Dedicated Support for Experience Sharing in Distributed Software Projects

Anna Averbakh*, Eric Knauss‡, Stephan Kiesling*, Kurt Schneider*
* Software Engineering Group, Leibniz Universität Hannover, Germany
{anna.averbakh, stephan.kiesling, kurt.schneider}@inf.uni-hannover.de
‡ Department of Computer Science and Engineering, Chalmers | University of Gothenburg, Sweden
eric.knauss@cse.gu.se

Abstract—Systematic management of experience and knowledge in distributed software development promises huge increases in effectiveness and efficiency. Yet, specific problems need to be overcome: Communication between partners is difficult and awareness about the knowledge available at different locations is impaired. Even motivated developers are often reluctant to share experiences, because they do not know where and how to submit them as well as if they are allowed to share sensitive information or intellectual property. If they submit, their experiences are often presented in a way not useful for others and cannot be easily refined into best practices. In this paper we identify barriers of knowledge sharing in a literature review and discuss how dedicated tool support and automated heuristic critiques can mitigate such problems by offering the following features and qualities: guidance of project participants in creating experiences that will more likely be helpful, positive influence of their motivation to share, and easily accessible and integrated into a trustworthy experience engineering processes. Preliminary evaluation with a prototype shows that this concept can increase willingness to submit experience as well as their quality.

Index Terms—experience elicitation; heuristic feedback; global software engineering

I. INTRODUCTION

Around the year 2000, many large companies recognized the importance of knowledge for software development. Several companies (e.g. Siemens [1], Ericsson [2], Daimler [3], etc.) invested heavily in knowledge management for software engineering problems. Those approaches were extended towards systematic exploitation of experiences. The intention was to avoid repeating mistakes, and rather learning from a company’s own specific experience. Distributed projects run by more than one company or business unit, however, have added new challenges to these goals [4], [5], [6]. Due to geographical, temporal and cultural differences, distributed projects suffer from an impaired communication flow and awareness [4], [7].

The success of an experience and knowledge management (EKM) initiative highly depends on the quality of the generated content. It is important that experiences are engineered into reusable guidelines and not just distributed as raw, subjective experiences [8], [9]. There are three major challenges for getting sufficient input into this engineering process. The most severe impediment for project participants to share information over the years has still remained the effort to benefit factor [10]: If the effort to share information exceeds the reward or benefit, people will not do it [11]. However, they are more willing to share, if they believe that their contributions will be valuable to others [12]. There has been evidence that “quality of the reported experience highly depends on the individual communication skills of the contributor, e.g. the ability to structure the content, to formulate the experience with accuracy, and to describe it properly according to the needs of the target audience” [13]. In addition, bearers of experience often do not know how and where to submit their experience [14]. More research is necessary to bridge this gap.

Based on a literature study we derive barriers for submitting experience in distributed software development and propose the use of heuristic critiques (automated computer based feedback on experience writing) to help lowering these barriers. Such critiques automatically analyze textual input against heuristic rules and offer immediate feedback to the author. In a preliminary evaluation, we evaluated the effects of this approach and found that the heuristic critique’s immediate feedback on how to write experiences had a positive influence on (i) the perceived effort to benefit ratio and willingness to contribute experience as well as (ii) the quality of experiences.

The rest of this paper is structured as follows. In the next chapter, we present the challenges of experience management in globally distributed software engineering projects based on related work and present our research questions. Then we discuss our concept and the preliminary evaluation. This paper concludes with a summary and suggestions for future work.

II. BACKGROUND: CHALLENGES TO EXPERIENCE SHARING IN GLOBAL SOFTWARE ENGINEERING

The first step in an experience management initiative based on an Experience Factory [8] is to collect (raw) experiences [9]. One simple way to spontaneously externalize an experience is to write an observation sheet, a (digital or paper) form to capture ad-hoc experiences [15]. An experience is a triple, consisting of 1) an observation, 2) an emotion (about the observed event) and 3) a conclusion or hypothesis [9]. Globally distributed software development adds a new dimension to experience management [6]. Systematic learning during the project becomes even more important, because different organizational and demographic cultures come together. The distributed organization of the project needs to learn how to...
use these characteristics to the advantage of the project. At the same time, new challenges arise that impede successful knowledge sharing.

To uncover these challenges we conducted a literature review. The search keywords were (“knowledge OR “experience”) AND “sharing” AND (“barriers” OR “impediments” OR “issues”) AND “software engineering”. Searched databases were Emerald (55 results), IEEEExplore (0), SpringerLink (102), ScienceDirect (2), Wiley InterScience Journal Finder (2), ACM Digital Library (994), SAGE Journals (28) and Google Scholar (19700). Included in the search were journal articles, workshop and conference papers. We applied our search on the title of the publications written in English language and the keyword “software engineering” on the text. Many publications were also found through an explorative search. For the Google Scholar search, we only used the first twenty pages (200 results).

As inclusion criteria, publications were considered relevant if they report on knowledge or experience barriers. This includes general personal sharing barriers or barriers that concern groups of people (social and cultural issues). In a first review iteration and motivated by experience in a distributed collaborative project [16] it became clear that many barriers are not specific to global software development. For this reason, this review is not restricted to global settings. It poses new challenges to experience and knowledge sharing but other barriers have influence on globally distributed projects as well.

Altogether, 78 relevant publications and 63 barriers were identified. From each publication relevant to this review, factors and barriers were collected into a table. Afterwards similar factors were consolidated and grouped according to their barrier category. From the publication causal relations between the factors were drawn.

Fig. 1 summarizes the review by displaying the higher-level barriers which are most symptomatic to global software development. The number of sub-causes (not discussed in this paper) are given in brackets. In this paper we are focusing on the elicitation of experiences and propose a tool as part of an experience environment to support experience writing and lower the barriers from Fig. 1 in distributed software development. In the scope of this paper we will focus on the gray shaded barriers.

1) No Immediate Benefit: A knowledge and experience management initiative must offer the experience bearer a perceived benefit. Furthermore, the beneficiary must be the experience bearer who has contributed his experience [11]. As most people are no altruists, without personal benefit from the EKM initiative they will unlikely share again [17]. They may interpret the situation as “invest now, and someone else might harvest later” [18]. The benefit must be perceived shortly after contribution [17], [18].

To produce a quick benefit, many researchers focus on reward mechanisms to motivate knowledge bearers to share their experiences. For example, Bartol et al. propose organizational rewards [19]) while Dencheva et al. discuss intrinsic reward mechanisms (levels, ranks and awards) [20].

The open question is if heuristic critiques can raise intrinsic motivation just by making the author feel guided or even proud of contributing a high(er) quality experience. There is evidence that developers are more motivated to share, if they believe that their contributions will be valuable to others [12]. To evaluate this assumption, we pose the following research question:

RQ1: Do heuristic critiques help to write experiences with higher quality?

2) Content Perceived Not Useful: An experience management initiative is not successful, if developers do not perceive its content as useful. The content may be perceived not useful, if it does not help the developer to solve his problem or specific task. Often, this is due to an inappropriate presentation or maturity level of disseminated experiences. It is important that experiences are engineered into reusable guidelines and not just distributed as raw, subjective experiences. Raw experiences are often too specific, unreliable, or contain confidential information. Thus, they have to be engineered to make them applicable to other projects [9].

Other research has tried to improve contributed content. Bettenburg et al. present a prototype that guides users through writing better bug reports [21]. Their study is related to our work when considering bugs as a special type of an experience. In a survey they reveal a mismatch between the information that developers need and the information that users supply. Shugerl et al. evaluate bug report quality with information retrieval techniques [22]. Support for general experience elicitation should provide the possibility to collect feedback not only to a product, but also to the development process. By this, we hope to support an active learning environment (in distributed software development) which is based on the exchange of relevant and well packaged experiences.

Similarly to the goal of heuristic critiques to improve the contributed quality, Wilson et al. present a system that aims to raise requirements quality with the help of natural language processing (NLP). It searches bad phrased requirements for improvement [23]. Bär et al. integrated NLP techniques into a Wiki to support users with the tasks of adding, organizing,

---

1 A complete overview over the identified barriers and sources can be found at http://www.se.uni-hannover.de/misc/aaeberkahl/all-exp-barriers.pdf
and finding content [24]. It remains to be shown that such concepts can improve the quality of experiences:

\textbf{RQ2: Do developers find experiences written with the help of critiques more helpful or richer in content?}

3) \textbf{Lack of Trust:} Lack of trust occurs when developers are not sure if sharing their experience with partner sites will be beneficial or detrimental (e.g. due to NDA policies) to them [29]. This barrier can generally be mitigated by a knowledge management infrastructure like e.g. an Experience Factory [8], which is a fundamental approach to systematically elicit, refine, package, and reuse experience. It is a basis for our concept of experience engineering. In our previous work, we presented an experience engineering process for collaborative projects that considers trust issues [30] and we will not further discuss this barrier in the scope of this paper.

4) \textbf{No Access to System:} No access to the experience base can be an issue when joint ventures contain partners that have a competitive relationship outside the collaborative project. Then, usually the strongest collaboration partner can restrict access to the stored knowledge for the competing partners. The developers of the partner organization(s) do not get enough access to system to benefit from it [16]. Similarly to lack of trust, our infrastructure from previous work [30] approaches this issue presenting a rights management structure and engineering process. It ensures that no sensible experiences can be viewed by competitive partners making access restrictions redundant. We will not further discuss this barrier in the scope of this paper.

5) \textbf{Lack of Awareness:} Lack of awareness is increased by physical distribution, leading to more difficult and less direct communication between project participants. Often, participants do not know where and how to submit experience [14]. In addition, participants often have difficulty to assess the relevance of their knowledge, as they are not aware about others’ problems [20] — a general sharing problem.

A dedicated tool support mitigates the lack of awareness where and how to submit experiences, if designed to be unobtrusive and easy accessible in the workplace for each project participant. Schön’s reflection-in-action approach [25] can raise awareness about one’s knowledge through specific techniques that may tickle tacit knowledge out of the author he would not reflect on otherwise. Mitigating the second awareness problem, Fischer’s domain-oriented design environments (DODEs) encourage practitioners to submit their experience providing an argument component with immediate feedback [26]. This feedback can initiate reflection.

As related research to raise awareness, Schneider [14] and Lübbe et al. [27] propose to collect spontaneous feedback by providing specialized feedback facilities. Our approach is inspired by DODEs. The argument component providing immediate feedback (or critiques) is based on heuristics [28]. We apply his idea to help writing better experiences and to support reflection-in-action leading us to the following research question:

\textbf{RQ3: Does heuristic critique support influence the amount of experiences submitted by developers?}

6) \textbf{Effort > Benefit:} All barriers in Fig. 1 affect the effort to benefit ratio of EKM initiatives and thus ultimately their success: Not trusting other partners means additional effort to decide which information to share. Without access to knowledge, developers will not see any usefulness in the base’s content and thus the whole system. Without enough perceived personal and immediate benefit, they will not make the effort to share [11].

As related approaches mitigating this issue, Schneider [31] and Meyer et al. [16] propose approaches aiming to extract experience and knowledge as by-product during usual work procedures without additional effort for the knowledge bearers.

Yet it remains an open question how heuristic critiques can motivate users to create more valuable experiences and make them feel more assured when writing experiences. At the same time, the concept should keep the experience writing effort as low as possible. It should not impose too many restrictions but, on the other hand, ensure the most important factors for a good experience. The above defined research question RQ3 can give indication if our approach mitigates this barrier.

In order to investigate our research questions, we need tool support that incorporates heuristic critiques. We created a prototype of such a tool as candidate solution based on the Heuristic Requirement Assistant (HeRA) [32]. Originally HeRA employed heuristic critiques to support writing of requirements and use cases. We reused this infrastructure and adapted HeRA’s critique system and graphical interface to write experiences instead of requirements.2

\section*{III. HEURISTIC SUPPORT FOR EXPERIENCE WRITING}

In this section, we present the main concepts of this paper: characteristics of a good experience for an experience engineer and the heuristics to enhance experience quality.

We define completeness, observability, readability or understandability, traceability and verboseness as characteristics of a good experience in the software engineering domain. To avoid misconception, verboseness in this context is meant positively in the sense of being detailed. Our quality aspects are based on established quality characteristics of codified knowledge [33] (accuracy, readability or understandability, accessibility, currency, authority or credibility), requirements [23] (complete, consistent, correct, modifiable, ranked, testable, unambiguous, valid, verifiable) and bug reports [34], [22], [21] (focused, observable, readable, verbose). Requirements and bug reports are similar to an experience in the way that they describe events or make proposals, and are created with the goal to be easy to process in a software development (or maintenance) process. Besides, a bug report can be viewed as a special type of experience or feedback [27]. We did not consider some very bug report-specific artifacts like stack trace, patches, code samples, test cases, etc. [21]. Considering that a goal of our concept is to stay light-weight in terms of effort, we do not want to impose too many restrictions or

\footnote{For more details and a screenshot of our prototype, please refer to http://www.se.uni-hannover.de/pages/en/heur_critiques_exp_elicit.}
demands on the experience bearer but also want to ensure the most important aspects of a good experience. Too many demands may discourage experience bearers fearing too much effort to fulfill them. Thus, we discarded the quality aspects focused, unambiguosness, consistency, and correctness, because experiences are subjective and impressions of the same event can differ depending on the point of view. We also discarded verifiability, validity, ranking, and modifiability, as we consider ranking, rephrasing and comparing experiences with others experiences engineer’s tasks. Also accessibility, currency and credibility can be ensured during experience engineering and maintenance and do not need to be imposed on the experience bearer.

In Table I we list the concrete heuristic critiques for experience writing derived from or implementing the characteristics. For each critique the table displays characteristics it supports or enables as well as a criticality. Error is the most severe criticality, followed by a warning. Hint signifies an optional advice and info a general information. Feedback can either be general or specific to the experience part currently edited. All critiques, except 11, are context sensitive and are visible within a certain experience part. The last critique being a general information on how to write an experience is visible in all fields. Heuristics 1, 2, 3 and 8 are applied on observation, emotion and conclusion separately but are consolidated in this overview. We implemented the heuristic rules for German language, but they are generally applicable to other languages. To determine the threshold for Heuristic 2, we took a value slightly above the average sentence length of 7.08 words in German literary prose [35]. For the readability measure in Heuristic 3 we implemented the Flesch Reading Ease score [36]. We included this algorithm to raise the experience bearer’s awareness about writing style, even though we were expecting low performance because of the brevity of most observations and conclusions. This was successful, as we could observe participants to actually rewrite texts into shorter and less nested sentences during evaluation. For Heuristics 2 and 3 we intentionally did not monitor the emotion field. In our experience, it is not important for an emotion to be very detailed. In Heuristic 4 we use emotion words from the WordNet Affect list. To control for the quality of our critiques, we pose the following research question:

RQ4: Do software engineers find our critiques helpful and justified?

IV. PRELIMINARY EVALUATION

Our research goal is to lower the impact of individual communication skills, improve the effort-benefit ratio, and give experience bearers general support on how to submit their experiences. We evaluated our four research questions.

A. Setting

Ten graduate students with a Bachelor of Science degree in computer science and software development experience participated in our evaluation. We specifically chose participants with recent software engineering (SE) experiences, for example students, who had recently participated in SE activities like writing a specification or developing a mobile application in a team. 30% of our participants work or have worked in industrial software projects beside their study.

B. Methodology

We conducted a cross validation with two groups (G1 and G2) to evaluate the helpfulness and acceptance of our heuristic support. The participants in G1 were asked to write 2-3 SE-related experiences without and then with our critique system. The control group G2 had a reverse order of experience writing: first with and then without critiques. To answer RQ1 we used the words set from http://www.cse.unt.edu/~rada/affectivetext/.

<table>
<thead>
<tr>
<th>ID</th>
<th>Characteristic</th>
<th>Criticality</th>
<th>Feedback</th>
<th>Heuristic rule</th>
<th>Helpfulness</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Complete</td>
<td>Error</td>
<td>The observation / emotion / conclusion does not contain text.</td>
<td>The observation / emotion / conclusion field is empty.</td>
<td>70%</td>
</tr>
<tr>
<td>2</td>
<td>Readable, Verbose</td>
<td>Warning</td>
<td>Your observation/ conclusion is very short.</td>
<td>Observation / conclusion text contains &lt; 8 words.</td>
<td>90%</td>
</tr>
<tr>
<td>3</td>
<td>Readable</td>
<td>Warning</td>
<td>Your observation/ conclusion could be hard to read.</td>
<td>The readability index is &lt; 30.</td>
<td>40%</td>
</tr>
<tr>
<td>4</td>
<td>Observable</td>
<td>Warning</td>
<td>Your emotion might not be clearly stated.</td>
<td>The text in emotion field does not contain an emotion word.</td>
<td>70%</td>
</tr>
<tr>
<td>5</td>
<td>Complete, Observable</td>
<td>Warning</td>
<td>Your conclusion might not convey a recommendation.</td>
<td>The text in conclusion field does not contain modal verbs.</td>
<td>60%</td>
</tr>
<tr>
<td>6</td>
<td>Observable, Traceable</td>
<td>Warning</td>
<td>Your conclusion possibly lacks actors or persons in charge.</td>
<td>The text in conclusion field is written in passive.</td>
<td>50%</td>
</tr>
<tr>
<td>7</td>
<td>Readable</td>
<td>Warning</td>
<td>Your observation / emotion / conclusion contains acronyms.</td>
<td>The text contains at least 2 successive capital letters.</td>
<td>30%</td>
</tr>
<tr>
<td>8</td>
<td>Traceable</td>
<td>Hint</td>
<td>The author is missing.</td>
<td>The author field is empty.</td>
<td>30%</td>
</tr>
<tr>
<td>9</td>
<td>Traceable, Verbose</td>
<td>Hint</td>
<td>Observation is…</td>
<td>-</td>
<td>60%</td>
</tr>
<tr>
<td>10</td>
<td>Traceable, Verbose</td>
<td>Hint</td>
<td>Emotion is…</td>
<td>-</td>
<td>60%</td>
</tr>
<tr>
<td>11</td>
<td>Traceable, Verbose</td>
<td>Info</td>
<td>Experience consists of an observation, emotion and conclusion.</td>
<td>-</td>
<td>70%</td>
</tr>
</tbody>
</table>
and assess the quality of the written experiences, we specified experience quality metrics. These metrics are very similar to the heuristics. Each experience was read by an experience engineer and evaluated if it complied with the metrics. The experience engineer however, did not know which critiques were fulfilled or not for this experience. Afterwards, in order to evaluate RQ2, RQ3 and RQ4, both groups answered a questionnaire\(^4\) about the helpfulness of the critiques in general and for each critique in particular. They also had to rank four experiences according to their perceived quality. Two of the experiences were of poor and average quality according to our metrics. They lacked a conclusion, contained very specific acronyms, or were too short to convey enough information. The other two were the same ones but enhanced with our critique system. We used real experiences from a past distributed SE project.

### C. Results

The boxplot in Fig. 2 shows a visible experience quality improvement for RQ1. The median rates of the fulfilled quality metrics for experiences created without feedback (G1 and G2) are distinctly lower in comparison to those with critiques (G1c and G2c). The boxplot for G2 indicates a slight learning effect compared to G1. The median of G2 is slightly higher than of G1 and the quality improvement between G2 and G2c is smaller than between G1 and G1c.

RQ2 can also be answered positively. In Fig. 3, 80% consider experiences enhanced with the help of critiques more helpful. Experiences enhanced with the help of critiques (E2 and E4) were ranked higher than the original ones (E1 and E3). The poor experience E1 (very short, acronyms, no conclusion) was unanimously ranked as of bad quality.

RQ3 was confirmed. 70% would write more experiences with a tool that has critiques support. For 50% critiques could lower the barrier and motivate to write more experiences. The latter is a rather inconclusive result. It indicates that users should be able to deactivate the heuristic critiques. RQ4 was also confirmed. The statistic for RQ4 shows that 100% of the students found the critiques (rather) helpful and 80% justified.

The helpfulness of each particular heuristic is displayed in Table I. The critiques made all participants feel more certain about the helpfulness of the critiques in general and what to write in an experience. On the other hand, the heuristic feedback was perceived as rather interrupting (70%). The participants often had to stop writing to read the feedback text. For 70% of the test participants, feedback slowed down their writing process. We think however, that this negative effect can vanish with repeated use and growing familiarity with the critiques.

### V. Threats to Validity

For the preliminary evaluation of our concept we prioritized a controlled environment over transferrability of our results in industrial practice. Therefore, we employed a controlled setup with graduate students. Though our results are affirmative, they are not statistically significant (\(p = 0.19\) for G1 and \(p = 0.13\) for G2) and we had a comparably small number of participants (\(n = 10\)). Nonetheless, our results provide a distinct indication.

Concerning the questionnaire, it is always possible that participants might misinterpret questions. We mitigated this risk by testing the questionnaire and by offering assistance during the experiment. There is a risk that students tried to confirm what they believed to be our research goals. We

\(^4\)The questionnaire (in German) can be downloaded from http://www.se.uni-hannover.de/misc/aaverbakh/Questionnaire.pdf

---

**Fig. 2. Experience quality without (G1, G2) and with critiques (G1c, G2c).** Note that G1 started without, while G2 started with critique support.

**Fig. 3. Survey results.**
mitigated this threat by formulating questions alternatively from different perspectives and by formulating our research goals in the training as neutral as possible. Rating the four experiences (fig. 3, RQ2(b)) after learning our quality criteria (expressed by the heuristics) may bias the ordering and let the participants better distinguish bad and good experiences. Future work should pose this question before the experiment.

Usability issues with HeRA might have influenced the results, e.g., some students complained that the Tab key did not work to navigate between text fields. To mitigate a negative impression, we warned the participants in advance about these issues and made clear that the editor was only a prototype.

Overly simplistic heuristic critiques may have negatively influenced the participants’ opinion towards their helpfulness. For example, in case of Heur. 4, the warning often did not vanish, even if an emotion was apparent but was not on the emotion words list. Nevertheless, 70% of the participants rated this critique as (rather) helpful. It did not significantly diminish the overall perceived helpfulness of our concept (80% positive).

Based on a positive indication of this preliminary evaluation, future work should evaluate such concepts in an industrial setting in a longitudinal study as well.

For such future evaluation, we plan to use more sophisticated natural language processing techniques in our heuristics.

VI. SUMMARY AND OUTLOOK

In this paper, we presented a concept to support writing ad-hoc experiences by providing feedback based on heuristic critiques. We defined quality aspects for a good experience (completeness, observability, readability, traceability, and verbosity) and derived 11 heuristic critiques from them. Our evaluation indicates that this approach can improve the overall experience quality and increase the willingness of experience bearers to share their experience. Both effects would be especially valuable in distributed software development. Future research needs to quantify these effects and we hope that others are able to build on our work. We are certain that practitioners would profit from integrating our concept in their experience and knowledge management initiatives and are willing to share our experience beyond the scope of this paper.

REFERENCES