Empir Software Eng (2014) 19:1921–1955
DOI 10.1007/s10664-013-9268-6

Reporting experiments to satisfy professionals' information needs

Andreas Jedlitschka · Natalia Juristo · Dieter Rombach

Published online: 11 August 2013

Abstract Although the aim of empirical software engineering is to provide evidence for selecting the appropriate technology, it appears that there is a lack of recognition of this work in industry. Results from empirical research only rarely seem to find their way to company decision makers. If information relevant for software managers is provided in reports on experiments, such reports can be considered as a source of information for them when they are faced with making decisions about the selection of software engineering technologies. To bridge this communication gap between researchers and professionals, we propose characterizing the information needs of software managers in order to show empirical software engineering researchers which information is relevant for decision-making and thus enable them to make this information available. We empirically investigated decision makers' information needs to identify which information they need to judge the appropriateness and impact of a software technology. We empirically developed a model that characterizes these needs. To ensure that researchers provide relevant information when reporting results from experiments, we extended existing reporting guidelines accordingly. We performed an experiment to evaluate our model with regard to its effectiveness. Software managers who read an experiment report according to the proposed model judged the technology's appropriateness significantly better than those reading a report about the same experiment that did not explicitly address their information needs. Our research shows that information regarding a technology, the context in which it is supposed to work, and most importantly, the impact of this technology on development costs and schedule as well as on product quality is crucial for decision makers.

Keywords Software manager · Information needs · Technology selection · Experiment · Reporting

Communicated by: Per Runeson

A. Jedlitschka (✉) · D. Rombach
Fraunhofer Institute for Experimental Software Engineering, Fraunhofer-Platz 1,
67663 Kaiserslautern, Germany
e-mail: andreas.jedlitschka@iese.fraunhofer.de

N. Juristo
Universidad Politécnica de Madrid, Campus de Montegancedo, 28660 Boadilla del Monte, Madrid, Spain

D. Rombach
Technische Universität Kaiserslautern, Postfach 3049, 67653 Kaiserslautern, Germany

1 Introduction

Aiming at solving practical problems, researchers develop a variety of new technologies (the term technology refers to technique, method, and tool (Basili et al. 2001)) in the hope of industry-wide adoption. The motivation of the SE research community is to provide decision makers with relevant state-of-the-art technologies they can choose from to reduce risks that might jeopardize the success of their software projects. For years, it has been argued that decision makers in industry hesitate to introduce new technologies if evidence about their benefits and risks is not available, not convincing (Pfleeger and Menezes 2000), not communicated in the right language (Glass 2006), or if they lack relevance and rigor (Ivarsson and Gorschek 2011). A major ambition of empirical SE research is to provide the decision makers in software development with the type of evidence about technologies they need to support informed decision-making when introducing new technologies.

Although the number of individual empirical investigations of technologies as well as secondary studies (e.g., syntheses) reported in journals and at conferences is steadily increasing, few success stories of technology adoption by industry have been reported where decisions were based on such empirical evidence. Taking into account the available literature, empirical SE research presently seems to enjoy only low recognition in the software developing industry. This issue and the accompanying loss of potential were identified, e.g., at NASA (Hinchey et al. 2006). For the area of software inspections, it was shown (Ciolkowski et al. 2003) that although success stories and empirical evidence regarding their usefulness had been available for some years at that time, they were not being widely used in industry. However, some years later, Rombach et al. (2008) described successful examples of technology adoption in the area of inspections, based on the results of empirical studies in combination with managerial support.

This raises the question of whether researchers provide the "wrong" kind of evidence (i.e., evidence that software managers are not interested in), the wrong format (i.e., publications that do not easily address the software managers' information needs), or whether they communicate evidence through the wrong channels (i.e., addressing the wrong people or using publications that software managers do not use as information sources). In that respect, Pfleeger (1999) summarizes her findings as follows:

> "Technology transfer in SE involves more than a new idea or evidence that it works."

Empirical research asks professionals to make a change (to improve the software development process, to use technology A instead of technology B). In the software process improvement community, it is widely accepted that the decision to make such a change requires higher-level management commitment, (e.g., Dybå 2001). So, researchers interested in applied research have to realize that they have to primarily address software managers. If they do not convince software managers in their role in the decision-making process, their research results will not be considered in industrial practice. Nevertheless, evidence from empirical SE is certainly not the only source of information for a decision maker and should not be overestimated.

Thus, in order to be considered in the decision-making process, results from empirical SE research have to contain information that is relevant for and understandable by the decision makers. In this paper, we describe how we built an empirical model of software managers' information needs. Based on that model, we propose **empirically founded guidelines for reporting results from experiments to software managers**.

In this research, managers comprise both software managers and senior managers. We consider project managers, software quality managers, and software process managers as

software managers. In the decision-making process, software managers are the ones who often initiate the process, collect relevant information, prepare and drive decision-making. However, the final approval of a decision concerning the introduction of a technology is made by senior managers (Jedlitschka et al. 2007).

To set the context of our research, we consider the following scenario. A software manager at company X identifies the need to reduce the number of defects in the final product. Company X is highly innovative and develops embedded systems using a model-based development approach. Company's X maturity was assessed to be between CMMI levels 2 and 3 (continuous). In particular they apply decision analysis and resolution practices that require following a formal, informed decision-making process, including the identification of relevant alternative solutions. Therefore, the software manager starts to collect information about potential solutions with the purpose of satisfying both his/her own information needs and those of senior managers.

Acknowledging that synthesized evidence would be the perfect source of information, we claim that in the absence of appropriate, synthesized evidence, results from individual studies could be a valuable source of information.

The model, we are proposing here shows the researcher what key information is required by software managers. So it serves two purposes. First, following the statement "Selling technological ideas requires an effective marketing strategy—one that understands its audience" by Pfleeger and Menezes (2000), researchers might learn how to report information in a way suited for software managers. Second, the software manager will benefit, because the required information shall be available. In addition, the model could serve as a checklist, which helps the software manager formalizing the information needs.

The paper is structured as follows. In Section 2, we present the background to our research. In Section 3, we discuss the research methods we employed. We summarize how we elicited managers' information needs in Section 4. In Section 5, we compare models of software managers' and senior managers' information needs before describing how we developed the extended model of software managers' information needs and what it is about in Section 6. We compare our proposal with findings from the literature in Section 7. What information needs to be reported when addressing software managers is described in Section 8. Our proposal is validated in Section 9. Threats to validity are discussed in Section 10. In Section 11, we present the conclusions of our research and an outlook.

2 Background

Taking the standpoint of software managers, Glass (2004) asks for help from research: "Here's a message from software managers to software researchers: We (software managers) need your help. We need some better advice on how and when to use methodologies". In his keynote at the Empirical SE and Measurement Conference 2008 (ESEM2008), Hönninger, vice-president of the corporate research division at Robert Bosch AG, asked which of the 200 currently available requirements engineering methods his organization should apply. He asked research to provide support, especially with regard to empirical studies, for finding the most appropriate ones.

Turner (2004), acknowledging the difficulties in really responding to these demands, stated that empiricism (if applied in a goal-oriented manner and not for the pure sake of quantification) can help to answer the following questions regarding the "value" of a technology: What are the real costs, what is the benefit, what is its origin, in which context it is applied, what is the latency, and what might be the barriers.

The answers to these questions will help to answer the overall question: What is the probability of successfully implementing and running the technology? Prior to any implementation, a positive decision is required based (in the best case) on systematic assessment of the alternatives, e.g., as proposed in the context of the CMMI process area *Decision, Analysis, and Resolution* (Chrissis et al. 2011).

In his position paper for the Dagstuhl seminar on Empirical SE in June 2006, Glass coined the term "communication chasm" for the gap that exists between academia and practice in the computing fields (Glass 2006): "Academics and software managers pay little attention to each other, do not read each other's literature, and tend to have different vocabularies. As a result, neither group has much influence on the other".

From these statements, we conclude that when a technology is to be selected, there might be a risk that only a **limited set of potential alternatives** is taken into consideration. This set is restricted to those alternatives for which information regarding the selection criteria is available. Furthermore, we conclude that there is a gap in the communication between research and software managers, especially with regard to empirical evidence about the pros and cons of software technologies under realistic circumstances. To help close this gap, an understanding of what has to be reported, in which way and to whom, is needed.

A **literature review regarding software managers' information needs** did not reveal any study that directly addresses the research question of what information is required by software managers to consider a technology in a selection process. The literature review was conducted in 2006 (1968-June 2006). We used software and manager and "information need" as keywords to search the following databases: ACM digital library and IEEExplore. The search resulted in 211 papers. None of them did address information needs with regard to technologies in the context of decision-making; rather, they address very specific topics such as cost estimation or architectural considerations. However, we found four proposals that provide a starting point. Regarding information needs in the context of SE technology selection, Vegas (2002) investigated which questions decision makers in the area of testing would want to have answered if they had to choose among different testing techniques. The criteria for the decisions are not explicitly discussed. The survey yielded 191 questions, which were categorized into 34 categories. Rogers (2003) describes four aspects that impact the relative speed of technology adoption: (1) the communication channels, (2) the social system of the potential users, (3) the effort required for technology diffusion, and (4) the technology's attributes.

Pfleeger (1999) lists a set of questions one should get answered sufficiently before considering a given technology as a good candidate. Pfleeger further provides a list of technology transfer inhibitors and promoters, which are arranged according to three categories: technological, organizational, and evidential. The list incorporates information needs, success factors, and general criteria. Reifer (2003) summarizes critical success factors that managers should obey when introducing state-of-the-art technologies. These factors mainly refer to technology maturity, that is, the availability of appropriate evidence, support, and an organization's readiness.

In a **literature review regarding technology selection**, nine approaches were identified that could serve as a basis for modeling technology-related information. The literature review was conducted in 2005. We started with known papers in the area of (technology) reuse: Prieto-Díaz (1985), Basili and Rombach (1991), Maiden and Rugg (1996), SEI (1997), and Henderson-Sellers et al. (1998), and performed a forward search for papers citing the initial ones in the context of technology selection. These approaches cover code reuse (Prieto-Díaz 1985), problem-solution pairs (Henninger 1996), characterization of specific techniques (Maiden and Rugg 1996, Henderson-Sellers et al. 1998, Vegas 2002), characterization of

technology application experience (SEI 1997; Birk 2000; Jedlitschka et al. 2005), and comprehensive reuse (Basili and Rombach 1991). Except for Prieto-Díaz' approach, which is quite specific for the reuse of software code, all other approaches could be appropriate for technology selection. The approaches specifically dedicated to technology selection, like those proposed by Birk, Vegas, and Jedlitschka et al., can be traced back to the original approach by Basili and Rombach. They extend the original approach for their specific purposes.

Birk's approach and its extension by Jedlitschka et al. are thought to be comprehensive, thus independent of any specific technology. However, they mainly focus on the contexts under which a technology has been (successfully) applied. In addition, the resulting collection of experience is specific for the purposes of a particular organization. Vegas's approach provides support for selecting testing technologies. The information needs were extracted from technology customers and technology providers. The focus of this schema is on the technology, the environment, and to some extent, on costs, benefits, and inherently on quality, which is due to the nature of testing as a quality assurance measure.

The schemas presented above focus mainly on technical and environmental aspects of the technologies. Most of them do not explicitly incorporate the business perspective. However, those approaches provide a starting point for the elaboration of software managers' information needs, especially incorporating the business perspective. In addition, most of the proposed models appear not to be based on empirical evidence.

Reporting guidelines are expected to support a systematic, standardized description of empirical research, thus improving reporting to support readers in finding the information they are looking for, understanding how an experiment is conducted, and assessing the validity of its results. The need for improved and preferably standardized reporting of controlled experiments in SE has been mentioned by several authors (e.g., Lott and Rombach 1996, Pickard et al. 1998; Runeson and Thelin 2003; Shull et al. 2003; Vegas et al. 2006; Wohlin et al. 2003; Jedlitschka and Ciolkowski 2004; Sjøberg et al. 2005).

Other disciplines, such as medicine and psychology, have also experienced problems with regard to extracting crucial information from empirical research and with regard to insufficient reporting. They have achieved various improvements through standardization and by instantiating reporting guidelines, for example for randomized controlled trials in biomedical research (Moher et al. 2001; Altman et al. 2001), for clinical practice guidelines (Shiffman et al. 2003), and for empirical results from psychological research (APA 2001; Harris 2002).

In the field of SE research, several reporting guidelines have been proposed. In 1999, Singer (1999) described how to use the "American Psychological Association (APA) Styleguide" (APA 2001) for publishing experimental results in SE. In 2001, Kitchenham et al. (2002) provided their first version of initial guidelines on how to perform, report, and collate the results of empirical studies in SE based on medical guidelines as well as on the personal experience of the authors. In 2003, Shaw (2003) provided a tutorial on how to write scientific papers, including the presentation of empirical research as a special case. Additionally, standard text books on empirical SE, such as Wohlin et al. (2000) and Juristo and Moreno (2001), address the issue of reporting. Wohlin et al. suggest an outline for reporting the results of empirical work. Juristo and Moreno provide a list of "most important points to be documented for each phase" in the form of "questions to be answered by the experimental documentation". Based on experience from evidence-based medicine, Kitchenham (2004) proposed a guideline for conducting and reporting systematic reviews. In order to arrive at a common reporting guideline for controlled experiments, Jedlitschka et al. (2008) compared the different proposals and consolidated them iteratively by also incorporating feedback from the community. Specific guidelines for reporting case studies were published by Runeson and Höst (2009).

As of today, it seems to be a common understanding that guidelines have to address the needs of different stakeholders (i.e., researchers, software managers, funding organizations) (Kitchenham et al. 2004; Jedlitschka 2007; Jedlitschka and Briand 2007). However, a researcher aiming at replicating a study certainly requires more details about the design than a software manager. This issue was already identified earlier by other research areas. Kvale (1996), for example, wrote "…, researchers in system evaluation and market research have been well aware of the effects of the form of their reports on the intended audiences…".

To summarize, the coverage of guidelines for certain types of studies has recently become more complete, but the existing guidelines seem to be explicitly tailored to the specific needs of researchers and do not address software managers.

3 Research Method

Figure 1 shows the research process followed to elicit managers' information needs and to develop the respective models. Six empirical studies were performed to build the model, one of which (the pilot) was conducted with participants from research (cf. *1), whereas the others were conducted with participants from industry (cf. *2). For the first set of studies (from pilot to Study 3), we used qualitative data analysis techniques (cf. *3); for the second set (Study 4 and Study 5), we employed quantitative methods (cf. *4). As intermediate results we obtained models of both software managers' and senior managers' information needs. Finally, we combined the models, which yielded a model that characterizes the information addressing managers' needs in the decision-making process, especially when selecting an SE technology. In the evaluation phase, the resulting model was investigated with regard to its effectiveness.

In the following, we describe the procedure for each of the studies (from the pilot study to Study 5).

To explore the information needed by software managers for software technology selection, we performed a **pilot study** aimed at eliciting technical and content-related requirements for a system to support empirically based selection of software technologies

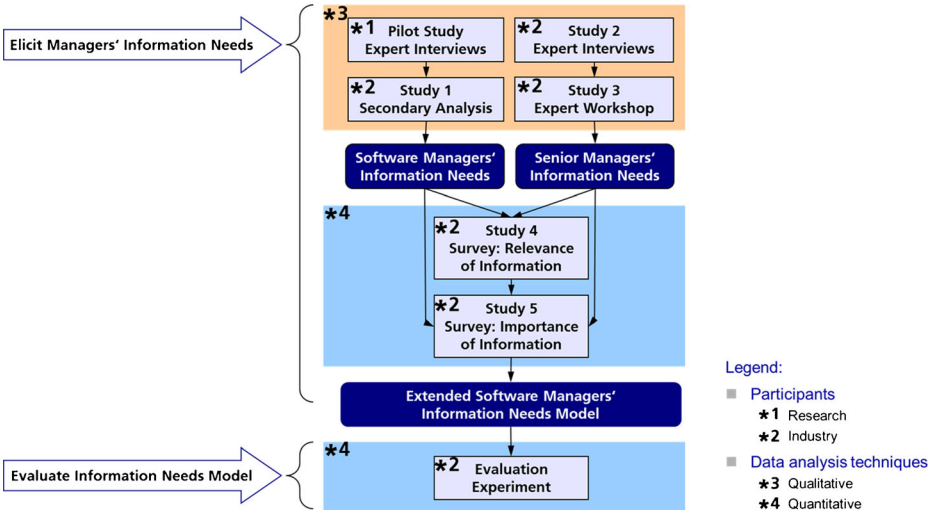

Fig. 1 Research Process

(Ayari et al. 2004; Jedlitschka and Pfahl 2004). The study was designed as scenario-based, structured expert interviews. The results were aggregated in a grounded-theory-like approach, yielding an indication of potentially relevant categories of information.

The next study was a **secondary analysis** (Study 1) of data from a survey by Vegas (2002) aimed at further eliciting information needs explicitly from software managers and developers (secondary analysis as defined in Marshall (1998)). They were asked which questions they, as decision makers, would ask technology providers. After carefully examining the layout of the original study with regard to its generalizability concerning technology selection in general (especially the scenario and the task presented to the participants as well as the participants representing the technology consumer), we generalized the participants' responses (questions) to technologies other than testing techniques.

For the analysis, we used the categories from the pilot study to group the questions that we thought belonged to a common topic before further extending the list of categories in the same way. Then we aggregated the questions into elements for the respective category. To get an indication of relevance, we counted how often each element was addressed by questions. For instance, we considered those aspects as less relevant that are not supported by any questions from the group of software managers.

In order to further corroborate the findings from the pilot study and Study 1, we conducted **expert interviews** (Study 2) with senior managers in industry. The rationale for using senior managers was that they might have a more general view, less technology-oriented and more business-success-oriented. The aim of this study was to identify the top five key performance indicators (KPI) for product development used in industrial practice. KPI are a tool for monitoring an organization's level of success with regard to its specific goals. The monitoring and reporting of KPI requires the presence of certain kinds of information, in the sense that they represent the information needs of management.

Our procedure for analyzing the expert interviews lies between meaning condensation and meaning categorization, but also follows the approach of using common sense in combination with quantitative and textual methods (Kvale 1996). Our approach to condensing and categorizing meaning was inspired by the grounded theory approach developed by Strauss and Corbin (1998). That is, we took the data from the first interviews and identified initial categories. In most cases, this was done by identifying the most common wording. Then we incorporated the data gained from the next interviews in the following way. If the data was considered to be already reflected in some existing categories, the inherent counter for this information was incremented to reflect that this category was more evident. In case the data was not yet reflected, a new category was added. In accordance with grounded theory, this procedure was performed until all new knowledge was categorized.

The results were further evaluated by means of an **expert workshop** (Study 3). For assessing the KPI, we uncovered a flip chart with a prepared table of strategic KPI. Using a brainstorming modus, we asked each participant to reflect on whether the presented KPI are used in their organizations or to nominate those that are currently used. If the KPI was not on the list yet, it was included after a brief description given by the participant. The number of KPI was not restricted and participants were allowed to continue adding further KPI. Each nomination was marked with a colored dot identifying the organization that contributed it. For the ranking, each organization was asked to mark the top five KPI (either nominated by themselves or by others) with an individual symbol. The ranking was done by counting the nominations: first the nomination of a KPI under the top five, and then number of its nominations as strategic KPI.

We now had two models: a model of software managers' information needs (SWIM) from the pilot study and Study 1 and a model of senior managers' information needs (SIM)

from Study 2 and Study 3. Before we could combine them into one common model, we needed to investigate whether, in addition to the already observed difference with regard to the level of information, there were other significant differences with regard to relevance. We decided to conduct quantitative studies (Study 4 and Study 5) by means of surveys, which allowed us to incorporate a larger number of participants. Thus the objective of the studies was to quantify the relevance and importance of certain kinds of information in the context of technology selection. We define relevance of information as: Information is relevant if it has the potential to contribute to answering a question at hand (i.e., in the decision-making context). We consider information as being important if it is required in the decision-making process. If importance is bound to a specific context, we consider importance and relevance as synonyms. We consider importance as the higher-level term.

For Study 4, in addition to the kinds of information required for the selection process, we investigated the preferred sources of information, identified the decision makers in charge of technology selection, and provided an initial model regarding the relevance of information. In addition, we collected demographic information, including information about the respondent's organization (business domain, size of the organization and the respondent's group/business unit, typical duration and kind of projects). Furthermore, we collected information about the respondents themselves, such as their role in the organization.

The purpose of Study 5 was to further investigate the importance of information used in support of technology transfer. In addition to quantifying the importance of several kinds of information, we contextualized the information needs with regard to certain strategies and preferred sources of information, and consolidated the role of the decision makers as identified in Study 4. Regarding the information needs, we wanted to know how important certain kinds of information are in the decision-making process. We collected additional information as described for Study 4.

Finally, we evaluated the resulting model of software managers' information needs with regard to its effectiveness in a controlled experiment with participants from industry.

4 Eliciting Managers' Information Needs

The purpose of this section is (1) to empirically identify the information needs of software and senior managers in order to support software managers in selecting appropriate software technologies, and (2) to discuss and justify what kind of information a researcher must provide and why it is pertinent for the technology selection process.

Software managers' information needs result from the need to choose the technology that promises the most benefits and the least risks when applied in the context of one's organization, process, or project. Comparable to a medication, it has to be assumed that under certain circumstances there might also be some negative effects (i.e., explicit risks), which would have to be known in order to be minimized.

In our **pilot study**, the participants (nine senior researchers from Fraunhofer IESE) assumed the role of software managers, representing domains of Fraunhofer IESE's customers, e.g., automotive, avionics, banking, insurances, and primary software. To have a specific and easily understandable example, the scenario was restricted to the selection of appropriate reading techniques in the context of software inspections.

As part of the content-related (non-functional) requirements, an initial set of questions that should be answered by the system was identified (cf. Ayari et al. (2004) and Jedlitschka and Pfahl (2004)):

- Which technologies are available (information on a highly aggregated level)?
- How **effective/efficient** is a certain technology with respect to which quality aspect?
- What are the **costs for introducing** the technology?
- What are the **costs for applying** the technology?
- What is the **impact** a single technology has on the whole development process?
- What is the validity of the empirical results associated with a particular technology?
- In which **context** (environment) can the technology be applied/has it been applied?
  - E.g., kind of system, programming language, process stage, description of the process in which a technology shall be applied.
  - Preconditions that have to be fulfilled prior to the application of the technology (e.g., skills, kind of documents available).

The categories, though quite generic, provide indications that were used as input for the following studies.

For **Study 1**, we reused data from 13 participants from industry (cf. Vegas (2002) for the original study and Jedlitschka (2009) for the secondary analysis). Vegas used two different groups of participants, technology customers and technology producers, to investigate which information software managers ask for when selecting testing techniques. Vegas presented a scenario to the participants (i.e., technology customers) explaining that they are responsible for testing a certain software product. During the selection process, they were to ask the technology provider questions regarding any information they wanted. The participants were asked to write down the questions they had. Only the technology customer group was considered to be relevant for our purposes; 13 participants remained, who came from the domains of avionics, electronics, medical systems, and primary software. We found that several questions submitted by Vegas' participants can be interpreted in a more general way (not only for testing) without any change (97/154), for instance, "What is the environment where it can be used?" Others had to be generalized, for example by replacing the word "test" (21/154) with "technology" or just removing it (e.g., "When can the technology be applied?"). A set of questions had to be rejected because they were specific for testing and not applicable to other technologies (46/154) (e.g., "Does the technique create repeatable tests?").

The secondary analysis of Vegas' raw data yielded 12 categories of information that were used to assess the appropriateness of a technology from a software manager's perspective. While developing the categories and elements of the model, we had to keep in mind that the model needs to be generic in order to allow domain-specific adaptations. For instance, a software manager in the avionics domain is much more concerned with technical safety than a software manager of an Internet start-up, who may have concerns regarding data security. For this example, we introduce a generic category *Quality* and respective elements that can be used to specify both information needs. Although we considered *Risk* as an important aspect, we did not include it because risks can be related to any of the given categories and elements of the model. Whether the information given within an element is a risk or an opportunity depends on the context of the decision.

Table 1 shows SWIM in the context of technology selection, as obtained from this study. SWIM focuses on the attributes of a technology and the context in which it might be used.

Table 1 SWIM: Software managers' information needs obtained from pilot study and secondary analysis

Category	Element	Attribute
Training	Need	
	Effort	
	Technology's ease of use	
Costs	Adoption	
	Application	
	Resources	
	Purchase	
	Maintenance	
	Effort	
	Impact on development costs	
Schedule	Adoption	
	Application	
	Impact on development (project) schedule	
Quality	Quality attribute	
	Quality metric	
	Impact on quality	
Return	Return on investment	
Result (generic)	Focus (e.g., cost, schedule, quality)	
	Metric	
	Metric definition	
	Measurable results	
	Other side effects	
Maturity	Reference (user)	
	Maturity	
Context	Application domain	
	Interdependency	
	Development environment	
	IT environment	
	Development process	
	Life-cycle phase	
	Paradigm	
	Type of project	
	Type of software	
	Input	Kind of document
		Format of document
	Output	Kind of document
		Format of document
	Qualification	Knowledge required for being able to apply the technology
		Experience required for being able to apply the technology
Tool support	Extent of automation	
	Extent of tool support	
	Tools that can be used	

Table 1 (continued)

Category	Element	Attribute
Support	Availability of support for the introduction	
	Availability of support for the application	
	Documentation	
Background	Alternatives	
	Basics	
	Classification	
	Complementary technologies	
Change	Extent of change required for current development process	
	Extent of cultural change	

This focus might be due to the type of managers used in these two studies. In addition to the traditional categories (cf. first column) of *Costs*, *Quality*, and *Schedule*, categories for *Context* and respective references for the technology were extracted. The number of elements in SWIM (cf. second column) describing context, does, on the one hand, underline the importance of this information in the course of technology selection, but on the other hand also indicates that organizations consider different influencing factors as relevant. Some categories, such as *Output*, have additional attributes (cf. third column), and cannot be considered to be complete yet. For example, in order to be consistent with the category *Input*, the category *Output* needs to be extended regarding the kind and format of documents. In addition, the categories *Costs*, *Schedule*, and *Quality* can be extended with the more general category *Results*.

In Study 2, expert interviews were conducted with 16 senior managers from 13 innovative industrial organizations in Germany; seven from the software domain, five from the automotive domain, and one from the engineering domain. Using semi-structured interviews, we asked them for the KPI they are using to assess the level of success of their organization as a whole and of their product development in particular. From the quantitative analysis, we conclude that although there is no common agreement on the top KPI, the importance of specific initiatives showing their potential impact on critical KPI is unquestionable. The categorization finally yielded well-known categories: innovation, quality, schedule, and cost (in order of their importance measured by numbers of KPI mentioned for those categories) (cf. Jedlitschka (2009)).

Based on the results from Study 2, we conducted an **expert workshop** with six senior managers from five software developing organizations in Germany (Study 3). The participants came from the following industry sectors: telecom, finance, IT consulting, automotive (electronic), software (Internet). The aim of the expert workshop was to assess the KPI for product development, to identify influencing factors, and to consolidate senior managers' information needs. The workshop mainly confirmed the findings from the previous study, namely, that there is no common agreement on the most important KPI (cf. Jedlitschka (2009)). However, we were able to agree on a consolidated, ranked list of 21 KPI, shown in Table 2.

The discussion about possible categories of KPI yielded three obvious ones, namely cost, quality, and schedule. Productivity was discussed as an additional category. The current value of a KPI depends on the context factors. In most cases, the context factors are not known completely and, even worse, the influence of the context factors varies not only over time (e.g., type of organization, process maturity increases, average experience of employees

Table 2 SIM: Senior managers' information needs obtained from expert interviews and expert workshop

TOP KPI nominations	# nominations	KPI on strategic level
2	3	Customer satisfaction
2	2	Productivity
2	2	Product quality
1	4	IT cost quote
1	3	Customer loyalty
1	2	Employee commitment
1	2	Degree of innovation
1	2	Cost per customer
1	2	Market share
1	2	Success of project (in-time, in-budget, in-function)
1	2	System availability
1	1	Order book (number and status of orders)
1	1	Best Practice
1	1	Competence
1	1	Utilization attributes (e.g., number of hits for certain tools)
1	1	Project management KPI
1	1	Qualification
1	1	Re-scopes
1	1	Return on investment
1	1	Time to market
1	1	Number of concurrent projects

decreases because of high turnover) but from company to company (e.g., some are more mature than others), and from domain to domain (e.g., the legal circumstances are totally different for medical systems than for entertainment systems). So it was found that interpreting KPI appropriately without knowing the context is impossible, underpinning the importance of the category context as already identified in the studies with the software managers. Column three of Table 2 presents the model of senior managers' information needs.

At this point, we had two heterogeneous models: SWIM and SIM. Before combining them into one common model, we needed to investigate whether there are significant differences with regard to interpretation, relevance, and importance.

We conducted two quantitative studies by means of surveys (Studies 4 and 5) to quantify the relevance and importance of certain kinds of information in the context of technology selection.

The first survey (Study 4) was conducted at the beginning of 2006 and was completed by 92 decision makers (cf. Jedlitschka et al. (2007)). Most of the participants worked in the areas of software development, IT service providers, and manufacturers of electronic components. The instructions for filling out the questionnaire had the following format: "Please state for each information item to which extent it would have been not relevant, less relevant, relevant, or very relevant for the decision". The results from Study 4 show that decision makers primarily require information about the impact of a particular technology on

a product's quality, costs, and development time (schedule). Although these three categories of information are most important when making the decision to introduce a technology, it seems that decision makers in industry do not have sufficient suitable information at hand. Information about the cost-benefit ratio of the technology is considered relevant, as is the domain from which the information is obtained. Less relevant, for many decision makers, is information on pure return on investment and whether a technique complements another one. Concerning information channels, we found that colleagues are considered the most relevant source of information, followed by textbooks, industry workshops, the Internet, and software managers' journals; scientific journals may also play a role. Less relevant information sources include consultants, conferences, and literature (other than textbooks and software managers' journals). But it seems to be relevant that the information stems from the same domain. While these results corroborate the findings of Rainer et al. (2003), they do not appear particularly surprising. Still, they serve as a step towards confirming what researchers believe regarding software managers' information needs. Although decision makers mainly rely on personal reference through a colleague to gather information, the findings of this survey also show that research can still reach software managers through textbooks and software managers' journals.

The results provide an initial set of information relevant in the context of technology selection. It seems that all information is of similar relevance. We hypothesize that the relevance of certain kinds of information regarding, for example, costs, quality, or schedule varies depending on the situation at hand (for example, depending on the strategy, but also driven by short-term objectives, or determined by time pressure in a current development project). The situation comprises, for example, the strategic orientation of the organization on the highest level, the requirements and status of a project on a lower level, the (collective) experience of the organization, and environmental factors.

This study quantitatively supports the relevance of the following categories of information:

- Product quality
- Costs
- Development time (schedule)
- Cost-benefit ratio
- Application domain

The second survey (Study 5) was conducted in 2008 and was completed by 48 decision makers (cf. Jedlitschka (2009)). The participants came from the automotive sector, followed by the telecommunications and communications sector, the production sector, and the finance and insurance sector. We asked about the importance of certain kinds of information for the decision-making process, which was fixed in a pre-defined context. The scale was (absolutely required, very important, important, not very important, not important, irrelevant, don't know). In addition, we gathered data to rank the indicators by making pair-wise comparisons and asking for agreement on a five-point scale.

The results show that there are differences regarding the importance of information. Information regarding the impact of a technology on a product's quality is most important for the responding software managers. Nevertheless, as we conclude from the analysis of the importance of missing information, software managers often also require information with regard to the other categories of information, such as impact on project schedule and project costs.

When analyzing the elements of the categories quality, schedule, and costs, we found that our respondents ranked information regarding a technology's impact with regard to its importance as follows:

1. Impact on productivity,
2. Costs for the application of the technique,
3. Impact on product reliability,
4. Impact on project costs, respectively its potential to reduce these, and
5. Impact on time to market

Regarding additional factors, we found that information regarding the domain, the knowledge required for being able to apply the technology, and the support available for the technology is important.

The contribution of Studies 4 and 5 is the quantitative justification of the importance of certain kinds of information in the context of technology selection. In terms of the importance of information presented to the participants, there was no significant difference between software managers and senior managers. The studies further yield the insight that due to the heterogeneity of information needs, it is not possible to reduce the model to a few most important categories. The absence of any significant difference between software managers' and senior managers' information needs in terms of content facilitates the development of a common information needs model.

Regarding the responsibility for selecting technologies, the picture is quite heterogeneous. The decision maker may be a project manager who is allowed to decide upon a testing strategy, a quality or software process manager designing general software development processes, or senior management operationalizing strategic decisions. As in most cases the decisions have to be approved by senior management (either before the formal decision or after the piloting), the people preparing the decision need to get supportive arguments for the technology. Following the findings from our surveys, these arguments go far beyond pure technical descriptions. They have to complement the business or project objectives and related strategies.

5 Comparing Senior Managers' Information Needs and Software Managers' Information Needs

Following up on the results of our studies, we now merged SWIM and SIM with the intent of obtaining one common information needs model. We compared the models to investigate whether it is possible for software managers to align their activities with business objectives and to report to their senior management. To achieve this goal, software managers first need to satisfy their own information needs. When this information is available, it needs to be integrated in order to fulfill the information needs of senior managers. Taking into account results from software process improvement research (e.g., Dybå 2001), which showed that management commitment is among the most important success factors for any activity, we assume that if software management's contribution to the success (business orientation) of the organization is made transparent, management commitment might be obtained more easily. Management commitment is required to release the necessary resources.

Our empirical studies showed no differences between the different groups of managers from our samples with regard to the general information they would request when selecting technologies. This led us to the assumption that there are categories of information that are

understood in a similar way by different kinds of managers. However, if a senior manager assesses a project, he will probably refer to the whole product development project. In some cases, this might also include developments not related to software. In the automotive industry, for example, many other parts belong to the final product; in some cases, software is seen as part of electronics. Thus, senior managers and software managers might use the same terms (categories), such as costs, but at different levels of abstraction. However, the relationship between the categories is quite direct. There are further categories where this category of information is not really available in one of the models. For example, customer satisfaction can be seen as such a category. While it is available in the senior managers' information needs, we were not able to find this category in the software managers' information needs, though it is implicitly available through other categories. In the case of software as part of a more complex product, there are actually two different customers, the internal customer (by whom the software is integrated) and the external customer (for whom the whole product is developed).

In summary, the possibility to match the information needs leads us to the conclusion that when the information needs of software managers are satisfied, the potential exists that the information needs of senior managers regarding software development could also be satisfied. From a technology provider's (or researcher's) point of view, this means that it might not be necessary to provide two different sets of information. The differences regarding the level of aggregation can be summarized as follows: Senior managers need highly aggregated information on the overall development or the business, whereas software managers need more specific information on a specific development project, especially the software part.

While analyzing which elements of software managers' information needs were used, we also found mostly elements related to the categories costs, schedule, and quality. Senior managers might not be interested in the details of a technology; however, information regarding the context in which the technology has been applied or information about reference users can also help them to assess the risk of introducing a technology.

The comparison of the information needs of software managers and senior managers yielded clear matches of the information needs regarding impact and risks, whereas for the information needs regarding the technology and the context, the matches were limited in pure numbers. We argue that this is due to the different responsibilities. Software managers, who are mostly concerned with the preparation of a decision, need to know much more technical details in order to judge whether a technology fits into their context. Senior managers are much more concerned about the impact and the risks associated with the introduction (application) of the technology.

6 A Common Model of Managers' Information Needs

Based on the comparison of the two models, we built a common model starting from the software managers' information needs and complemented it with elements from the senior managers' information needs. The motivation to start with the SWIM lies in the decision-making process. The software manager will be the one collecting the information, trying to satisfy both his/her own information needs and those of senior managers. So the software manager will be the beneficiary of the model. The categories of the resulting model of managers' information needs are:

1. Information regarding the technology.
2. Information about the context in which the technology has been applied.
3. Information regarding the technology's impact.

Table 3 shows the resulting model. The first column gives the name of the category, the second shows the element, the third column gives the attributes, the fourth column gives the reference to the model where this information comes from, and the fifth column provides examples to support the understandability of the model (i.e., example operationalization of the attributes). Some additional elements have to be included that do not come directly from the elicitation of the information needs, such as the name of the technology, but that are required in order to make the model self-contained (marked as administrative (ADM) in column four (origin)).

Empirical evidence from our studies supports the importance of the context in which a technology was used (or evaluated). Software managers from our samples want to have information about the domain and the circumstances under which the technology has shown its benefits. For example, information regarding (1) the domain in which the technology has been applied, (2) the knowledge required for being able to apply the technology, and (3) general preconditions that have to be fulfilled before the technology can be applied seem to be crucial for decision makers. To determine the category impact, we combined the categories information regarding a technology's impact and associated risks (a risk is considered to be a potential negative impact).

While matching the elements of SWIM indirectly to those of SIM, we discussed potential relationships between them. SIM elements are incorporated through several aspects, for example the element *impact on the process outcome*.

7 Comparing the Proposed Model to the Literature

To demonstrate the contribution of our research, we surveyed the literature, particularly in the area of reuse and technology selection (Jedlitschka 2009), starting with the comprehensive reuse model proposed by Basili and Rombach (1991) (cf. Table 4).

Inspired by Basili and Rombach's (1991) characterization of an experience package, we investigated whether a similar attribute was used by other authors. If this was the case, we added their name to the list in column 5. In some cases, we had to decide whether the meaning is similar enough to represent either the same category or a new category, or whether a more generic term would summarize them better (e.g., characterization:purpose). Further, we found that the organization of information on the level of the elements differs quite widely. For example, the name of the element "interface" of the literature model was introduced by Basili and Rombach (1991). In our model, the element "interface" is represented by its attributes; most of them are grouped into the elements pre-requisites and output.

We found that for some attributes there is quite a consensus in the literature (e.g., purpose), while there are only few references for others (e.g., tool costs). Table 5 compares the proposed empirical model with the aggregated findings from the literature. The first three columns correspond to our model, whereas columns four and five show information items that have been proposed in the literature.

An asterisk (*) in Table 5 indicates elements of the literature model that we consider to be too generic to be matched with any of our elements, e.g., if detailed, benefits can easily be matched with any of the impact elements of our model.

The proposed model consists of 47 attributes, of which 19 have not been explicitly reported in the literature. 39 of the attributes from our model can be matched to attributes proposed in the literature. There is quite a good match with regard to the technology and context categories (i.e., our model contributes with two new attributes), whereas for the impact category, the match is quite weak (i.e., our model contributes with 17 new attributes).

Table 3 Model of managers' information needs

Category	Element	Attributes	Origin	Example
Technology	Name	Name	ADM	Checklist-based Reading
		Abbreviation	ADM	CBR
	Type	Type	ADM	Technique
	Description	Short description	SWIM	"Inspection method using checklist-based reading to find defects in requirements documents"
		Long description	SWIM	"…", or a reference to literature about inspection methods.
		Belongs to family	ADM	E.g., CBR belongs to reading techniques
		Complements	SWIM	E.g., structural testing and functional testing
		Literature	SWIM	Reference to literature etc.
		Background	SWIM	The technology was developed by …
		Alternatives	SWIM	Perspective-based reading is an alternative approach
	Applied in	SE phase	SWIM	Conceptualization, Analysis, Design, Development, Evolution/Maintenance, Reengineering
	Applied on	SE object	SWIM	Requirements, Architecture, Code, Process Model, …
Context	Application domain	Industrial sector	SWIM/ SIM	Automotive, telecommunication, electronics
		Type of Organization	SWIM/ SIM	Small and medium-sized enterprise (100 employees)
	Pre-requisites	Qualification required	SWIM	Inspectors need to understand the format of the requirements documents.
		Training required	SWIM	Two person-days of initial training
		Experience required	SWIM	Student developers in their fourth year of studies/professional testers with 10 years of experience with XYZ testing.
		Input	SWIM	Requirements document
		Output	SWIM	Improvement suggestions
	Project	Size	SWIM	in person-months (e.g., 10 person months)
		Kind of software	SWIM	Embedded system, information system
		Type of project	SWIM	Maintenance project
	Environment	IT environment	SWIM	Tools, hardware
		Development environment	SWIM	E.g., Programming Language
		Development process	SWIM	E.g., V-Model, SCRUM.
		Paradigm	SWIM	Object-oriented
		Interdependency	SWIM	The technology cannot be used together with …
Impact	Impact on the development	Training	SWIM	Five inspectors got two days of training each in the new technique.

Table 3 (continued)

Category	Element	Attributes	Origin	Example
	(project) costs	Introduction	SWIM	For CBR, no costs were incurred for the introduction.
		Application	SWIM	Each reviewer inspected the documents for 2 h.
		Maintenance	SWIM	For CBR, no costs were incurred for maintenance.
		Total cost of ownership	SIM	The total costs of ownership are limited to the buy-in of an expert to train the developers.
		Project	SWIM	Given the positive ROI the project will gain after the first successful application of CBR by means of a lower defect-slippage rate.
	Return	Return on investment	SIM	The ROI is factor 2.
		Latency of ROI	SIM	ROI was obtained after the first application.
	Impact on the quality of the product	ISO 25010 SQuaRE	SWIM	ISO 25010 SQuaRE stands for the list of quality attributes that can be listed here. E.g., reliability of the product; comprehensiveness of a requirements specification document.
		Product	SIM	End user reported defects decreased by 80 %, defect slippage rate decreased by 20 % from requirements to design.
	Impact on the process outcome	Process outcome	SIM	Effort estimation accuracy
	Impact on SE object	SE object	SWIM	Effort estimation, requirements, architecture, code, process model, …
	Impact on the development (project) schedule	Introduction	SWIM	Developers have to participate in a two-day training session.
		Application	SWIM	Applying CBR costs 1 day of development time
		Latency time	SWIM	First effects will be observed after first application; with more experience on the technique, efficiency will increase.
		Project	SWIM	The project will gain in terms of less re-work.
		Time to market	SIM	There will be no effect on time to market.
	Impact on productivity	Productivity	SIM	Productivity will initially decrease because developers are not producing code.
	Latency in SE Phase	In SE phase	SWIM	Conceptualization, Analysis, Design, Development, Evolution/Maintenance, Reengineering
	Impact Description	Description	SWIM	The technology reduces the development time in the design phase by 20 %.
	Reference	Origin	SWIM	Organization X

Table 4 Structure and example of an experience package (cf. Basili and Rombach 1991)

Element	Attribute	Example
Object	Name	Sel_inspection.cleanroom
	Function	Certify appropriateness of design documents
	Use	Process
	Type	Inspection method
	Granularity	Design stage
	Representation	Informal set of guidelines
Object Interface	Input/Output	Specification and design document required/defect data report produced
	Dependencies	Assumes a readable design, qualified reader
Object Context	Application domain	Ground support software for satellites
	Solution domain	Cleanroom (Fortran) development model; step-wise, refinement-oriented design, statistical testing
	Object quality	High defect detection rate (e.g., > 1.0 defects detected per staff hour) wrt. interface faults

This might mean that the attributes for describing the technology and the context are quite well understood (at least after aggregating the different proposals from the literature). However, an understanding of the information category managers considered most important—impact and related elements—seems to be completely forgotten in the literature. With few exceptions (e.g., application costs), only very high-level attributes have been proposed in the research literature (e.g., benefit). Our model provides a comprehensive extension of the knowledge about the information needs regarding impact.

8 Reporting Experiments Following the Proposed Model

The purpose of this section is to show how the proposed model is used to extend reporting guidelines for experiments so that a researcher who reports a specific study can see which information is required in the report (for a detailed guideline for reporting experiments, see Jedlitschka et al. 2008). Based on our empirical model of managers' information needs, we propose putting special emphasis on the description of the technology as such, the context in which the technology is supposed to or where it is proven to work, and the impact the technology has on development costs and schedule, as well as on the quality of the product when writing a report about an experiment addressing software managers. Further information shall be provided regarding the risk of applying the technology. This is well in line with Glass' request made to researchers to tell professionals about which technologies should be used under which circumstances (Glass 2004).

The language is certainly another aspect related to providing information. As Glass (2006) puts it, there are two different languages spoken in research and industry. To support software managers in aligning their activities to business objectives, it is important for researchers to adequately report their findings using a language that is understood by software managers. This includes proper definition of the measurements, for example: What does efficiency mean; how is it defined; where is the focus? In addition, this is also true for the description of technologies. Without using a common characterization, that is, saying what the technology is about, in which phase of the development cycle it can be applied, etc., there will be little chance of it being accepted by software managers, who are not necessarily experts on the specific technology.

Table 5 Comparison of the proposed model with information items from the literature

Proposed Model			Information Items from the Literature	
Category	Element	Attribute	Element	Attribute[1]
Technology	Name	Name	Name	Technology name [BR91, Pri85, HSY98 (focus), Veg02, SEI97, MR96, Bir00, Hen96]
		Abbreviation		
	Type	Type	Characterization	Type [BR91, Hen96 (type of resource)]
				Use [BR91]
				Representation [BR91]
	Description	Short description	Characterization	Purpose [BR91 (function), Pri85, HSY98 (focus), Veg02, SEI97, MR96, Bir00]
		Long description	Description	Description [HSY98, SEI97, MR96, Hen96]
		Belongs to family		
		Complements	Background	Complementary Technologies [SEI97, MR96]
		Literature	Maturity	References [MR96, SEI97 (references and information sources), Veg02 (personnel)]
		Background	Background	Alternatives [Hen96 (related cases)]
	Applied in	SE phase	SE phase	Functional area [Pri85], Granularity [BR91]
	Applied on	SE object	Description	Element [Pri85 (objects), Veg02]
Context	Application domain	Industrial sector	Context	Application domain [BR91, Pri85 (setting)]
		Type of organization	Context	Size of organization [Bir00]
	Pre-requisites		Interface	Precondition* [MR96, BR91 (implicit in dependencies)]
		Qualification required	Interface	Knowledge required [Veg02, BR91 (implicit in dependencies)]
		Training required		
		Experience required	Interface	Experience required [Veg02, BR91 (implicit in dependencies)]
		Input	Interface	Input [HSY98, BR91, Veg02]
	Output	Output	Interface	Output [HSY98, BR91]
	Project	Size	Context	Size [Veg02]
		Kind of software	Context	System type, Software type [Pri85 (system type), Veg02]
		Type of project	Context	Solution domain [BR91, Bir00 (environment)]
	Environment	IT environment	Context	Tools environment[Veg02]
		Development environment	Context	Programming language, Software architecture [Veg02]
		Development process	Context	Development method [Veg02]

Table 5 (continued)

Proposed Model			Information Items from the Literature	
Category	Element	Attribute	Element	Attribute[1]
		Paradigm		
		Interdependency	Interface	Technology dependencies [SEI97, Veg02, BR91, MR96 (interdependencies)]
Impact			Results	Effectiveness* [Veg02]
			Results	Usage consideration* [SEI97]
			Results	Limitations* [SEI97 (costs and limitations)]
			Results	Problems* [MR96 (perceived weaknesses), Veg02]
			Results	Solution* [Hen96]
			Results	Benefits* [MR96 (perceived strengths), Veg02]
	Impact on the development (project) costs	Training		
		Introduction	Costs	Tool costs [Veg02]
		Application	Costs	Costs of application [Veg02]
		Maintenance		
		Total cost of ownership	Costs	Costs [SEI97 (costs and limitations)]
		Project		
	Return	Return on investment		
		Latency of ROI		
Impact (contd.)	Impact on the quality of the product	ISO 25010 SQuaRE	Results	Quality* [BR91]
		Product		
	Impact on the process outcome	Process outcome		
	Impact on SE object	SE object	Characterization	Aspect [Pri85 (medium), Veg02, Bir00 (product quality)]
	Impact on the development (project) schedule	Introduction		
		Application		
		Latency time		
		Project		
		Time to market		
	Impact on productivity	Productivity		
	Latency in SE Phase	In SE phase		
	Description	Description		
	Reference	Origin	Maturity	Maturity [SEI97]

For the sake of readability, we use abbreviations for the references as follows: [Bir00] = Birk 2000, [BR91] = Basili and Rombach 1991, [Hed96] = Henninger 1996, [HSY98] = Henderson-Sellers et al. 1998, [MR96] = Maiden and Rugg 1996, [Pri85] = Prieto-Díaz 1985, [SEI97] = SEI 1997, [Veg02] = Vegas 2002

In order to attract software managers, it has to be ensured that information that they consider to be important is included in the empirical study's reports. The critical issue is where to get this information if, for example, the study addresses other goals, like exploring or understanding cause-effect relationships. Nevertheless, the objective, the environment, and the results of the study have to be communicated clearly. If comparing certain technologies or evaluating a technology's effectiveness or efficiency, the researcher is expected to have gained insights that help to answer the crucial questions.

The model described in the sections above is syntactically split into technology, context, and impact. This does not infer that context and impact can be seen as being separated from the technology. In fact, the motivation is that the technology is the stable part, while context and impact might change, e.g., from study to study. The first and second columns of Tables 6, 7 and 8 are the elements, respectively attributes, of the model. The third column provides information in terms of questions and examples that should help authors to provide the information required by the information needs model.

Regarding reporting on the **technology** category, the information should enable the manager to relate the technology to common knowledge. Therefore, it is important to describe the technology investigated in the study and clarify how technology-related research fits into existing work.

As identified in our studies, managers need to get information regarding the technology in order to understand and judge the appropriateness of the technology and identify alternative approaches. Although the importance of technology-related information is stated by most published guidelines, there is no common consensus on where this section fits best; Jedlitschka et al. (2008) suggest presenting it as special section of the background chapter. Following the common model of managers' information needs, the description of each of the technologies used in the study should include information such as listed in Table 6.

In most cases, the technology and the alternatives to be described here will be the treatments in the study. For example, if one intends to compare two reading technologies, their descriptions would have to be provided with regard to the research objectives. The description of any control treatment should be sufficient for readers to determine whether the control is realistic. The detail of the required description depends on the availability of earlier publications. With regard to the content of the description, it is most important that all identifying characteristics are provided from a general perspective.

Software managers need context-related information to understand to which extent the research situation relates to their specific situation and under which circumstances the results were achieved. In particular, this comprises all environmental information that will allow a reader to compare his situation to the situation investigated during the course of the specific study as shown in Table 7. Regarding the description of the **context**, the CONSORT Statement (Moher et al. 2001; Altman et al. 2001) suggests that the setting and location of a study are described. Speaking formally, the context is used to evaluate external validity, meaning transferability of the results from one context to another, i.e., from the context in which the study was performed to the context of the manager reading the report. Thus, any contextual aspect that might have an impact on the results should be taken into account.

While describing the context of a study, the authors have to describe, e.g., the population and the sample. Each sample has its individual characteristics, like those mentioned in the prerequisites. For example, the required qualification for an experiment could be to understand the format of the requirements documents. The training given to the participants consists of two-person days of initial training. Experience is related to what the participants bring with them, e.g., CS master students in their fourth year of studies. If results from individual studies are synthesized, the resulting guideline shall use this information, too.

Table 6 Reporting technology

Element	Attributes	Key questions to answer
Name	Name	What is the name of the technology?
	Abbreviation	Is there a commonly used abbreviation available? If yes, what is it?
Type	Type	Is it a method (inspections), technology (refactoring), tool (eclipse), paradigm (OOSE), technology (.net)?
Description	Short description	Brief description capturing the main elements of the category technology. What is it good for and when will it be applied?
	Long description	The description or respective references should allow a person to understand a technology in detail.
	Belongs to family	A classification in the sense that someone can judge to which family of technologies it belongs.
	Complements	What are complementary technologies, i.e., which technologies fit best?
	Literature	After all, the application of the technology should be reproducible. This is important for software managers who would like to pilot the technology. For readers who do not have the necessary background, a more general reference, e.g., to a textbook, might be helpful. Documentation of the technology or a reference to a text book describing all the details of the technology or to other research published concerning the technologies.
		Appropriate citation is required to enable readers to access the information.
	Background	Where does the technology come from? Who developed it? Further related studies, i.e., empirical studies that have investigated the same or similar treatments, and, if appropriate, levels of relevance to practice; that is, with which results has the technology been applied in industry?
		If available, existing evidence, in the form of earlier studies and especially experiments should be described. The relation to other studies in the field will help to arrange this work in a larger context and supports reuse of this study for replication or systematic review, improving the value of the research and providing a sound basis for this work.
		If the reported study is a replication, the parental study and its findings must also be described. This will help the reader follow the comparison of the findings.
		If applicable, that is, if one of the treatments (technologies) has been applied to real software projects or under realistic circumstances, it is suggested to provide a short summary of the findings and related references.
	Alternatives	Are there any alternatives for the technology available? A description of alternative solutions, especially of those addressing the same problem or those that are comparable from a technology point of view.
		The relation to alternative approaches in the field will help to arrange this work in a larger context. Shaw (2003) recommends that the background should not only be a simple list of research (i.e., experiments) but an objective description of the main findings relevant to the work at hand. Authors are advised by several guidelines to report background, whether supportive or contradictory. In addition, other possible alternatives and superseding technologies could be mentioned.
Applied in	SE phase	In which phase(s) of the development was the technology applied?
Applied on	SE object	To which objects of the development process was the technology applied?

To support software managers in easily finding the most important results with regard to the **impact**, we emphasize providing a description of the impact on costs, schedule, and quality as well as a summary of the limitations in one place. During our discussions with researchers, we often heard that they do not have all this information.

Table 7 Reporting context

Element	Attributes	Key questions to answer
Application domain	Industrial sector	In which industrial sector has the technology been applied?
	Type of organization	What is the size of the organization? Managers want to have as much information as possible about the organization in order to judge whether the findings can be compared to their own situation. For large organizations, it might be interesting to get the overall size, but also (even more important) the size of the unit that applied the technology, e.g., the software development unit (50 employees) of a very large company (+10 k employees). Other characteristics of the organization that might have an impact on the results, e.g., degree of team distribution.
Pre-requisites	Qualification required	What type of experience is required from the people applying the technology?
		If experience is not appropriate, the respective costs have to be added or the technology needs to be rejected.
	Training required	What kind of training is required to be able to apply the technology?
	Experience required	What kinds of people have applied the technology?
	Input	What kind of documents have to be available before the technology can be applied? What is the required format? What is the content of the documents?
	Output	What kind of documents will be available after having applied the technology? What is the format? What is the content of the documents?
Project	Size	What was the size of the project in terms of person-months in which the technology was applied?
	Kind of software	What kind of software was developed? E.g., embedded software for a motor management component of a car or information broker for an accounting system.
	Type of project	In what kind of project was the technology used? E.g., development from the scratch or maintenance project.
Environment	IT environment	Is a specific environment required? Is a specific IT infrastructure required?
	Development environment	What kind of development environment is required, e.g., software architecture, or development language?
	Development process	What kind of development process is required?
	Paradigm	What development paradigm is required?
	Interdependency	Are there any interdependencies with other technologies?

However, we think that, for example, regarding costs, respectively effort, something could be stated, e.g., if the effort required to train the participants and the effort for applying the technology were given, software managers could at least estimate some important aspects.

The impact shall be described as part of the conclusions. According to the information needs model, this information should comprise the elements listed in Table 8.

When describing the results, it is important to distinguish between statistical significance and practical importance (Kitchenham et al. 2002) or meaningfulness (Harris 2002). Thus, it

Table 8 Reporting for impact

Element	Attributes	Key questions to answer
Impact on the development (project) costs	Training	To which extent will training increase/decrease the development costs (e.g., for an experiment this could be the hours of training the participants got)?
	Introduction	To which extent will the introduction increase/decrease the development costs (e.g., because of more expensive people to be used)?
	Application	To which extent will the application increase/decrease the development costs (e.g., because fewer people are used or because it is much more complicated)?
	Maintenance	To which extent will maintenance of the technology increase/decrease the costs?
	Total cost of ownership	This is to some extent a summary of the attributes mentioned above. It incorporates all aspects of technology ownership costs, such as license costs.
	Project	What is the impact on the costs of the current project?
Return	Return on investment	What is the return on investments?
	Latency of ROI	How long will it take until it pays off?
Impact on the quality of the product	ISO 25010 SQuaRE	Several quality attributes can be defined here (ISO 9126 and sub-sequent standards provide a starting point). The focus is on the quality of the final product.
	ISO 25010 SQuaRE	Several quality attributes can be defined here. The focus is on the quality of the products and documents in the process.
	Product	What does it mean for the end product?
Impact on the process outcome	Process outcome	Many techniques, especially those in the area of management, do not necessarily have an impact on the final product, but they might, for example, reduce the risk of project failure because they improve the accuracy of effort estimation.
		E.g., What exactly will be improved?
Impact on SE object	SE object	On which object of the SE process does the technology have an impact?
Impact on the development (project) schedule	Introduction	How long will it take to introduce the technology such that it can be used?
	Application	Will it take more/less time (time-span) than before?
	Latency time	How long will it take until the technology is applied efficiently?
	Project	What is the impact on the current project schedule?
	Time to market	What is the impact on time to market?
Impact on productivity	Productivity	Productivity can be either increased or decreased at the beginning because of the experience required.
Latency in SE Phase	In SE phase	What is the latency and in which phase of the SE process does it occur?
Description	Description	Textual description of the impact.
Reference	Origin	Who observed the impact?

might happen that the analysis reveals statistically significant results, e.g., regarding a certain level of effectiveness. Nevertheless, the size of the effect might indicate that from a business perspective, the results cannot be of any importance.

Besides the description of the impact, we ask for a discussion of the approach's level of maturity, when the investments will pay back, and consequences arising from the implementation. Although in most cases artificial, we assume a rough estimate is better than no information.

If applicable, limitations of the approach with regard to its practical implementation shall be described, i.e., circumstances under which the approach presumably will not yield the expected benefits or shall not be employed. Furthermore, any risks or side-effects associated with the implementation or application of the approach shall be reported, just like the package inserts accompanying medications.

9 Validation

We performed an experiment to analyze the information needs model with respect to its effectiveness (Jedlitschka 2010). We evaluated the extent to which the information needs model supports the ability to judge a technology's appropriateness and impact from the point of view of software managers in the context of SE technology selection in their organization within a predefined scenario.

The main **hypothesis** for this experiment was that the value of a technology can be judged better by software managers when relevant information appears in an experiment report. This means that software managers who read experiment reports containing information requested by the information needs model perceive that

(H1) they are able to judge the appropriateness of the technology better and
(H2) they are able to judge better whether the technology has an impact than those who read experiment reports following regular reporting practices.

We further hypothesize that

(H3) this will yield a higher level of consideration of the technology as a candidate for the technology selection process.

For measuring users' perception, we follow the general concept of Venkatesh et al. (2003).

The **population** for this study was software managers. The experiment design followed the traditional two independent samples design (Wohlin et al. 2012). According to our setting with one factor and two treatments and based on the hypotheses, we considered the one-tailed *t*-test as appropriate; i.e., calculating whether there is a significant difference between two independent means given for the ratings to the statements. The final **sample** consisted of 22 software managers from 19 different organizations representing the following domains: automotive, avionics, information systems, finance, and telecommunication. As a conclusion from the characterization, we considered the participants to be appropriate for the experiment. In fact, the participants are involved in the technology selection process and have the roles that are relevant in the process as shown in our studies. The participants' business experience and technology selection experience were also considered to be reasonable. The participants were **assigned randomly** to either the model version (E) or the author version (C) of the experiment report. Half of the participants got the original experiment report, whereas the other half got the model-based version.

The **material** for this study included the description of an experiment (Juristo and Vegas 2003) in two versions, the instructions, and the questionnaire. In agreement with the authors of the original experiment report, for the author version, we had to remove a second experiment that was also reported in the same publication in order not to confuse the evaluation. For the model version, we reengineered the author version with support from the authors of the original

experiment report. The original version lacked information required by the model, but from the insights they gained while running and analyzing the experiment, the authors were able to provide most of the required information. However, obtaining information related to the technology's impact on costs was most crucial and required some discussions.

The author version has 10 pages, whereas the model version has 10.5 pages. The instructions, including the scenario, were adopted from a baseline experiment (Jedlitschka 2009). The participants were asked to read the experiment reports and answer a questionnaire, which asked them about their ability to judge a technology's appropriateness and impact, as well as its intended use. The participants were further asked to write down the time when they started the experiment and the time when they completed the main part of the questionnaire. In addition, the participants were asked to provide information on whether they experienced any significant interruptions while working on the study.

We assume that the experience of the participants, especially with regard to reading reports on controlled experiments or research reports, might have an impact on the results. In addition, the time pressure industry people are subject to might influence the accuracy of their reading. Both aspects were controlled by asking corresponding questions in the post-study questionnaire (the experimental package is available at http://www.jedlitschkas.de/downloads/im_exp.pdf). Because the task is comparable to reading an article and because we expected that the participants had some experience with that task, no training was provided.

The participants got the material for the study by email. The next day, the researcher contacted the participants to ask them whether they had received the mail. The participants were asked to return the completed questionnaire within 2 weeks. In case of delays, we personally approached the participants and asked them to complete the study.

Before we started with the analysis, we performed a quality check on the data. We performed an outlier analysis by adding the ratings given to the single questions. Then we tested the overall score with regard to significant differences to the other participants in the same group. The outliers were removed from the sample. This means that the following analyses were performed based on a set of 20 questionnaires (10 for the experimental group, 10 for the control group). We carefully analyzed the confounding variables and did not find any significant influence (the detailed analysis can be obtained from (Jedlitschka 2009)); therefore, we conclude that the results (cf. Table 9) come from the treatment, which are the two versions of the experiment report. In the following, we discuss the implications. We assumed an alpha-level of p=0.05 and the power (1-β) was to be at least 0.8.

9.1 H1: Ability to Judge a Technology's Appropriateness

Based on the results for hypothesis *H1*, we can conclude that experiment reports following the information needs model increase the software managers' perception of being able to

Table 9 Hypotheses tests: *t*-tests for equality of means on a confidence interval of 95 % (level p=0.05)

Hypothesis	$t_{crit0..95}$ (one-tailed)	t	df	Sig. (one-tailed) p=0.05	Mean Difference	Std. Error Difference	95 % Confidence Interval of the Difference		Effect size d	Power (1-β)=0.8
							Lower	Upper		
H1	1.73	3.03	18	***0.01***	1.10	0.36	0.34	1.86	1.36	***0.9***
H2	1.73	2.01	18	***0.03***	1.00	0.50	−0.04	2.04	0.90	0.61
H3	1.73	4.02	18	***0.00***	1.45	0.36	0.69	2.21	1.80	***0.99***

judge a technology's appropriateness. This means that the information needs model obtained efficiently supports the selection of alternative solutions. The information needs model, represented by the reporting guidelines that were used to "re-engineer" the paper for the experimental group, yielded the expected difference.

9.2 H2: Ability to Judge Whether the Technology has an Impact

Based on the results for hypothesis *H2*, we cannot conclude that the experiment reports following the information needs model increase software managers' perception of being able to judge a technology's impact on the development process. However, this conclusion arises from the low power of the result.

One participant from the experimental group stated that it would be preferable to have results from "real-world studies". This leads us to the assumption that the impact as described in the model version is still considered too weak because it stems from a toy example. Another participant stated that the experience has shown slightly other impacts, especially regarding the impact on costs. However, several participants from the experimental group stated that the conclusions section in the model version provided them with the information for their judgment. This section contained explicit information regarding the technology's impact.

9.3 H3: Consider the Technology as a Candidate

Based on the results for hypothesis *H3*, we conclude that the experiment reports following the information needs model yielded higher consideration of the technology as a candidate for the selection process.

In summary, the significant difference between the experimental group and the control group regarding the participants' perceived ability to judge a technology's appropriateness supports our main hypothesis. The information related to the impact also yielded the expected significant difference, but the power is too low to accept the hypothesis. However, the trend is clearly in the expected direction.

The general conclusion from the evaluation is that the research hypothesis can be accepted. That is, reports on experiments that satisfy software managers' information needs are more effective with regard to judging a technology's appropriateness and impact (relevance) than those not following this approach. And if a software manager is able to judge the relevance, there is greater probability for the technology to be considered as a candidate in the selection process. This means that decisions would be made considering a more complete set of relevant alternatives, which will hopefully yield better decisions.

Regarding the cost required for applying the guidelines, it was found that there might be an increase in the number of pages. From the perspective of a software manager, the costs are related to the time required for reading the experiment report and judging whether the technology is relevant or not. Based on the results from the baseline experiment, we conclude that with growing experience and increasing understanding of what is reported where, the required time will decrease.

Regarding the time required for applying the guidelines, information from the authors of the two experiment reports implies that they needed extra time to collect and provide the information required by the model. For software managers, we have discussed that time is a factor that restricts the identification of alternatives, i.e., decision makers will obviously stop searching before they have found an appropriate set of alternatives. We conclude from our results that if the information is provided following the guidelines, or is available in a repository, the time required for the search will not increase.

Regarding the quality of the publication, the experiment showed that there is an implicit increase in quality because the readers were better able to judge the appropriateness of a technology. We did not investigate the impact on the quality of the selection. However, using the knowledge from decision theory that a more appropriate set of alternatives yields better decisions, we conclude that the proposed model will have a positive impact on the quality of the decision.

The implications of our results are summarized in Table 10.

Concluding, we found that software managers show a significantly increased ability to judge a technology's appropriateness if the paper included information requested by the proposed model. The model yielded higher ability to judge a technology's impact and significantly higher consideration of the technology in the selection process. Participants who read the version that was re-engineered according to the information needs model believed that it would be easier to convince their management to introduce the technology.

Thus, participants who read the re-engineered version showed a significantly increased ability to judge a technology's appropriateness and increased willingness to consider the technology in the selection process.

10 Threats to Validity

The proposed model, although based on models available in the literature and qualitative as well as quantitative data, is a subjective construction. This means that another researcher might have arrived at different categories and at a different information needs model. Citing Guba and Lincoln (1994), Dybå wrote that the question is not "…whether the model is more or less true in any absolute sense, but simply if it is more or less informed and/or sophisticated" (Dybå 2001).

In the following, we discuss **threats** (cf. Wohlin et al. 2012) that might have an impact on the validity of the results for the entire set of empirical studies presented here. This includes (1) threats to construct validity, (2) threats to internal validity, and (3) threats to external validity.

Threats to conclusion validity would address only the experiment. Potential threats were handled by respective measures, such as a power analysis and explanations for the applied tests.

Table 10 Summary of implications

Hypothesis	Result	Implication
H1 (appropriateness)	accepted	Experiment reports following the information needs model increases software managers' perception regarding their ability to judge a technology's appropriateness.
H2 (impact)	H_{02} rejected H_{12} not accepted	Experiment reports following the information needs model do not increase software managers' perception regarding their ability to judge a technology's impact. However, there is some indication that the model supports their judgment, although the power of the result is too low.
H3 (consider)	accepted	Experiment reports following the information needs model result in higher probability of the technology being considered an alternative.

The operationalization of **constructs** has been carefully designed and tested where appropriate, e.g., for the items of the measurement instrument of the experiment a reliability analysis was conducted.

For the experiment, we used only one report for each group. We decided to do so because we were using software managers on the one hand and were interested in a high statistical power on the other hand (number of participants for each report).

The participants in our studies did not know about the design of the study or the details of the treatment. However, for the experiment, the questionnaire might have allowed them to identify more information from the paper than they would have identified without it. But this would not change the general result of the experiment because the questionnaire was the same for both groups.

Concerning **internal validity**, the quality of the instruments might have had an impact on the effects. The questionnaires were designed carefully, and based on questionnaires used in earlier studies. In addition, we piloted the questionnaires with several colleagues. Nevertheless, the type of questions might have influenced the way the respondents answered our questions. Whereas in Study 1, the participants were asked which information they would ask from a technology provider, a different format of the questions, such as "Which information would you need to interpret the results of this study in the context of your organization"? might have yielded different answers. In Study 4, we used open text fields in addition to the given kinds of information, and asked for information that was supporting a decision as well as for information that was missing. Therefore, we argue that our multi-method approach helped us to overcome this potential threat.

We might have unintentionally introduced a restriction with regard to the kinds of information our participants thought of when we explained to them that we were considering information from controlled experiments. Examples are the impact on intellectual property rights (IPR), improved capabilities, and synergy effects with other organizations. These have to be incorporated into the individual decision-making process, especially in the context of a risk analysis. From that perspective, it is interesting that nobody mentioned these aspects. However, we think that our participants did not expect to get an answer from research here.

Possible prejudice of the participants against controlled experiments was not controlled. However, one participant in the experiment stated that according to his experience, research papers will not get the attention of senior managers; in addition, the findings from Study 4 showed some indication in that direction.

To ensure that the procedure was the same for all participants, the interviews (Study 2) were conducted by the same interviewer. For the online surveys (Study 4 and 5) and the experiment, task performance was not controlled. However, for the surveys, the time needed for filling in the questionnaires was measured and analyzed, and for the experiment, the participants were asked to write down the time they needed, which was also analyzed with regard to potential impacts on the results. Furthermore, as far as the process, e.g., for the experiment, was prescribed, the participants were asked to rate their process conformance. In addition, professionals are heterogeneous by nature, e.g., with regard to experience, background, role. A careful analysis of confounding factors related to experience showed that the results are due to the treatment and not due to individual differences.

All studies were performed with volunteers, who might have been more motivated than those who did not partake for various reasons. For the interviews and the experiment, we used convenient sampling. There was no drop-out from the experiment. For the surveys, the drop-out rate was quite high, and a non-respondent analysis was not conducted. We have to accept that and for this reason the results of the surveys (Study 4 and Study 5) might not be representative.

With regard to **external validity**, the 175 participants in our studies had different roles, came from different industrial organizations in Germany (except for Study 1), and from different domains.

Given potential limits with regard to the representativeness of the samples, our research might have yielded models that do not reflect the information required by certain kinds of managers or do not reflect all domains. In addition, roles are not defined coherently across organizations; consequently, no comparison is really possible. However, the involvement of the participants in the decision-making process (as controlled in Study 3, Study 4, Study 5, and the experiment) led us to the conclusion that for systematic technology selection, the results might be generalizable. Nevertheless, we claim that the variety of domains and organizations from which the participants came support the validity of our results. Regarding this aspect, we also claim that if the results of the different managers converge, this gives an indication that the needs are similar for managers in different organizations and domains.

The experiment report was chosen for reasons of convenience because the authors of the original report supported the writing of the model version; thus, it might not be representative of average reports. However, the original report was published as a chapter in a peer-reviewed book (Juristo and Vegas 2003), which is taken as an indication of its general quality.

The studies were conducted using exemple technologies. It is not clear whether the results would be similar for other kinds of technologies. We did not ask for specific technology-related information; therefore, we assume that the results can be transferred.

11 Conclusions

One reason for software managers making limited use of experiments results when selecting technologies might be that experiment reports lack information that is relevant for decision-making. We propose an approach for reporting results from experiments to software managers. The focus is restricted to experiments because experiments represent a very structured instrument for technology evaluation.

We elicited software and senior managers' information needs in a series of empirical studies with a total of 175 participants from industry. We developed a common model of managers' information needs that might be used by researchers to learn what is important to report when they address software managers. There are three categories of information a software manager needs in an experiment report: technology, context, and impact. Regarding the technology, the information provided should enable the software manager to relate the technology with common knowledge including development phase and products on which the technology is to be used. Regarding context, software managers need information to understand to which extent the experimental setting relates to their specific situation. Finally the impact on cost, schedule and quality is crucial. For information on the impact, a rough estimate or a subjective opinion based on researchers' findings might be sufficient initially.

We evaluated the effectiveness of the proposed information needs model in an experiment with 22 software managers. The results showed that reporting experiments with the information of this model might yield a change in the acceptance of experiment reports as a valuable source of information in a technology selection process. If SE researchers want their research findings to have an impact in industrial practice, the information required by software managers need to be delivered and packaged in the way they expect. If results from experiments are reported to software managers by following our model, the chances of having such an impact, might increase.

Current research towards synthesizing evidence from primary studies aimed at providing technology guidelines for software managers could benefit also from the proposed model. In a next step it needs to be investigated to which extent our model could also be used while planning syntheses and reporting the results from syntheses. In particular, the model would then be used to describe results from a synthesis, the category technology would contain the general attributes of the technology, the category context would contain a summary of the contexts in which the technology was used, and the category impact would contain the synthesis of the results.

References

Altman DG, Schulz KF, Moher D, Egger M, Davidoff F, Elbourne D, Gøtzsche PC, Lang T, for the CONSORT Group (2001) The Revised CONSORT Statement for Reporting Randomized Trials Explanation and Elaboration. Ann Intern Med 134(8):663–694

Association AP (2001) Publication Manual of the American Psychological Association, 5th edn. American Psychological Association, Washington, DC

Ayari B, Rombach D (Supervisor), Jedlitschka A (Supervisor), Weibelzahl S (Supervisor) (2004) Anforderungsanalyse, Entwurf und Implementierung eines web-basierten Entscheidungsunterstützungssystems für Software Engineering Improvement Management, Diploma Thesis, Dept. of Computer Science, University of Kaiserslautern, Germany

Basili VR, Rombach D (1991) Support for comprehensive reuse. Softw Eng J, IEEE Br Comput Soc 6(5):303–316

Basili VR, Caldiera G, Rombach HD (2001) Experience Factory. In: Marciniak JJ (ed.), Encyclopedia of Software Engineering, Vol.1, John Wiley & Sons, 2001, pp. 511–519.

Birk A (2000) A Knowledge Management Infrastructure for Systematic Improvement in Software Engineering, Dissertation, Dept. of Computer Science, University of Kaiserslautern, Germany, Stuttgart Fraunhofer IRB Verlag

Chrissis MB, Konrad M, Shrum S (2011) CMMI for Development Guidelines for Process Integration and Product Improvement, 3 revisedth edn. Addison-Wesley Longman, Amsterdam

Ciolkowski M, Laitenberger O, Biffl S (2003) Software reviews the state of the practice. IEEE Softw 20(6):46–51

Dybå T (2001) Enabling software process improvement-an investigation of the importance of organizational issues, Dissertation, NTNU, Trondheim, Norway

Glass RL (2004) Matching methodology to problem domain. Column Pract Program Commun ACM 47(5):19–21

Glass RL (2006) The Academe/Practice Communication Chasm—Position Paper. Dagstuhl Seminar on Empirical SE 27.06.-30.06.06 (06262), Participant Materials. http://www.dagstuhl.de/Materials/Files/06/06262/06262.GlassRobert.ExtAbstract!.pdf Accessed on 26 June 2013

Guba EG, Lincoln YS (1994) Competing paradigms in qualitative research. In: Denzin NK, Lincoln YS (eds) Handbook of qualitative research. Sage, London, pp 105–117

Harris P (2002) Designing and Reporting Experiments in Psychology, 2nd edn. Open University Press, Berkshire

Henderson-Sellers B, Simons A, Younessi H (1998) The OPEN Toolbox of Techniques. Harlow Addison-Wesley (The OPEN Series)

Henninger S (1996) Accelerating the Successful Reuse of Problem Solving Knowledge Through the Domain Lifecycle. In Proc. of the 4th Intern. Conf. on Software Reuse (ICSR '96). IEEE Computer Society, Washington, DC, USA, pp. 124–133

Hinchey MG, Pressburger T, Markosian L, Feather MS (2006) The NASA software research infusion initiative successful technology transfer for software assurance. In: Proc. of the 2006 intern. workshop on Software technology transfer in software engineering (TT '06). ACM, New York, pp 43–48

Ivarsson M, Gorschek T (2011) A method for evaluating rigor and industrial relevance of technology evaluations. Empir Softw Eng 16(3):365–395

Jedlitschka A (2007) How to improve the use of controlled experiments as a means for early technology transfer? Position Paper. In: Basili VR, Rombach D, Schneider K, Kitchenham B, Pfahl D, Selby RW (eds.) Empirical Software Engineering Issues Critical Assessment and Future Directions, International Workshop Dagstuhl Castle, Germany, January 2007, Springer Verlag, LNCS 4336, p. 130

Jedlitschka A (2009) An empirical model of software managers information needs for software engineering technology selection, Dissertation, Dept. of Computer Science, University of Kaiserslautern, Germany, Fraunhofer IRB Verlagm Stuttgart, Germany

Jedlitschka A (2010) Evaluating a model of software managers' information needs an experiment. In Proc. of the 2010 ACM-IEEE Intern. Symposium on Empirical Software Engineering and Measurement (ESEM '10). ACM, New York, NY, USA, Article No. 19, 10 pages

Jedlitschka A, Briand LC (2007) The role of controlled experiments Working group results. In: Basili VR, Rombach D, Schneider K, Kitchenham B, Pfahl D, Selby RW (eds.) Empirical Software Engineering Issues Critical Assessment and Future Directions, International Workshop Dagstuhl Castle, Germany, January 2007, Springer Verlag, LNCS 4336, pp. 58–62

Jedlitschka A, Ciolkowski M (2004) Towards Evidence in Software Engineering. In Proc. Intern. Symposium on Empirical Software Engineering 2004 (ISESE2004), Redondo Beach, California, USA, August 2004, 2004, pp. 261–270

Jedlitschka A, Pfahl D (2004) Requirements of a Tool supporting decision making for SE Technology Selection. In Proc. Of 16th Intern. Conf. on Software Engineering and Knowledge Engineering (SEKE2004), Banff, Canada, 2004, pp. 513–516

Jedlitschka A, Ciolkowski M, Denger C, Freimut B, Schlichting A (2007) Relevant Information Sources for Successful Technology Transfer A Survey Using Inspections as an Example, In: Proc. Intern. Symposium on Empirical SE and Measurement 2007 (ESEM2007), Madrid, Spain, September 2007, 2007, pp. 31–40

Jedlitschka A, Hamann D, Göhlert T, Schröder A (2005) Adapting PROFES for Use in an Agile Process An Industry Experience Report. In: Bomarius F and Komi-Sirviö S (Eds.) 6th Intern. Conf. on Product Focused Software Process Improvement (Profes2005) . Springer-Verlag, Berlin 2005, pp. 502–516

Jedlitschka A, Ciolkowski M, Pfahl D (2008) Reporting controlled experiments in Software Engineering. In: Shull F, Singer J, Sjøberg D. (eds.) Guide to Advanced Empirical Software Engineering, Springer, 2008 pp. 201–228

Juristo N, Moreno A (2001) Basics of Software Engineering Experimentation. Kluwer Academic Publishers

Juristo N, Vegas S (2003) Functional Testing, Structural Testing and Code Reading What Fault Type Do They Each Detect? In: Conradi R, Wang AI (eds) Empirical Methods and Studies in SE—Experiences from ESERNET. Springer-Verlag, LNCS, Berlin, pp 208–232

Kitchenham BA (2004) Procedures for Performing Systematic Reviews. Keele University Joint Technical Report TR/SE-0401, ISSN 1353–7776 and National ICT Australia Ltd. NICTA Technical Report 0400011T.1

Kitchenham BA, Pfleeger SL, Pickard LM, Jones PW, Hoaglin DC, El Emam K, Rosenberg J (2002) Preliminary guidelines for empirical research in Software Engineering. IEEE Trans Softw Eng 28(8):721–734

Kvale S (1996) InterViews. An Introduction to Qualitative Research Interviewing. Sage Publications, Thousand Oaks, p 326 S

Lott CM, Rombach HD (1996) Repeatable Software Engineering experiments for comparing defect-detection techniques. Empir Softw Eng J 1(3):241–277

Maiden NAM, Rugg G (1996) ACRE selecting methods for requirements acquisition. Softw Eng 11(3):183–192

Marshall G (1998) "Secondary analysis". A Dictionary of Sociology. 1998. Encyclopedia.com http://www.encyclopedia.com/doc/1O88-secondaryanalysis.html Accessed 26 June 2013

Moher D, Schulz KF, Altman D, for the CONSORT Group (2001) The CONSORT statement revised recommendations for improving the quality of reports of parallel-group randomized trials. JAMA 285(15):1987–1991

Pfleeger SL (1999) Understanding and improving technology transfer in SE. J Syst Softw 47(1999):111–124

Pfleeger SL, Menezes W (2000) Marketing technology to software practitioners. IEEE Softw 17(1):27–33

Pickard LM, Kitchenham BA, Jones PW (1998) Combining empirical results in software engineering. Information and Software Technology, 40(14) 1998, pp. 811–821

Prieto-Díaz R (1985) A Software Classification Scheme (Reusability, Libraries, Development). Doctoral Thesis. Department of Information and Computer Science, University of California, Irvine

Rainer A, Hall T, Baddoo N (2003) Persuading Developers to 'Buy into' Software Process Improvement: Local Opinion and Empirical Evidence. In Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE '03). IEEE Computer Society, Washington, DC, USA, pp. 326–335

Reifer DJ (2003) Is the SE state of the practice getting closer to the state of the art? IEEE Softw 20(6):78–83

Rogers EM (2003) Diffusion of Innovations, 5 (Paperback)th edn. The Free Press, New York

Rombach D, Ciolkowski M, Jeffery R, Laitenberger O, McGarry F, Shull F (2008) Impact of research on practice in the field of inspections, reviews and walkthroughs learning from successful industrial uses. ACM SIGSOFT Softw Eng Notes 33(6):26–35

Runeson P, Höst M (2009) Guidelines for conducting and reporting case study research in SE. Empir Softw Eng 14(2):131–164

Runeson P, Thelin T (2003) Prospects and Limitations for Cross-Study Analyses—A Study on an Experiment Series. In: Jedlitschka A and Ciolkowski M (eds) The Future of Empirical Studies in Software Engineering, Proc. of 2nd Int. Workshop on Empirical Software Engineering, WSESE 2003, Roman Castles, Italy, Sept. 2003, Fraunhofer IRB Verlag, 2004, pp. 141–150

SEI (1997) C4 Technology Reference Guide—A Prototype. Technical Report CMU/SEI-97-HB-001. SE Institute, Pittsburgh
Shaw M (2003) Writing Good SE Research Papers—Mini-tutorial. In: Proc. of the 25th Intern. Conf. on SE (ICSE'03). IEEE Computer Society, Portland, pp 726–736
Shiffman RN, Shekelle P, Overhage JM, Slutsky J, Grimshaw J, Deshpande AM (2003) Standardized reporting of clinical practice guidelines a proposal from the conference on guideline standardization. Ann Intern Med 139(6):493–498
Shull F, Carver J, Travassos GH, Maldonado JC, Conradi R, Basili VR (2003) Replicated Studies Building a Body of Knowledge about Software Reading Techniques. In: Juristo N, Moreno A (eds) Lecture Notes on Empirical Softw. Engg. USA World Scientific Publishing, River Edge, pp 39–84
Singer J (1999) Association (APA) Style Guidelines to Report Experimental Results. In: Proc. of Workshop on Empirical Studies in Software Maintenance. Oxford, England. September 1999. pp. 71–75 http://dec.bmth.ac.uk/ESERG/WESS99/singer.ps Accessed on 25 June 2013
Sjøberg D, Hannay J, Hansen O, By Kampenes V, Karahasanovic A, Liborg N-K, Rekdal A (2005) A survey of controlled experiments in software engineering. Trans Softw Eng 31(9):733–753
Strauss A, Corbin J (1998) Basics of Qualitative Research. Techniques and Procedures for Developing Grounded Theory, 2nd edn. Sage, Thousand Oaks
Turner R (2004) Why we need empirical information on best practices. CROSSTALK—The Journal of Defense Software Engineering. April 2004, pp. 9–11
Vegas S (2002) Characterisation Schema for Selecting Testing Techniques. Dissertation Universidad Politéchnica de Madrid, Facultad de Informática, Departamento de Lenguajes y Sistemas Informáticos e Ingeniería del Software, Madrid, Spain
Vegas S, Juristo N, Basili VR (2006) Packaging experiences for improving testing technique selection. J Syst Softw 79(11):1606–1618
Venkatesh V, Morris MG, Davis GB, Davis FD (2003) User acceptance of information technology—towards a unified view. Vol. 27, No.3 (Sep., 2003), pp. 425–478
Wohlin C, Runeson P, Höst M, Ohlsson MC, Regnell B, Wesslén A (2000) Experimentation in Software Engineering—An Introduction. Kluwer Academic Publishers
Wohlin C, Petersson H, Aurum A (2003) Combining Data from Reading Experiments in Software Inspections. In: Juristo N, Moreno A (eds) Lecture Notes on Empirical Softw. Engg. USA World Scientific Publishing, River Edge, pp 85–132
Wohlin C, Runeson P, Höst M, Ohlsson MC, Regnell B (2012) Experimentation in Software Engineering. Springer

Andreas Jedlitschka received his PhD degree (Dr.-Ing.) from the Computer Science Department of the University of Kaiserslautern in 2009. Currently Andreas heads the department "Measurement, Prediction, and Empiricism" at the Fraunhofer Institute for Experimental Software Engineering. Before joining Fraunhofer IESE in 2000, he worked for seven years as an IT-consultant. His research interest is empirically-based decision support where he has published several papers. In 2008, Andreas published guidelines for reporting controlled experiments in software engineering (SE), thus providing one starting point for the adoption of the evidence-based SE paradigm.

Natalia Juristo received the PhD degree from the Technical University of Madrid in 1991. She is currently a professor of software engineering at Universidad Politecnica de Madrid. Natalia was the Director of the UPM MSc in Software Engineering from 1992 to 2002 and the coordinator of the Erasmus Mundus European Master on SE (with the participation of the University of Bolzano, the University of Kaiserslautern and the University of Blekinge) from 2006 to 2012. Her main research interests are experimental software engineering, requirements and testing. Natalia is the coauthor of the book Basics of Software Engineering Experimentation (Kluwer, 2011). She is a member of the editorial board of Empirical SE Journal. She has recently been awarded with a FiDiPro (Finland Distinguish Professor) research grant. She began her career as a developer in the European Space Agency (Rome) and the European Center for Nuclear Research (Geneva). She was a resident affiliate at the Software Engineering Institute in Pittsburgh in 1992.

Dieter Rombach is a professor and chair of the Software Engineering Department at the University of Kaiserslautern and executive director of the Fraunhofer Institute for Experimental Software Engineering. His research interests include software methodologies, modeling and measurement of the software process and resulting products, software reuse, and distributed systems. Rombach received a PhD in computer science from the University of Kaiserslautern in 1984. In 2009, the University of Oulu, Finland bestowed upon him an honorary doctorate degree in recognition of his lifetime achievements as a software engineer. He is a member of the Gesellschaft für Informatik (GI) and is a Fellow of the ACM (since 2010) and a Fellow of the IEEE Computer Society (since 2003).