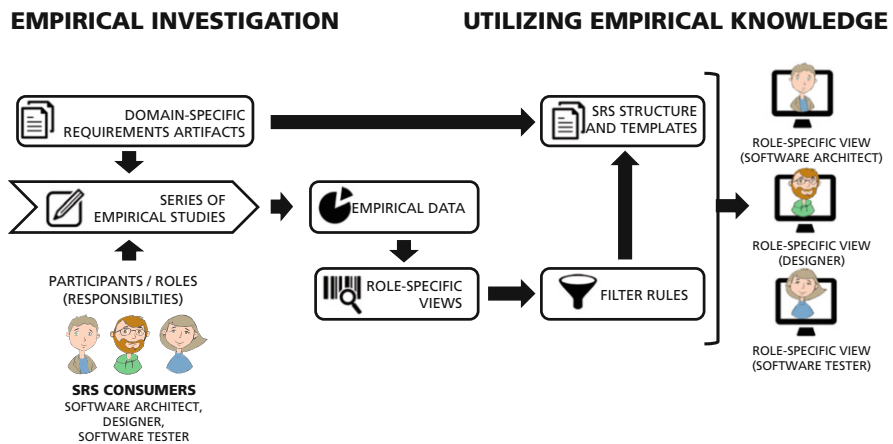





Fig. 1 Solution idea

The approach is based on the idea of conducting *a series of empirical studies* on a given set of *domain-specific requirements artifacts* in order to collect *empirical data* with regard to role-specific information from the perspective of *SRS consumers*. To limit the scope of the thesis, we focused our investigations on three types of SRS consumers: software architects, usability experts in the role of designers, and software testers. We selected these three types because their SE-related activities strongly rely on information documented in SRS. Table 4 briefly summarizes the main responsibilities of the three different roles.

The collected empirical data serves as a baseline for the derivation of *role-specific views* on the investigated requirements artifacts. In this context, a role-specific view comprises a set of requirements artifacts that are relevant for performing role-specific tasks specified at a preferred level of detail using preferred notations.

These role-specific views in turn serve as input to a technical solution that utilizes the empirical knowledge to apply *filter rules* to a given *SRS structure*, as well as *requirements artifact templates*, in order to ultimately *generate role-specific views* on the specification that meet the role-specific information needs.

Table 4 Roles and responsibilities of selected SRS consumers [33–35]

	<p>Software architects are responsible for designing an architectural solution that satisfies expected system functionalities and has a long-lasting impact on major quality attributes of a software-intensive system (like cost, evolution, performance, safety, and security). To achieve this, software architects have to fulfill a variety of complex tasks in order to make, realize, validate, and document design decisions regarding a high-quality software architecture based on a detailed understanding of the expected functionalities and qualities of a software system.</p>
	<p>Designers (usability experts) are responsible for conceptualizing and designing human-system interactions based on documented requirements with the aim of ensuring effectiveness, efficiency, and satisfaction when performing tasks with the system (interaction designer). Moreover, efficient task execution requires suitable organization and structuring of information (information architect) as well as suitable visualization and implementation of the user interface (UI) on the target platform (user interface designer).</p>
	<p>Software testers have different responsibilities that require skills, such as in-depth knowledge about test design and execution methods and techniques and a good understanding of the system to be tested. The main responsibilities of software testers include: (1) reading all relevant documents and understanding what needs to be tested; (2) creating and defining test designs, test processes, test cases, and test data; (3) executing testing as per the defined procedures; and (4) preparing test reports that document procedures as well as test results.</p>

Within the thesis work, we followed a design science research approach as proposed by [36]. Our approach is illustrated in Fig. 2, including the various contributions we elaborated in each phase of the approach.

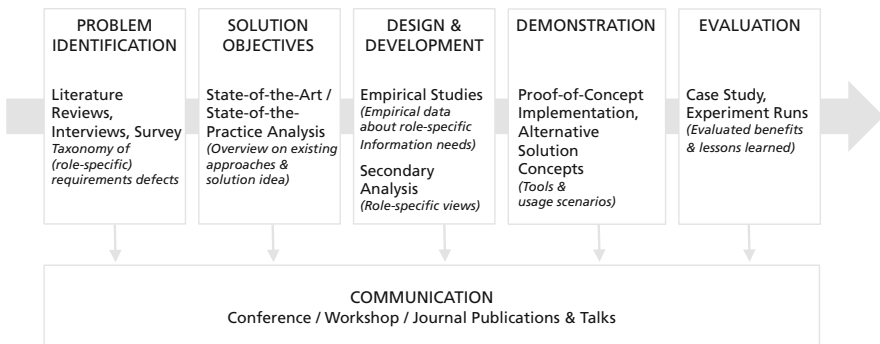


Fig. 2 Research approach and contributions

The previously introduced *taxonomy of (role-specific) requirements defects* (see Sect. 2.2) as well as *the results of the literature review activities* conducted to shape the scientific contributions of the thesis (see Sect. 2.3) were identified and elaborated

in the *problem identification phase*, as well as in the *solution objectives phase* of the research approach.

In the subsequent *design and development phase*, we conducted a series of empirical studies to first collect empirical data with respect to *role-specific information needs*, from which *role-specific views* were then derived and defined in a subsequent secondary analysis. A more detailed discussion of the empirical work can be found in Sect. 4.

In the *demonstration phase*, we realized a *proof-of-concept implementation* (and *alternative solution concepts*) that is capable of utilizing the gained empirical knowledge about role-specific views to support *typical usage scenarios* of SRS both from the consumers’ and from the requirements engineers’ point of view [37].

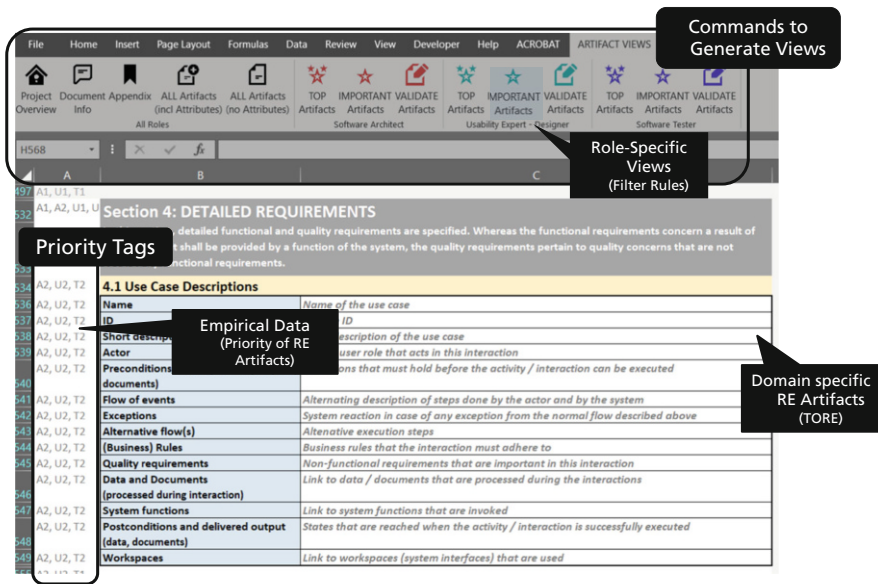


Fig. 3 Proof-of-concept implementation

This technical solution (depicted in Fig. 3) was implemented as an extension of Microsoft Excel®, a common tool often used in practical settings for creating and reading requirements specifications. It comprises an SRS template that defines the overall structure of the SRS as well as a set of domain-specific requirements artifact templates (see Sect. 4.1). The empirical data we collected and analyzed in the series of empirical studies (see Sects. 4.2 and 4.3) was incorporated in the form of priority tags and filter rules (see Sect. 4.3). These filter rules can be applied via commands in the menu bar (see Fig. 3) to generate different views for each of the three roles (software architect, designer, and software tester):

- *TOP Artifacts*: Executing this filter function displays only requirements artifacts and related information details that are of high priority for the corresponding role. Such requirements artifacts include key information for the artifact consumers that is critical for fulfilling their role-specific tasks. Hence, these artifacts and the related information have to be specified timely and precisely.
- *IMPORTANT Artifacts*: Executing this filter function displays requirements artifacts that are rather important, respectively of medium priority. In contrast to high-priority artifacts, these artifacts and the related information are less critical. That is, the artifact stakeholders could also do their tasks based on high-level descriptions of these artifacts.
- *VALIDATE Artifacts*: Executing this filter function displays all requirements artifacts that are of both high and medium priority. This artifact view is intended to support validation activities following perspective-based techniques to assure SRS quality.

Besides these role-specific views, our implementation also offers the possibility to display all requirements artifacts and related information details (both with and without meta-information). Additional views such as “Project Overview,” “Document Info,” and “Appendix” are intended to further reduce the complexity of the information displayed in the SRS.

Finally, within the *evaluation phase*, the proof-of-concept implementation was applied in a *case study* within the context of a team-based software engineering project at the University of Kaiserslautern. There we investigated several hypotheses with regard to the practical and scientific improvement goals of the thesis. Even though the validity of the case study is limited, the case study revealed promising results. Among the conclusions and lessons learned from the case study, we found that role-specific views have the potential to efficiently support SRS consumers during the analysis of SRS, that the views reflect role-specific information needs, and that they are helpful for supporting the communication and discussion of requirements in software-developing teams. On the other hand, we found that the usefulness of the views strongly depends on the project setting. That is, they are possibly most beneficial in the context of the development of large systems resulting in complex SRS that serve as a major source of requirements-related information.

In addition to the case study, we performed *two trial runs of experimental investigations* with the overall goal of investigating hypotheses that compare the efficiency of SRS analysis with our view-based solution and that of “traditional” SRS. We found that properly designing such experiments is indeed a challenge in terms of ultimately drawing valid conclusions. However, we gained interesting insights and learned lessons from these investigations that will serve as guidelines for future evaluation activities.

We continuously shared and published the results of our research activities with and to research and industry as part of the *communication phase*. The presentations of our results, the feedback we gained from the reviewers, and the fruitful discussions we had at the various conferences and workshops confirmed the relevance of our research for the RE community. The feedback we obtained from

both practitioners and researchers helped us to continuously improve and validate our work.

4 Empirical Studies

This section shares some insights into the series of empirical studies we conducted to address the scientific improvement goal of the thesis, which ultimately served as a baseline for the definition of role-specific views. In fact, we consider the results of these studies as well as the underlying methodological approach as a core contribution of the thesis that can be utilized and transferred into different solutions aiming to improve requirements communication in software development.

4.1 Research Goals and Agenda

The overall *research goal* that we aimed to investigate in the empirical studies was to *identify and characterize requirements-related information that downstream development engineers (SRS consumers) seek in an SRS to satisfy their needs and to accomplish their role-specific tasks.*

As outlined in the background section (Sect. 2), requirements-related information is typically specified in the form of requirements artifacts in SRS. Due to the variety of existing requirements artifacts, and to ensure comparability between the different study results, we focused our investigation on well-known requirements artifacts that are typically created when RE is performed within the information systems domain. In particular, we considered artifacts that are created during the application of the Task-oriented Requirements Engineering Framework – or TORE Framework for short [38]. Due to their general nature, these requirements artifacts and thus the results of our investigations can be easily mapped and transferred to other RE approaches that produce requirements artifacts of this kind.

The investigated artifacts comprised:

- *Descriptions of stakeholders* capturing relevant information and characteristics about stakeholders who are to be supported by or have an influence on the system to be built.
- *Descriptions of project goals* that different stakeholders would like to achieve with the system to be built. The project goals include and refine the vision of the system and are considered the rationale for the system's requirements.
- *Descriptions of as-is situations* that illustrate the execution of the tasks/business processes that are to be supported by a system in the current situation, i.e., without the system to be built.
- *Descriptions of to-be situations* that illustrate the execution of tasks/business processes to be supported in the future by the system.

- *Descriptions of the system context* that define the system's environment (e.g., users, external systems), including an overview of functionalities that the system offers to it.
- *Descriptions of interactions* that describe how the system interacts with entities in its environment (e.g., users, external systems).
- *Descriptions of system functions* that specify the input, internal behavior, and output of system functionalities. In contrast to interactions, system functions represent functionalities that are automatically performed by the system.
- *Descriptions of quality requirements* that specify desired qualities (nonfunctional requirements) of the system to be built.
- *Descriptions of technical constraints* that limit the solution space beyond what is necessary for meeting the requirements.

Considering these requirements artifacts, we further refined the above research goal into three research questions and corresponding metrics, which are summarized and stated in Table 5.

Table 5 Research questions (RQ)

<i>Research Question RQ-1_{EmpStudies}</i>
What are typical requirements artifacts that should be contained in an SRS from the viewpoint of document stakeholders (SRS consumers) in order to accomplish their role-specific tasks?
Metric RQ-1 _{EmpStudies} : Importance level of TORE requirements artifacts
<i>Research Question RQ-2_{EmpStudies}</i>
At what level of detail should relevant requirements artifacts be specified from the viewpoint of document stakeholders (SRS consumers)?
Metric RQ-2 _{EmpStudies} : Relevant description items of TORE requirements artifacts
<i>Research Question RQ-3_{EmpStudies}</i>
Which notation should be used to specify relevant requirements artifacts from the viewpoint of document stakeholders (SRS consumers)?
Metric RQ-3 _{EmpStudies} : Preferred notation for documenting TORE requirements artifacts

In order to investigate the research goals and questions above, we designed and conducted a series of four empirical studies as part of our research agenda. These studies comprised:

- *Study #1: An eye-tracking study* conducted with two software architects and two usability experts in the role of designers to investigate the relevance of TORE artifacts (*RQ-1_{EmpStudies}*) as well as the suitable level of detail (*RQ-2_{EmpStudies}*) and the preferred notations (*RQ-3_{EmpStudies}*).
- *Study #2: A survey study (U-KL)* conducted in two practical software engineering courses with a total of 18 participants in the role of software architects. In this study, we retrospectively evaluated the relevance of TORE requirements artifacts for architecture design activities (*RQ-1_{EmpStudies}*).
- *Study #3: A survey study (MUC)* conducted with ten usability experts during a tutorial session that aimed to investigate the relevance of TORE requirements

artifacts ($RQ-1_{EmpStudies}$), the suitable level of detail ($RQ-2_{EmpStudies}$), and the preferred notations ($RQ-3_{EmpStudies}$) for interaction/UI design activities.

- Study #4: A *survey study* ($FHNW$) conducted in semi-industrial software development projects with a total of 47 participants in the role of software architects, 40 participants in the role of usability experts/designers, and 49 participants in the role of software testers. This study aimed to collect data with regard to the relevance of requirements artifacts for architecture design, interaction/UI design, and testing activities ($RQ-1_{EmpStudies}$).

Detailed descriptions of the design and the procedures of each of these studies can be found in [11]. The elicited raw data of the studies is publicly accessible via [39].

4.2 Analysis of Individual Studies: Empirical Baseline

Even though the setting and the procedures varied between the four different studies, we asked the participants in each of the studies to rate the importance of various requirements artifacts on a 4-point rating scale with the help of a questionnaire. This rating scale was defined as follows:

- 1 = “This requirements artifact is very important for my task”
- 2 = “This requirements artifact is rather important for my task”
- 3 = “This requirements artifact is rather unimportant for my task”
- 4 = “This requirements artifact is very unimportant for my task”

We decided to use this 4-point scale in order to evoke a decision between (very) important and (rather) unimportant.

4.2.1 Data Analysis Strategy: An Example

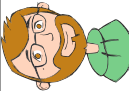
In the following, we would like to share some insights into the data analysis strategies we applied on the questionnaire data based on the example of survey study #4. This study was conducted within the scope of a practical project course offered at the University of Applied Sciences and Arts Northwestern Switzerland ($FHNW$). A first study run ($FHNW_{A-2013}$) was executed during the autumn term of 2013. We repeated the same study in a second run ($FHNW_{S-2014}$) in the subsequent spring term of 2014. A third run ($FHNW_{A-2014}$) was executed during the autumn term of 2014, and the final fourth run ($FHNW_{S-2015}$) in the subsequent spring term of 2015. At the end of a term (i.e., after the students had completed their role-specific tasks), the coach responsible for the requirements engineering phase distributed our questionnaire, which captured the importance of specified requirements artifacts for role-specific tasks on the 4-point scale introduced above.

Table 6 shows an extract of the results of the application of our data analysis strategy to the questionnaire data we collected from the viewpoint of usability experts in the role of designers in the different study runs at $FHNW$.

Table 6 Data analysis results of FHNW study (usability experts/designers)

	<i>Importance of requirements artifacts^a</i>										KW (N = 40)
	FHNW A-2013 (N = 13)	FHNW S-2014 (N = 11)	FHNW A-2014 (N = 10)	FHNW S-2015 (N = 6)	FHNW ALL (N = 40)	OS-WSRT (FHNW ALL)					
	Mdn (M)	Mdn (M)	Mdn (M)	Mdn (M)	Mdn ^b (M)	HMdn	Z	p			
	Min-Max	Min-Max	Min-Max	Min-Max	Min-Max						
Descriptions of stakeholders	2 (2.08) 1-4	1 (2.18) 1-4	1.5 (1.50) 1-2	2 (2.00) 1-3	2*** (1.95) 1-4	3	-4.462 <0.001	0.762			
Descriptions of goals	4 (3.00) 1-4	1 (2.09) 1-4	1 (1.20) 1-2	1.5 (1.50) 1-2	1 (1.62) 1-4 (N = 27)	3	-4.230 <0.001	0.012			
Descriptions of as-is situations	4 (3.08) 1-4	1 (2.09) 1-4	1 (1.20) 1-2	1.5 (1.50) 1-2	1*** (1.63) 1-4 (N = 27)	3	-4.230 <0.001	0.009			
Description of to-be situations	4 (3.00) 1-4	1 (2.09) 1-4	1 (1.20) 1-2	1.5 (1.50) 1-2	1*** (1.63) 1-4 (N = 27)	3	-4.230 <0.001	0.012			

(continued)



Descriptions of system context	4 (3.08) 1-4	1 (2.09) 1-4	2 (1.90) 1-4	1.5 (1.67) 1-3	3* (2.53) 1-4	2	2.729 0.006	0.106
Descriptions of interactions	2 (2.38) 1-4	1 (1.64) 1-4	2 (1.90) 1-4	1 (1.17) 1-2	1*** (1.88) 1-4	3	-4.604 <0.001	0.210
Descriptions of system functions	1 (1.23) 1-2	1 (1.36) 1-4	1 (1.10) 1-2	1 (1.17) 1-2	1*** (1.23) 1-4	3	-5.872 <0.001	0.917
Descriptions of quality reqs.	1 (1.23) 1-2	1 (1.36) 1-4	1 (1.10) 1-2	1 (1.17) 1-2	1*** (1.23) 1-4	3	-5.872 <0.001	0.917
Descriptions of technical constraints	1 (1.46) 1-4	1 (1.36) 1-4	1 (1.10) 1-2	1 (1.17) 1-2	1*** (1.30) 1-4	3	-5.762 <0.001	0.691

^a Response rating scale: 1 = The RA is very important for my task, 2 = The RA is rather important for my task, 3 = The RA is rather unimportant for my task, 4 = The RA is very unimportant for my task

^b One-sample Wilcoxon signed-rank test:

if $Mdn(x) < 3$: H_0 : $H-Mdn(x) = 3$; * $p < 0.05$; ** $p < 0.01$; *** $p < 0.001$

if $Mdn(x) > 2$: H_0 : $H-Mdn(x) = 2$; * $p < 0.05$; ** $p < 0.01$; *** $p < 0.001$

To analyze the questionnaire data, we first calculated *descriptive statistics*, particularly the median (Mdn), sample means (M), minimum (Min), and maximum (Max) values, for each requirements artifact in each study run based on the questionnaire data [39]. As all the data sets were not normally distributed (according to the Shapiro–Wilk test), we applied the *Kruskall–Wallis test* (KW) for independent samples in order to determine possible differences in the medians between the four groups (FHNW_{A-2013}, FHNW_{S-2014}, FHNW_{A-2014}, and FHNW_{S-2015}) for each of the investigated requirements artifacts. For those variables that did not reveal any significant difference in their median, we calculated the aforementioned descriptive statistics on the *consolidated data set* (FHNW_{ALL}). Moreover, we applied the *one-sample Wilcoxon signed-rank test* to check whether the participants shared a meaningful opinion with regard to the relevance of the investigated artifacts. To do so, we verified whether an observed median (Mdn(x)) was equal to hypothetical values $H\text{-Mdn} = 2$ (if $\text{Mdn}(x) > 2$), respectively $H\text{-Mdn} = 3$ (if $\text{Mdn}(x) < 3$). The significance level (p) was set to 0.05 in all tests. We performed all data analysis activities with IBM[®] SPSS[®] PASW Statistics 18.

As highlighted in Table 6, we detected significant differences between the samples FHNW_{A-2013} and FHNW_{A-2014} in the case of descriptions of goals, as-is situations, and to-be situations. Due to this significant difference, we decided to exclude the values of sample FHNW_{A-2013} from the consolidation step. This decision was also underpinned by our assumption that the observed difference between the samples might be attributed to a slight difference in the questionnaire design that we used in the FHNW_{A-2013} study. However, we did not yet discard this sample at this point in our analysis but kept the values of FHNW_{A-2013} separately as input to the secondary analysis (see Sect. 4.3).

4.2.2 Data Interpretation

In order to draw a conclusion with regard to the overall importance level of a certain requirements artifact – and hence to contribute to $RQ\text{-}I_{\text{EmpStudies}}$ – we interpreted the mean values (M) that resulted from the analysis of the descriptive statistics on the consolidated data set. To do so, we applied the following rules:

- If $1.0 \leq M(x) < 1.5$, the RA is very important for role-specific tasks.
- If $1.5 \leq M(x) < 2.5$, the RA is rather important for role-specific tasks.
- If $2.5 \leq M(x) < 3.5$, the RA is rather unimportant for role-specific tasks.
- If $3.5 \leq M(x) \leq 4.0$, the RA is very unimportant for role-specific tasks.

Applying this scheme to the example data presented in Table 6 led to the following conclusions:

From the viewpoint of the participants of the FHNW study in the role of designers, we identified descriptions of system functions and quality requirements as very important requirements artifacts. In order of importance, these were immediately followed by descriptions of goals, as-is situations, to-be situations, interactions, and descriptions of stakeholders, which were all identified as rather important artifacts. Only descriptions of system context were identified as rather unimportant.

All these findings were also underpinned by the application of the one-sample Wilcoxon signed-rank test (see column OS-WSRT, Table 6). This observation proves the existence of a meaningful opinion shared among the participants regarding the importance rating of the investigated requirements artifacts.

We applied the data analysis and interpretation scheme to the questionnaire data of all studies for each role [39]. Ultimately, we consolidated the various interpretations with regard to the importance of the investigated requirements artifacts that we received from these initial data analysis activities.

Table 7 provides an overview of the consolidated findings over all requirements artifacts from each of the investigated viewpoints.

This cross-study comparison of the analysis results revealed interesting findings. *We identified differences between the different studies (and even between different study runs) with respect to the importance rating of the various requirements artifacts.* At this point of our research, it was still unclear whether the observed differences between the different studies are significant or not. This analysis was the subject of the secondary data analysis (see Sect. 4.3) that consolidated the results we gained in the individual studies in order to arrive at final and more fine-grained conclusions with regard to role-specific information needs and implications on role-specific views.

In this fine-grained conclusion, we also aimed to address the following observation: *We identified many requirements artifacts as “rather important” – some of them close to the border to “very important.”* In other words, the range underlying our categorization based on the mean value ($1.5 \leq M(x) < 2.5$) was possibly too broad for the definition of role-specific views.

Apart from the differences between the different studies, *we also observed variances within samples of the individual studies.* This was indeed an interesting observation that motivated future research on factors such as personality or individual reading behavior, which might possibly influence the relevance of requirements artifacts from the viewpoint of their consumers (see Sect. 5).

Table 7 Cross-study comparison

Investigated requirements artifacts	Software architect (N = 67)			Usability expert/designer (N = 52)			Software tester (N = 49)	
	Eye-Tracking	U-KL	FHNW	Eye-Tracking	MUC	FHNW	FHNW	
Descriptions of stakeholders	RI	RI	RU	VI	RI	RI	RU	
Descriptions of goals	VI	RI	RI	RI	RI	VU	RI	RU
Descriptions of as-is situations	RI	VU	RU	RU	RI	VU	RI	RU
Descriptions of to-be situations	RI	RI	RI	RI	VI	VU	RI	RU
Descriptions of system context	RI	RU	RI	RI	N/A	RU	VU	RU
Descriptions of interactions	RI	RI	RI	RI	RI	RI	RI	
Descriptions of system functions	RI	VI	VI	RI	RI	VI	RI	VI
Descriptions of quality requirements	RI	RI	VI	RI	N/A	VI	RI	VI
Descriptions of technical constraints	VI	RI	VI	RU	N/A	VI	RI	VI

Very important (VI)

Rather important (RI)

Rather unimportant (RU)

Very unimportant (VU)

4.3 Secondary Data Analysis: Role-Specific Views

Based on the empirical baseline introduced in the previous section, we performed a secondary data analysis. The overall goal of this secondary data analysis was twofold: First, we aimed to compare and consolidate the data we had obtained in the various empirical studies in order to ultimately arrive at a conclusion with regard to the *overall importance level* of the different TORE requirements artifacts from the viewpoint of software architects, designers, and software testers (RQ-1_{EmpStudies}).

Second, we aimed to link our findings to our envisioned solution idea (see Sect. 3) and interpret the findings in terms of suitable contents of *role-specific views* for each of the three downstream roles. The latter decision explicitly incorporated the results we obtained from investigating RQ-2_{EmpStudies} and RQ-3_{EmpStudies}.

In the following, we will introduce and illustrate our data analysis strategy and interpretation of this secondary data analysis with a concrete example.

4.3.1 Data Analysis Strategy: An Example

The statistical data analysis strategy we followed during the secondary data analysis comprised six analysis steps:

1. For each requirements artifact, we created a table summarizing the final priority ratings of the different samples we derived from the data analysis of the individual studies (see Sect. 4.1).
2. To test whether our data is normally distributed, we applied the *Shapiro–Wilk test*, which led us to assume a non-normal data distribution.
3. Hence, in order to test for possible significant differences between the different study samples, we applied nonparametric tests of independent samples (in particular *Kruskal–Wallis tests*, respectively *Mann–Whitney U tests*) and reported the resulting p -values, which correspond to the asymptotic significance values (for $N \geq 30$) and the exact significance values if $N < 30$.
4. In the case of any significant differences identified in Step 3, we performed a *pairwise comparison* to identify the particular samples that revealed differences and reported the corresponding significance level (p), the observed value (Z), the sample size (N), and the effect size (r).
5. Next, we calculated *descriptive statistics* based on the consolidated sample data sets comprising median (Mdn), mean (M), minimum (Min), and maximum (Max) values and generated a histogram visualizing the frequencies of the importance ratings.
6. Last, we performed a *one-sample Wilcoxon signed-rank test* to check whether a meaningful opinion shared among the participants can be observed with regard to the importance of a particular requirements artifact. To do so, we tested for possible significant differences to a hypothetical median value $H\text{-Mdn}(x) = 2$, respectively $H\text{-Mdn}(x) = 3$. We report the Wilcoxon signed-rank observed value (Z) as well as the significance level (p).

Table 8 illustrates our data analysis scheme for the example of the requirements artifact “Descriptions of as-is situations” from the viewpoint of designers. The columns Sample S1 to Sample S4 summarize the final priority ratings of the different study samples we derived from the data analysis of the individual studies (see Sect. 4.1). The Kruskal–Wallis test applied to the samples S1 to S4 revealed significant differences between the four study groups ($p = 0.006$, $N = 51$). The subsequent pairwise comparison identified significant differences between sample S₃ and sample S₄ ($p = 0.006$, $Z = 3.291$, $N = 40$) with a strong effect size of $r = 0.5$. The fact that this secondary analysis also revealed a significant difference between the FHNW_{A-2013} and other samples supported our previously stated claim that the difference might be attributed to a slight difference in the design of the questionnaire that we used in FHNW_{A-2013} to elicit the importance of requirements analysis. Hence, to mitigate this possible threat, we excluded sample S₃ from the consolidation and the calculated descriptive statistics (see Column ConData_{S1S2S4}) and applied the one-sample Wilcoxon signed-rank test to this consolidated data set. The latter revealed a significant difference between the observed median $Mdn = 2$

and the hypothetical median value $H-Mdn = 3$ ($p < 0.001$, $N = 38$, $Z = -4.833$). This shows a clear opinion regarding the high relevance of descriptions of as-is situations in the consolidated data set. We also observed minor variances in the consolidated data set, as indicated in the histogram in Fig. 4.

Table 8 Secondary data analysis of “as-is situations” from the designer’s viewpoint



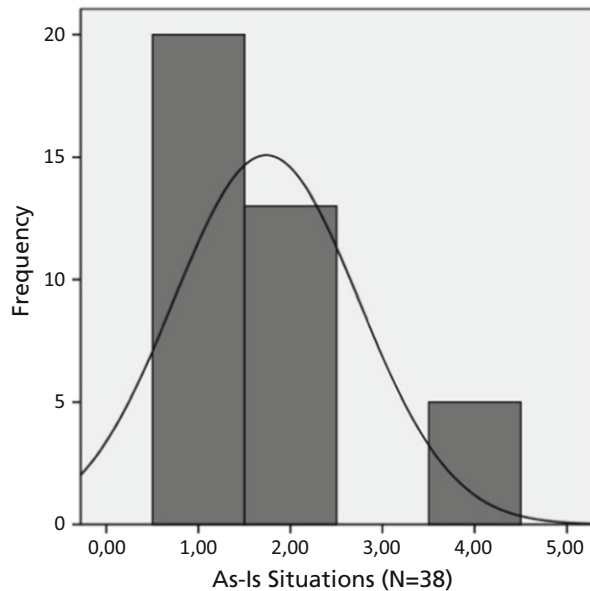
Importance of descriptions of as-is situations^a

Sample S ₁	Sample S ₂	Sample S ₃	Sample S ₄	ConData _{S1S2S4} (N = 38)
Eye-tracking (N = 2)	MUC ₂₀₁₁ (N = 9)	FHNW _{A-2013} (N = 13)	FHNW _{ALL} (N = 27)	
Mdn (M) Min–Max	Mdn (M) Min–Max	Mdn (M) Min–Max	Mdn (M) Min–Max	Mdn ^b (M) Min–Max
3 (2–4)	2 (1.78) 1–2	4 (3.08) 1–4	1 (1.63) 1–4	1*** (1.74) 1–4

^aResponse rating scale: 1 = The RA is very important for IxD/UI design, 2 = The RA is rather important for IxD/UI design, 3 = The RA is rather unimportant for IxD/UI design, 4 = The RA is very unimportant for IxD/UI design

^bOne-sample Wilcoxon signed-rank test $H_0: H-Mdn(x) = 3$; * $p < 0.05$; ** $p < 0.01$; *** $p < 0.001$

Fig. 4 Histogram visualizing frequencies of importance ratings



4.3.2 Data Interpretation

In order to draw general conclusions with regard to the *overall importance level* of a certain requirements artifact, we first interpreted the mean values (M) resulting from the analysis of the descriptive statistics on the consolidated data. To do so, we applied the same rules as during the data analysis and interpretation of the individual studies introduced in Sect. 4.2.2.

As highlighted in our conclusions in Sect. 4.2.2, we aimed to introduce a more fine-grained differentiation of the classification of those requirements artifacts that were identified as “rather important.” The fine-grained conclusions link our empirical data and knowledge to the *role-specific views* utilized in our envisioned technical solution (see Fig. 3).

As introduced in Sect. 3, we envisioned a role-specific view containing high-level or detailed descriptions of requirements artifacts that are of high priority for a certain role (TOP Artifacts). Besides that, we envisioned a second view (IMPORTANT Artifacts) containing high-level or detailed descriptions of requirements artifacts that are important but less critical than the artifacts contained in the TOP Artifacts view. Requirements artifacts and related descriptions considered as rather unimportant or even very unimportant will not be included in the role-specific views. Based on these concepts, we defined the following interpretation scheme for drawing fine-grained conclusions toward the role-specific views:

- IF ($1.0 \leq M(x) \leq 1.5$), THEN assign detailed descriptions of the requirements artifact to the view TOP Artifacts for the corresponding role.
- IF ($1.5 < M(x) \leq 2.0$), THEN assign detailed descriptions of the requirements artifact to the view IMPORTANT Artifacts for the corresponding role.
- IF ($2.0 < M(x) < 2.5$), THEN assign only high-level descriptions of the requirements artifact to the view IMPORTANT Artifacts for the corresponding role and hide further details in the role-specific views.
- IF ($2.5 \leq M(x) \leq 4.0$), THEN hide descriptions of the requirements artifact in the role-specific views.

The concrete shape and representation of “detailed descriptions” or “high-level descriptions” for each of the requirements artifacts were defined based on the results regarding the suitable level of detail (RQ-2_{EmpStudies}) and the preferred notations (RQ-3_{EmpStudies}) for relevant requirements artifacts that we investigated in Study #1 (eye-tracking study) and Study #3 (MUC) (see Sect. 4.1).

Referring to our example illustrated in Table 8, the application of our data interpretation scheme led to the following conclusions for descriptions of as-is situations from the viewpoint of designers:

In general, descriptions of as-is situations are rather important for designers. We assign *detailed descriptions* (in particular process models (e.g., in the form of UML Activity Diagrams) and detailed activity descriptions) to the view *IMPORTANT Artifacts* for designers.

Again, we applied the data analysis and interpretation scheme to all requirements artifacts from all three viewpoints (software architects, usability experts in the role of designers, and software testers).

4.3.3 Data Utilization

Table 9 provides an overview of the consolidated results of the secondary data analysis. For each of the three roles (i.e., software architects, usability expert and tester), the table reflects the priority as well as the preferred level of detail of the investigated artifacts based on the interpretation scheme introduced in Sect. 4.3.2 respectively the data utilization introduced in Sect. 4.3.3.

As introduced previously in Sect. 3, our technical solution utilizes our empirical knowledge about role-specific information needs by applying filter rules on a given SRS structure and requirements artifact templates that are defined in Microsoft Excel[®]. These filter rules are based on priority tags that are assigned to each description item of the artifact templates (see Fig. 3). Thereby, each priority tag corresponds to the relevance of the investigated requirements artifacts from the viewpoint of software architects (A), usability experts/designers (U), and software testers (T). In particular, depending on the preferred level of detail, selected description items of artifacts that were assigned to the view “TOP Artifacts” were tagged with priority A1 (respectively U1 and T1), whereas selected description items of artifacts that were assigned to the view “Important Artifacts” were tagged with priority A2 (respectively U2 and T2). That is, executing the command “Top Artifacts” for software architects in the menu bar (see Fig. 3), for instance, applies a filter rule that displays only requirements artifacts and description items tagged with priority tag A1.

5 Limitations and Future Work

The empirical work of the thesis is rather complex, and besides interesting observations and lessons learned, we also identified and discussed several threats to validity that have to be considered when interpreting the results and that motivated future work. For instance, the majority of subjects were students rather than practitioners with a different level of expertise. Even though we mitigated this threat by cross-

Table 9 Role-specific views on requirements artifacts

Requirements artifacts	Software architect		Usability expert		Software tester	
	TOP	IMPORTANT	TOP	IMPORTANT	TOP	IMPORTANT
Descriptions of stakeholders	<i>Hidden</i>			Detailed	<i>Hidden</i>	
Descriptions of goals		High-level		Detailed	<i>Hidden</i>	
Descriptions of as-is situations		High-level		Detailed	<i>Hidden</i>	
Descriptions of to-be situations		High-level		Detailed	<i>Hidden</i>	
Descriptions of system context		High-level		High-level	<i>Hidden</i>	
Descriptions of interactions		Detailed		Detailed		Detailed
Descriptions of system functions	Detailed		Detailed		Detailed	
Descriptions of quality requirements	Detailed		Detailed		Detailed	
Descriptions of technical constraints	Detailed		Detailed		Detailed	

checking our results with role-specific information needs reported in the literature, further studies with practitioners are needed to increase the external validity of our empirical results. Moreover, in all of our studies, we observed differences in the ratings of requirements artifacts, both between different studies and even within one sample. Even though these differences were (mostly) not significant, we claim that there exist factors (such as working experience, project scope, or personality) that influence the relevance of requirements artifacts. In the future, we aim to investigate such influencing factors in more detail.

In the thesis, we focused our investigations on requirements artifacts that are typically created when RE is performed within the information systems domain, particularly when following the TORE framework. Due to the widespread use of agile methodologies in industry, we aim to transfer our results to the context of agile project settings, which are characterized by extensive collaboration, e.g., face-to-face communication rather than in-depth documentation. In [40], we presented a research agenda and early results that allowed us to better understand RE-related challenges and their implications as well as the relevance of RE-related agile practices from the viewpoint of different agile team members. We also elaborated first guidelines on the example of user stories that incorporate our findings regarding role-specific information needs.

In fact, improving requirements communication in interdisciplinary teams is a highly interesting and relevant topic that motivates our current and future research activities in the context of “learning from non-SE disciplines.” The core objective of this research field is to cross the borders of the SE world and get inspired by best practices from disciplines such as psychology, criminology, film studies, etc. Thereby, we aim to identify synergy potential between these best practices in order to ultimately incorporate and adapt them into new methods.

Currently, we are pursuing the idea of so-called “conspiracy walls,” which are typically used by detectives in criminal investigations to visualize any data, information, assumptions, and potential interrelations related to a particular crime. We envision that a similar visualization applied to requirements-related information could bring several benefits [41]. For instance, joint discussions and analysis of the visualized data could foster communication and information exchange in interdisciplinary teams and hence contribute to a better-shared understanding of both the requirements and the information needs and responsibilities of the various team members.

6 Summary

Delivering high-quality SRS that fit the demands of their consumers is a challenging task for requirements engineers, as different information needs and expectations have to be addressed. These information needs and expectations are strongly dependent on the particular role(s) that the SRS consumers have within a project.

Within the context of the thesis “Role-Specific Views on SRS: An Empirical Approach,” we aimed to address practical problems experienced and observed by us in industry that can be traced to role-specific requirements defects in SRS. In fact, such problems might have a negative influence on the efficient usage of SRS, as it becomes time-consuming and/or cognitively hard for the document stakeholders to resolve and reconcile these problems while working with the SRS. This is critical as it might in turn lead to failures, delays, and frustration in subsequent SE activities, and ultimately to costly changes and budget or time overruns.

In order to address and reduce these defects, we aimed to empirically investigate role-specific information needs regarding SRS from the viewpoint of different SRS consumers involved in downstream tasks such as architecture design, interaction/UI design, and testing. Following a design science research approach, we achieved various contributions, which are summarized in this chapter. Besides a *taxonomy of role-specific defects* as well as an *overview of existing quality assurance and view-based approaches* in the context of requirements documentation activities, we derived *role-specific views* from a series of empirical studies. These views ultimately served as a baseline for a *proof-of-concept implementation* that is capable of automatically generating role-specific views on SRS by filtering the information in accordance with the empirical knowledge about role-specific information needs. The main purpose of this implementation was to demonstrate our solution idea of generating role-specific views on SRS. Moreover, it was subjected to a *case study* that we conducted to investigate hypotheses with regard to the improvement goals of the thesis. We found that role-specific views have the potential to efficiently support SRS consumers during the analysis of SRS and that they are helpful for supporting the communication of requirements in software-developing teams. However, their usefulness depends on the project setting. That is, they are possibly most beneficial in the context of the development of large systems resulting in complex SRS.

In future work, we aim to conduct further empirical studies in industry contexts in order to increase the external validity of our results. Moreover, we will continue our follow-up activities on improving requirements communication in agile project contexts, thereby exploring synergy potential with best practices from non-SE disciplines, such as psychology, criminology, or film studies.

Acknowledgments I sincerely thank my supervisors Prof. Dieter Rombach, Prof. Barbara Paech, and Prof. Schneider as well as Dr. Jörg Dörr, Dr. Marcus Trapp, and Prof. Norbert Seyff for their continuous support and valuable advice, which helped me a lot to shape the thesis and to align and focus my research activities.

References

1. Zhang, W., Pastel, R.: Interdisciplinary team collaboration between software engineers and technical communicators. Proc. Hum. Factor. Ergon. Soc. Annu. Meet. **59**(1), 1137–1141 (2016)
2. Whitehead, J.: Collaboration in software engineering: a roadmap. In: Future of Software Engineering (FOSE '07), pp. 214–225. IEEE, Piscataway, NJ (2007)

3. Glinz, M., Fricker, S.A.: On shared understanding in software engineering: an essay. *Comput. Sci. Res. Dev.* **30**(3–4), 363–376 (2015)
4. Bjarnason, E., Sharp, H.: The role of distances in requirements communication: a case study. *Requir. Eng.* **22**(1), 1–26 (2017)
5. Pohl, K., Rupp, C.: Requirements engineering fundamentals. In: *A Study Guide for the Certified Professional for Requirements Engineering Exam, Foundation Level, IREB Compliant*. Rocky Nook, Santa Barbara, CA (2015)
6. Firesmith, D.: Common requirements problems, their negative consequences, and the industry best practices to help solve them. *J. Obj. Technol.* **6**(1), 17–33 (2007)
7. Alshazly, A., Elfatary, A.M., Abougabal, M.S.: Detecting defects in software requirements specification. *Alex. Eng. J.* **53**(3), 513–527 (2014)
8. Simon, T., Streit, J., Pizka, M.: Practically relevant quality criteria for requirements documents. In: *2008 International Conference on Software Engineering Research & Practice (SERP 2008)*, pp. 115–121. CSREA Press, Las Vegas (2008)
9. Lopes Margarido, I., Faria, J.P., Vidal, R.M., Vieira, M.: Classification of defect types in requirements specifications: literature review, proposal and assessment. In: *6th Iberian Conference on Information Systems and Technologies*, pp. 1–6. IEEE, Piscataway, NJ (2011)
10. Macaulay, L.: Requirements for requirements engineering techniques. In: *Proceedings of 2nd International Conference on Requirements Engineering (ICRE'96)*, pp. 157–164. IEEE, Piscataway, NJ (1996)
11. Hess, A.: Role-specific views on software requirements specifications – an empirical approach. In: *PhD theses in experimental software engineering 69*. Fraunhofer Verlag, Stuttgart (2020)
12. Adam, S., Riegel, N., Gross, A., Uenalan, O., Darting, S.: A conceptual foundation of requirements engineering for business information systems. In: *Enterprise, Business-Process and Information Systems Modeling*, pp. 91–106. Springer, Berlin (2012)
13. Mich, L., Franch, M., Novi Inverardi, P.: Market research for requirements analysis using linguistic tools. *Requir. Eng.* **9**(1), 40–56 (2004)
14. Femmer, H., Unterkalmsteiner, M., Gorschek, T.: Which requirements artifact quality defects are automatically detectable? A case study. In: *2017 IEEE 25th International Requirements Engineering Workshops (REW 2017)*, pp. 400–406. IEEE, Piscataway, NJ (2017)
15. Sommerville, I.: *Software Engineering*. Pearson, Boston (2011)
16. Femmer, H., Mund, J., Méndez Fernández, D.: It's the activities, stupid! A new perspective on RE quality. In: *2nd International Workshop on Requirements Engineering and Testing (RET 2015)*, pp. 13–19. IEEE, Piscataway, NJ (2015)
17. Femmer, H., Vogelsang, A.: Requirements quality is quality in use. *IEEE Software.* **36**(3), 83–91 (2018)
18. Firesmith, D.: Specifying good requirements. *J. Obj. Technol.* **2**(4), 77–87 (2003)
19. Gross, A., Doerr, J.: What you need is what you get! The vision of view-based requirements specifications. In: *2012 IEEE 20th International Requirements Engineering Conference (RE'12)*, pp. 171–180. IEEE, Piscataway, NJ (2012)
20. IEEE 830-1998.: *Recommended Practice for Software Requirements Specifications*
21. Sommerville, I., Sawyer, P.: *Requirements Engineering. A Good Practice Guide*. Wiley, Chichester (1997)
22. Robertson, J., Robertson, S.: Volere requirements specification template. <https://www.reqview.com/blog/2019-02-27-news-volere-requirements-specification-template.html>. (2019). Accessed 30 Apr 2021
23. Wiegers, K.E.: *Peer Reviews in Software. A Practical Guide*. Addison-Wesley, Boston (2002)
24. Gilb, T., Graham, D., Finzi, S.: *Software Inspection*. Addison-Wesley, Boston (1993)
25. Shull, F., Rus, I., Basili, V.: How perspective-based reading can improve requirements inspections. *Computer.* **33**(7), 73–79 (2000)
26. Felderer, M., Beer, A.: Using defect taxonomies for requirements validation in industrial projects. In: *2013 IEEE 21st International Requirements Engineering Conference (RE'13)*, pp. 296–301. IEEE, Piscataway, NJ (2013)

27. Shah, U.S., Jinwala, D.C.: Resolving ambiguities in natural language software requirements: a comprehensive survey. *Software Eng. Notes*. **40**(5), 1–7 (2015)
28. Hayes, J.H., Dekhtyar, A., Sundaram, S.K., Holbrook, E.A., Vadlamudi, S., April, A.: REquirements TRacing On target (RETRO): improving software maintenance through traceability recovery. *Innov. Syst. Software Eng.* **3**(3), 193–202 (2007)
29. Nair, S., de La Vara, J.L., Sen, S.: A review of traceability research at the requirements engineering conference@21. In: 2013 21st IEEE International Requirements Engineering Conference (RE'13), pp. 222–229. IEEE, Piscataway, NJ (2013)
30. Kotonya, G.: Practical experience with viewpoint-oriented requirements specification. *Requir. Eng.* **4**(3), 115–133 (1999)
31. Sommerville, I., Sawyer, P.: Viewpoints: principles, problems and a practical approach to requirements engineering. *Ann. Software Eng.* **3**, 101–130 (1997)
32. Pohl, K.: The three dimensions of requirements engineering. In: *Seminal Contributions to Information Systems Engineering*, pp. 63–80. Springer, Berlin (2013)
33. Kruchten, P.: What do software architects really do? *J. Syst. Software.* **81**(12), 2413–2416 (2008)
34. Fischer, G.: User modeling in human-computer interaction. *User Model User Adapt. Interact.* **11**(1/2), 65–86 (2001)
35. Meyer, B.: Seven principles of software testing. *Computer.* **41**(8), 99–101 (2008)
36. Peffers, K., Tuunanen, T., Rothenberger, M.A., Chatterjee, S.: A design science research methodology for information systems research. *J. Manag. Inf. Syst.* **24**(3), 45–77 (2007)
37. Hess, A., Doerr, J., Seyff, N.: How to make use of empirical knowledge about testers' information needs. In: 2017 IEEE 25th International Requirements Engineering Conference Workshops (REW 2017), pp. 327–330. IEEE, Piscataway, NJ (2017)
38. Adam, S., Riegel, N., Doerr, J.: TORE. A Framework for Systematic Requirements Development in Information Systems. <https://re-magazine.ireb.org/articles/tore> (2014). Accessed 30 April 2021.
39. Hess, A.: Empirical Baseline for the Investigation of Role-Specific Information Needs. (2021). <https://fordatis.fraunhofer.de/handle/fordatis/202>
40. Hess, A., Diebold, P., Seyff, N.: Understanding information needs of agile teams to improve requirements communication. *J. Ind. Inf. Integr.* **14**, 3–15 (2018)
41. Hess, A., Mennig, P., Bartels, N.: Conspiracy walls in requirements engineering - analyzing requirements like a detective. In *CEUR Workshop Proceedings Volume 2584*. <http://ceur-ws.org/Vol-2584/Creare-paper1.pdf> (2020). Accessed 30 Apr 2021

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

