An Evaluation Method for Requirements Engineering Approaches in Distributed Software Development Projects

Michael Geisser and Tobias Hildenbrand and Franz Rothlauf
Department of Information Systems I
University of Mannheim, 68131 Mannheim, Germany
{geisser, hildenbrand, rothlauf}@wi.uni-mannheim.de

Colin Atkinson
Department of Software Engineering
University of Mannheim, 68131 Mannheim, Germany
atkinson@informatik.uni-mannheim.de

Abstract

The distribution of software engineering tasks is becoming ever more common. Requirements engineering, as the most critical phase, therefore requires methods and tools to support distributed teams. However, nearly all requirements engineering methods have originally been designed for collocated scenarios. Before applying these methods in distributed settings in practice and for scientific rigor, proper evaluation has to be conducted. Hence, we developed a methodologically sound and cost-effective evaluation method for distributed requirements engineering methods as well as the corresponding tool infrastructure for conducting evaluation projects.

1 Introduction

Even though common in other fields of research, methodologically sound empirical evaluation is rather scarce and often incomplete both in the fields of software engineering (SE) and information systems (IS) research [21, 12]. Furthermore, many publications on new SE approaches fail to clearly state their goals and provide no research transparency or inadequate evaluation of the proposed concepts [24]. Often, the standard of empirical evaluation methods in SE and IS research is insufficient and neglected in comparison to other research fields [14].

This is especially true for the recent trend towards distributed software development (DSD) projects. Despite the fact that more and more SE projects are geographically distributed, nearly all requirements engineering (RE) methods have been primarily developed for traditional, collocated scenarios. Therefore, a systematic empirical evaluation of RE methods in distributed scenarios is highly important to identify which methods are most useful in DSD contexts. Collaborative software development platforms (CSDP) [18] enable better requirements traceability and thus new and more precise methods of evaluating RE methods, e.g. by capturing and utilizing information on the relationships among different artifacts, starting with requirements in the pre-specification phase [9, 17].

This paper describes a concept for an empirical evaluation of RE methods in DSD settings—both in terms of RE process efficiency and effectiveness (i.e. quality of specification). Moreover, we develop measures and present a tool to evaluate the quality of the overall traceability network originating from requirements.

In section 2, our evaluation context, namely DSD utilizing a CSDP and particularly in RE, and our target hypotheses are introduced. Section 3 presents a brief overview of current practices and quality issues in evaluation research as well as our method of choice. Section 4 introduces the actual design (setting and process) of our evaluation approach and section 5 presents our measures and tool infrastructure for data collection. We conclude with a summary, current limitations, and an outlook on future work.

2 Evaluation Context

This section outlines the characteristics of distributed SE and in particular of distributed RE. In addition, it discusses the problem that these methods have never been empirically and systematically evaluated. Furthermore, hypotheses to verify the usefulness of the respective RE method in DSD settings are defined.
Distributed Software Development  SE generally demands a high degree of collaboration among various stakeholders. Therefore, software developers spend large parts of their time working with others [5]. Reasons for spatial distribution of SE can be manifold. Intra-organizational distribution can be due to historical reasons, whereas inter-organizational distribution may have its roots in outsourcing activities, cost reduction strategies (such as offshoring development activities to low-wage countries, cf. [6]), cross-organizational cooperations, or the need for close customer collaboration. DSD is always afflicted with additional communication issues, exacerbating the efficient coordination of tasks and knowledge management [11]. The use of collaboration software can improve communication and coordination in DSD projects. Groupware, for instance, can help to bridge distances by providing appropriate means of communication, whereas code repositories, forums, or wikis can provide a common space for consistent management of knowledge (e.g., developed artifacts). CSDPs incorporate a wide range of different collaboration functions (including those mentioned before) in one platform [18]. They thus provide synchronous and asynchronous means of communication to negotiate, brainstorm, discuss, share knowledge, and collaborate. CSDPs also integrate different repositories for the consistent storage and concurrent editing of knowledge and artifacts. Hence, CSDPs are particularly important for distributed large-scale SE projects (e.g., in offshore software development, cf. [6]).

Requirements Engineering  During the RE phase, the desired outcome of a SE project is specified. The RE process includes the sub-processes of requirements elicitation and analysis, requirements specification and validation, and requirements management [19]. Requirements elicitation and analysis encompasses the project stakeholders’ work on a complete set of functional and non-functional characteristics which should be incorporated in the final software product. This set should be agreed upon by all stakeholders and more importantly the stakeholders’ understanding of their content must be unambiguous. During the requirements specification phase, this collection of characteristics is transformed to a complete and consistent set of technically specified and distinct requirements, the software requirements specification (SRS). In the requirements validation phase, all specified requirements are checked by the customers in order to detect wrong specifications which could have been caused by misconceptions. The process of requirements management is mainly concerned with the systematic capturing and tracing of requirements changes. Traceability describes the degree to which relationships among different artifacts or between artifacts and actors are consistently documented and reproducible within a SE project (cf. section 5.2). Artifacts that can be traced from or to requirements are, for instance, design documents and rationale, code fragments, as well as test cases. Furthermore, vertical and horizontal traceability can be distinguished [15], where vertical describes the relationships between artifacts within one phase of the SE life cycle, e.g., RE, and horizontal covers relationships of artifacts across different phases. The documentation of both vertical and horizontal relationships leads to a structured traceability network identifying all relationships among artifacts and actors [15]. Traceability has also been identified as a major quality factor in facilitating communication between distributed stakeholders [17].

Hence, RE is often considered as the most important phase during a SE project, especially in DSD [16]. Mistakes made during the RE phase impact the entire SE life cycle and thus traceability. Imprecise, wrong, or misunderstood SRS lead to faulty schedules and cost estimations, additional expenses for rework efforts as well as stakeholder frustration. High-quality SRS are therefore essential for successful SE projects. There are a variety of methods and techniques for RE (an overview of techniques and methods for RE can be found in [23]). However, nearly all RE methods were not originally conceived for distributed environments and have never been evaluated under rigorous conditions. In the wake of the strong trend towards DSD, the research question arises as to how well a particular RE method is suited for such a distributed environment. Also, when designing new RE methods and tools especially for distributed environments, it is important to study whether the novel approaches have positive effects on the RE process.

Hypotheses  In order to prove the usefulness of existing or new RE methods, our evaluation concept is designed to test the following three classes of hypotheses:

- The use of a specific RE method with its underlying tools improves the effectiveness of the distributed RE process in terms of achieved quality of the SRS.
- The use of a specific RE method with its underlying tools improves the effectiveness of the distributed RE process in terms of the quality of the traceability network.
- The use of a specific RE method with its underlying tools improves the efficiency of the distributed RE process in terms of the output/effort ratio.

Before introducing our approach in sections 4 and 5, we first analyze current evaluation practices and guidelines together with different candidate evaluation methods.
3 Evaluation in Software Engineering

This section starts with a discussion of the usefulness of and current state-of-the-art in evaluating SE artifacts in both genuine SE and IS research. Moreover, quality criteria and guidelines for evaluating SE artifacts as well as relevant methods are introduced. Afterwards, our choice of an appropriate evaluation method is explained and known issues as well as related studies are discussed.

Current Practice  The purpose of evaluation is to determine the usefulness of new technologies prior to their adoption and usage. However, in reality practitioners usually adopt new technologies without any evidence for their effectiveness and usefulness [25]. In SE research, there are deficiencies regarding the completeness and quality of evaluations of artifacts and theories [7, 14, 22, 24]. Tichy et al., for instance, examined publications in SE journals and compared them to journal publications from the fields of neural computation and optical engineering. They revealed that articles in SE journals show a significantly lower coverage of experimental validation than articles in other fields of research [22].

Quality Criteria and Guidelines  Different criteria exist for assessing the quality of an evaluation. In general, evaluations have to be objective and free of intentional and unintentional manipulations by the researchers. Reliability and validity are two other general quality criteria for quantitative empirical research. If the used quality criteria are reliable (i.e. the stability of the resulting model is high), they may still not measure the effect they are intended to measure. Hence, validity determines whether an evaluation actually measures the intended effects and how accurate the results are. Even though reliability and validity are both rooted in quantitative research, both also apply to qualitative research [8, 20]. Evaluation practices can lack sufficient transparency. Complex and transparent experiments do not allow for sound findings. High transparency is necessary for replicability, which increases the significance of research results by enabling other researchers to conduct similar, comparative studies [2]. Considering these basic principles, Kitchenham et al. have developed guidelines for empirical SE research [14] in the form of a collection of research instructions based on knowledge of empirical statistics and mistakes observed in previous SE research. These guidelines serve as a framework for developing our evaluation concept. The guidelines are structured according to the following research process: setting the experimental context and design, data collection and analysis, and finally presentation and interpretation of results [14].

Evaluation Methods  Based on the well-established evaluation method taxonomies developed by [1, 12, 24] two basic categories of evaluation methods can be identified that are relevant for both SE and IS research: observational and controlled methods. Following the work of Zelkowitz et al. [24, 25], we studied the different methods within these two categories.

Observational methods  collect data by monitoring the subjects and factors of interest. Observational methods include project monitoring, case studies, and field studies. They are usually conducted “in vivo”, i.e. in productive environments. These methods can be applied to complex scenarios, but are unable to deliver scientific proof for phenomena [24, 25]. They can only provide insights in complex processes and evidence for theoretical models.

Controlled methods  are classically referred to as “experiments”. They are usually conducted “in vitro” (i.e. artificially), since the variables are usually controlled. The evaluation quality aspects discussed above are especially important for controlled experiments, since their objective is scientific proof of theories.

Choice of Evaluation Method  Due to the types of hypotheses to be investigated (cf. section 2) and the resources that are usually available at reasonable costs, we propose a student-based replicated experiment with some additional traits of observational evaluation. Instead of expensive and hard-to-get SE professionals our approach can also be applied to a population of undergraduate students with extensive SE experiences. Since total control of multiple projects is hard to accomplish, we propose the use of complementary observational methods (e.g. questionnaires on subjective experiences) to gain additional insights.

Some qualitative aspects concerning the degree of external validity should be considered when students substitute the actual target population. The researchers’ goal of achieving high external validity can be contradictory to the motivations of students [4]. The students’ motivation might be the achievement of good scores, the acquisition of knowledge, and probably the minimization of time or effort required to achieve the desired academic degree. Depending on the research objectives and evaluation method, motivational aspects thus have to be considered in design, setup, analysis and interpretation of the evaluation.

Berander found commitment to be an important aspect in his evaluation of requirement prioritization behavior. For the evaluation purposes, the commitment shown by students makes them a good substitute for SE professionals [3]. As an exercise in release planning, students attending class were grouped into teams and asked to prioritize a set of requirements for a specific software. The results achieved with classroom students were compared to the results achieved in a similar study with students attending a SE term project
as well as to a case study conducted with professional software engineers. Berander concludes that for requirements prioritization the student projects achieved very similar results to the professional SE teams, while pure classroom students achieved different results. Hence, practical student projects are more appropriate for SE experimentation due to the higher commitment to the given tasks [3].

Therefore, our choice of evaluation method seems to be appropriate for our research objectives. Even though our evaluation concept aims at practical student projects, motivational issues still remain. Thus, in addition to a certain amount of credits, successful students will be awarded with a secondary incentive, i.e. a certificate for DSD experiences using a CSDP which will be issued by the particular CSDP vendor. However, the experience, knowledge, and skills of students can significantly differ from those of the actual SE professionals. These differences must be considered when generalizing research results. Based on the guidelines and quality criteria as well as the above insights from related studies, the following section will present our concrete approach to evaluating RE methods in DSD projects using students as subjects.

4 Experimental Design

This section describes the experimental design consisting of the experimental setting and the treatment process.

4.1 Setting

Our evaluation concept is designed for large-scale university courses (50 and more subjects) in SE with the main objective to develop software in small teams of students with about five participants during the course of one term (i.e. in about 16 weeks). All student groups should use a CSDP with comprehensive state-of-the-art tool support for DSD (cf. section 2). For the comparability among the groups it is highly important to achieve a homogeneous distribution of skills on the basis of a SE-related aptitude test and self-assessment. Moreover, a predefined high quality sample SRS is necessary which will serve as a benchmark for the quality assessment of the SRS developed by the students. The sample SRS must encompass all mandatory and optional requirements possible in the project. A delimitation of the development scope must be defined by an extensive documentation of unnecessary requirements. The development of the sample SRS should in particular focus on atomic requirements in order to ensure a high degree of comparability.

One half of the teams will use the RE method under evaluation and corresponding tools while the other half will employ a traditional RE method as a control group. As a rather traditional approach, a waterfall model with document-based RE can be applied. Each group will have a student research assistant (SRA) serving as project manager and guide for the team. In addition, he motivates the students and ensures an equal workload distribution within the group. The researchers represent the customers to the SE teams.

4.2 Treatment Process

Together with the registration for the course, a subject classification questionnaire has to be filled out by the students. This facilitates a clear description of the evaluation subjects and thus a better understanding of the obtained results. Moreover, the students’ self evaluation serves as the basis for composing the teams, i.e. to achieve homogeneous groups. The questionnaire includes questions regarding demographic data, academic data, SE experience, and SE projects participated in.

In the first half of the term, student subjects attend parallel SE classes to learn method and tool usage. The course also requires introductory meetings where students are briefed on the development task and assigned to their development team. In order to enhance students’ RE skills in particular the researchers have to offer a special RE lecture. This will leverage the professionalism of students in their general approach to RE. During the whole SE course all students have to hand in weekly questionnaires (the specific measures are presented in section 5). Furthermore, the SRAs will assess the students working habits, skills and motivation on a weekly basis. The results of these weekly assessments will give further insights into actual distribution of personal attributes among and within the teams.

From the evaluation perspective, the SE process will be divided into a pre-SRS-phase and a post-SRS-phase (cf. [9]). During the pre-SRS phase, which includes all development activities from requirements elicitation until handing in the SRS document, the student groups work on the creation of the SRS. The customer representatives are consulted for the requirements elicitation and analysis activities. After the SRS documents have been completed and handed in, the SRAs and the customer representatives assess the SRS quality levels in a first review (assessment criteria are explicated in section 5). During the post-SRS phase, the development teams design, code, test, and debug the software. Customer representatives will stay available for the teams’ consultation. After the SE process is completed, there will be a second review in which SRAs and customer representatives assess the final product, i.e. code and documentation.

The researchers do not intend to manipulate the course of the projects in any way. On the contrary, they take the responsibility for conscientiously taking care of an unbiased treatment and record any influences which could corrupt
the measured results (cp. [14]). This documentation is extremely important for a truthful interpretation and the identification of possible improvement for similar evaluations in the future. Since all student groups use one common CSDP, this enables the researchers to build up an evaluation knowledge base serving as a reference for evaluation decisions in order to provide consistency and high reliability. Moreover, it constitutes an additional data source for complementary qualitative analyses.

Our evaluation method is thus similar to that of a replicated experiment. Many instances of the same project are conducted simultaneously, while two different treatments (the RE method under evaluation and the traditional RE approach) are applied to the subjects. This type of study can be identified as in vitro due to a non-industrial environment. Yet, it is not as artificial as a laboratory experiment, since the subjects work on actual development task in a complex setting. The involvement of the researchers as customer representatives and additional qualitative data are characteristic of a case or field study.

5 Data Collection Approach

In the following, we define the measures we propose to investigate the hypotheses from section 2. Measures for SRS quality and traceability network quality are necessary to examine the first two hypotheses related to the effectiveness of RE approaches. The third hypothesis requires the measurement of the output/effort ratio. The output of the RE process is the validated SRS and, in addition, we need to measure the effort put in the RE phase in particular.

5.1 Quality of Specification

A good SRS shall be correct, unambiguous, complete, consistent, prioritized and stability ranked, verifiable, modifiable, and traceable (for definitions of these criteria see [13]). In the following, we point out how these criteria can be measured in DSD settings. We do not consider these characteristics as binary variables and therefore assess the degrees of the different quality criteria achieved.

Correctness \(S_{cor}\) is determined through a comparison between the sample SRS and the SRS handed in by the teams. Requirements included in the SRS which are explicitly or implicitly excluded compared to the sample SRS represent incorrect requirements. The degree of correctness specifies the percentage of correct requirements in the teams’ SRS.

Unambiguity \(S_{una}\) is assessed through a formal inspection of the SRS. To be able to do so, the terms which require a definition and the requirements which might contain ambiguities have to be identified. Requirements containing undefined terms automatically get the attribute ambiguous. In a dialog between the researchers the remaining requirements are checked for their common understanding. With the customer view provided by the researchers and the team view provided by the team’s SRA, it is decided whether a requirement is to be considered unambiguous. The degree of unambiguity is the percentage of requirements found to contain no ambiguity.

Completeness \(S_{com}\) represents the average of (a) the percentage of requirements covered, (b) the percentage of specified responses to possible inputs for the system, and (c) the percentage of correctly labeled and referenced requirements (according to the IEEE definition of a complete requirement [13]).

Consistency \(S_{con}\) describes the percentage of requirements not involved in any conflicts with other requirements. This measure is assessed manually by the researchers according to the IEEE definition [13].

Prioritization and stability ranking \(S_{psr}\) is measured by the percentage of requirements which are attributed with priority indicators or stability indicators. Since it is impossible to objectively measure the adequacy of both rankings, the measures can only capture the raw attribution.

Verifiability \(S_{ver}\) is assessed manually by the researchers. An ambiguous requirement is generally not verifiable [13]. Hence, all unambiguous requirements are checked whether they could be clearly defined as fulfilled or not fulfilled after the software product is finished.

Modifiability \(S_{mod}\) depends on (a) the SRS organization, (b) the absence of redundancies and (c) the atomic expression of requirements. Therefore, each aspect is captured in one of three equally weighted separate sub-measures. The SRS organization depends on the table of contents and explicit cross referencing of dependencies. The degree of redundancy absence describes the percentage of requirements with no redundant contents. The level of atomicity states the percentage of requirements which cannot be divided into two or more separate, but still meaningful, requirements.

Since traceability constitutes one of the most important quality criteria for a SRS and the SE process as a whole (cf. section 2), we measure the complete traceability network starting with requirements over the entire project life cycle (see following section).

5.2 Quality of Traceability Network

Measures concerning the quality of the traceability network can be differentiated into pre-specification traceability (among SRS, other RE documents, and sources, equals source traceability), specification traceability (only within the SRS, equals requirements traceability) as well as post-specification traceability (from SRS to design models, code, etc., i.e. design traceability, cf. [9, 19]). We analyze
traceability in two steps: pre-SRS-traceability, including SRS traceability, is examined during the first design review whereas post-SRS-traceability is examined during the final design review (cf. section 4). Instead of referring the measures to the referenced requirements they refer directly to the traces between artifacts and those between artifacts and actors.

**Specification Traceability** ($S_{tra}$) is calculated by adding a measure for backward traceability ($S_{traB}$) and a measure for forward traceability ($S_{traF}$). $S_{traB}$ measures the percentage of requirements which can be traced back to their origins (project stakeholders and/or corresponding documents). $S_{traF}$ captures the degree of requirements attributed with an identifier, allowing the future referencing of artifacts spawned from a requirement. $S_{traB}$ depends on rigor documentation during SRS development and cannot be reproduced easily after SRS completion. In contrast, $S_{traF}$ can be enhanced easily at the time after SRS completion. Therefore, $S_{traB}$ is weighted with 80 percent and $S_{traF}$ with 20 percent. This measurement of traceability follows the definition of [13].

**Pre-Specification Traceability** i.e. its coverage and relevance are measured by source traceability and requirements traceability within the traceability network (TN): Coverage of source traceability and coverage of requirements traceability are both defined as the percentage of traces documented out of the total number of discoverable correct traces ($TN_{srcC}$: source trace coverage; $TN_{reqC}$: requirements trace coverage). Source and requirements trace integrity describes the percentage of correct traces among the total of documented traces ($TN_{srcI}$: source trace integrity; $TN_{reqI}$: requirement trace integrity).

**Post-Specification Traceability** is measured as design traceability during the final project review—in both forward and backward direction: Trace design coverage describes the percentage of traces documented out of the total of discoverable correct traces ($TN_{fC}$: forward design trace coverage; $TN_{bC}$: backward design trace coverage). Trace design integrity describes the percentage of correct traces among the total of documented traces ($TN_{fI}$: forward design trace integrity; $TN_{bI}$: backward design trace integrity). In general, coverage describes the overall percentage of correct links documented whereas integrity measures the percentage of desirable links among all links documented (cp. recall and precision, [10]). The final decision as to whether a link between artifacts or artifacts and actors is adequate can only be done manually by researchers and SRAs [10].

**Traceability Visualization and Analysis** To support researchers in analyzing traces, we developed a tool for Trace Visualization (TraVis). This tool extracts the traceability network from the CSDP and visualizes the relationships according to different filters and views. Thus, we use TraVis to gain additional insights for our qualitative observations and analyses. In SE practice, TraVis can also be applied to improve traceability management during SE projects, enabling real-time analyses and editing of the traceability network. Figure 1 depicts a network of different requirements and users from one of our evaluation projects (cf. also [6]).

**Figure 1. Visualization with TraVis**

### 5.3 Effort

In addition to measuring specification and traceability network quality, we have to capture the effort in order to be able to examine efficiency questions (cf. section 2).

The RE effort has to be measured in terms of time spent for the requirements elicitation and analysis phase ($T_{re}$) and for the requirements specification and validation phase ($T_{rs}$). Moreover, the time spent for the subsequent post-SRS phases is measured separately for design ($T_{des}$), coding ($T_{cod}$), testing and debugging ($T_{tdb}$), as well as for project management tasks ($T_{mgt}$) to be able to relate traceability network quality to the different phases. This allows for more differentiated efficiency statements. All these variables are measured through weekly reports by all participants. To this end the SRAs encourage the consequent documentation and assist in correctly associating tasks to corresponding RE phases.

### 5 Conclusion

This papers presents an adaptable, cost-effective, and student-based evaluation concept for RE approaches in distributed settings. Our experimental design considers rele-
vant quality criteria and guidelines proposed by both the SE and IS research communities. We use an evaluation method that combines quantitative measures from replicated experiments with some complementary qualitative observational methods. Our measures are supposed to examine research questions pertaining to RE process effectiveness and efficiency. Moreover, the concept is designed to measure the effectiveness of capturing traces from and to requirements. To support this task, we have deliberately developed TraVis, a tool for trace visualization and analysis. Altogether, our systematic approach intends to provide for high-quality evaluation results at reasonable realization costs.

The experimental design described here discusses many facets of data collection. In this step, we will consistently document all relevant occurrences and findings. For instance, a change in team structures due to drop outs, or the substitution of SRAs are occurrences that must be documented in order to sustain scientific correctness. The subsequent data analysis phase will encompass a statistical analysis of the quantitative data measured—complementarily, depending on recurring structures, the questionnaires can also be statistically analyzed. Since the researchers are involved in the development process and the development results are unique and recognizable software products, a blinded analysis is not feasible. A detailed documentation of analysis methods and proceedings should be produced in order to inform the research community and eliminate suspicions about biased analysis and a lag of rigor. Our approach will be further adapted as we conduct more and more evaluation studies as instances.

References