Fraunhofer Institut
Experimentelles
Software Engineering

Requirements for a Tool in Support of SE Technology Selection

Authors:
Andreas Jedlitschka
Dietmar Pfahl

IESE-Report No. 017.04/E
Version 1.0
February 2004

A publication by Fraunhofer IESE

Fraunhofer IESE is an institute of the Fraunhofer Gesellschaft.
The institute transfers innovative software development techniques, methods and tools into industrial practice, assists companies in building software competencies customized to their needs, and helps them to establish a competitive market position.

Fraunhofer IESE is directed by
Prof. Dr. Dieter Rombach
Sauerwiesen 6
67661 Kaiserslautern

Abstract

Decision support in software engineering is an emerging field. The need to select the best method, technique or tool in a given business context is becoming more and more important in an increasingly competitive world where time-to-market, budget constraints, and the achievement of functionality and quality goals are becoming crucial for achieving a company's business goals. In today's software development organizations, technologies are employed that frequently lack sufficient evidence regarding their suitability, their limits, qualities, costs, and inherent risks. This paper presents ongoing research towards the development of a decision support system that aims at improving software engineering technology selection by software managers. To develop such a system, a multiple-step requirements analysis, consisting of a literature survey, a pilot study amongst research managers, and the analysis of additional use cases, was performed. The focus of this paper is on presenting and discussing the results of this three-step requirements analysis process.

Table of Contents

1	**Introduction**	**1**
2	**Generic Requirements for SE-DSS**	**3**
3	**Requirements for a Comprehensive SE-DSS**	**4**
3.1	Requirements Elicited from Experts	4
3.2	Additional Requirements	8
4	**Structured List of Requirements**	**10**
5	**Summary and Future Work**	**13**
References		**14**

1 Introduction

Software engineering decision support (SE-DS) is an emerging field [1, 2]. One of the major goals of SE-DS is to support software managers in selecting suitable SE technologies. Suitability implies the existence of a defined level of evidence about the effectiveness of a specific SE technology in a given context. Similar to work done in the area of empirical software engineering (ESE), SE-DS implies data collection, analysis, and modeling. In addition, SE-DS involves model application, possibly supported by software infrastructure.

In the past, research in both ESE and SE-DS has focused on evaluating technologies in isolation (e.g., empirical research on inspection techniques [3]). This has produced many results that help software managers to better understand the effectiveness of SE technologies in a stable – but undefined – context. Moreover, much effort has been invested into analyzing results from different empirical studies [4] focusing on the same type of techniques, e.g., [3, 5], hence increasing the level of evidence about the *local* effectiveness of SE methods, techniques, and tools. This evidence can be used for *local* SE-DS.

Unfortunately, by relying on nothing but a body of knowledge with – mostly isolated – pieces of local evidence, conclusions about the effectiveness of combinations of SE technologies can hardly be derived without further methodological and tool support. Why is this a problem? It is a problem because managers are often not interested in the number of defects that a particular QA technique might potentially find; they are interested in the impact of a particular QA technique on the overall project goals (which include many issues, e.g., functionality, quality, time-to-market, budget constraints, etc.). One can even go one step further and say that managers are only interested in the impact of a particular SE technology on the overall business value. In order to address these broader questions, methods and systems that support comprehensive (instead of local) SE-DS are needed. To give an example, comprehensive SE-DS answers questions of the following type: Which *combination* of inspection techniques (incl. requirements, architecture, design, code inspections) and test techniques (incl. unit, integration, system, acceptance test) shall be selected in a given business context? In other words, it is not sufficient to compare the effectiveness of different types of code inspections, but the effectiveness of combinations of a specific type of code inspection with other SE techniques along the whole software development life cycle [6] must also be taken into account.

One step towards comprehensive SE-DS emerged from the process simulation research community. By emulating the project context with the help of generic process simulators that are instantiated by adding SE technology specific information as needed, many questions similar to the example question above can be answered [7, 8, 9]. Specifically, a simulation-based approach would facilitate the explicit modeling of the various interactions between software development and quality assurance technologies. Moreover, the software development contexts could simply be modeled via environment parameters. There is, however, a practical limitation to this approach. Even though much progress has been made in the last couple of years, it is still very difficult and costly to come up with company-specific generic process simulators that are sufficiently flexible and valid at the same time.

Therefore, in this paper, we rely on a new approach to comprehensive SE-DS [6]. This approach does not require process simulators but enhances the power of existing software engineering decision support systems (SE-DSS) that focus on providing local evidence. Examples of such systems include the ESERNET[1] web-based repository [10]. The trade-off as compared to using process simulators is that technology interaction and development context can only be modeled in a black box like manner via pre/post-conditions.

The focus of this paper is on requirements elicitation for comprehensive SE-DSS serving software managers in making SE technology selection decisions aligned to project goals and/or business objectives.

The structure of the paper is as follows. Section 2 reviews the sets of generic requirements to SE-DSS proposed by others. Section 3 presents our requirements elicitation process for a comprehensive SE-DSS. Section 4 maps our findings to a standard architecture and a generic requirements classification framework. The paper concludes with a brief discussion of the results and an outlook to future work.

---

[1] ESERNET (Experimental Software Engineering Network) was a thematic network funded by the European Commission from 2001 – 2003. Information is available from [13] and http://www.esernet.org

2 Generic Requirements for SE-DSS

Several authors have proposed sets of requirements for SE-DSS on various levels of abstraction. Ruhe [1], for example, suggests nine categories of SE-DSS requirements. The focus of his analysis is on requirements "that combine the intellectual resources of individuals and organizations with the capabilities of the computer to improve effectiveness, efficiency and transparency of decision-making". In addition, Ruhe defines associated functional components and proposes a generic SE-DSS architecture.

Li et al. [11] suggest a generic set of requirements, which is based on Ruhe's proposal, but primarily focuses on web-based aspects, and particularly on user interface friendliness.

Also related to our proposal of building a comprehensive SE-DSS is the work done by Biffl et al. [12]. They describe functional and non-functional requirements of a knowledge management system thatbuilds upon a framework to support software inspection planning.

All these proposals from literature were a good starting point for our research, but in order to come up with a set of empirically-based requirements specifically aiming at a comprehensive SE-DSS, more work had to be done.

3 Requirements for a Comprehensive SE-DSS

An alternative to Ruhe's architecture, which mainly gives a hierarchy of components related to the requirements (process view), is the standard three-tier architecture, which consists of (1) the user interface layer, (2) the process management layer, and (3) the data management layer.

In the following sub-sections, we first present the results of expert interviews that were conducted in order to elicit specific requirements for a comprehensive SE-DSS. Then, we list additional requirements derived from lessons learned of a European research project that aimed at establishing an international repository of experience on the effectiveness and efficiency of SE technology [13][14].

3.1 Requirements Elicited from Experts

The purpose of conducting expert interviews was to elicit a relevant and reliable set of requirements for a comprehensive SE-DSS. In order to be relevant, interviewees had to be sufficiently mature with regards to software management experience. In order to be reliable, a sufficient number of subjects had to be interviewed. Being a research institute that is largely involved in conducting research and transfer projects with software industry, Fraunhofer IESE offered enough experts to conduct a pilot study. In total, seven business area managers, one institute director, and one department head participated in the pilot study. Business area managers are senior consultants who establish and maintain contacts with industrial partners, acquire projects, and help transfer research results into industrial environments. Personal industrial project experience within the group of interviewees ranged from 5 to 17 years.

We used structured interviews for requirements elicitation. Each interviewee had to answer seven questions. All questions were formulated as open questions (i.e., "yes" or "no" answers were not feasible).

In order to help interviewees imagine concrete decision support tasks and situations in which a comprehensive SE-DSS might (or might not) be helpful, we offered three scenarios. A scenario consisted of a common part that served for setting the scene of management decision-making (i.e., what kind of information can be obtained, what is the basis for decision support, what is not available), and specific parts linked to the following: (1) quality manager, (2) project manager, and (3) product manager.

The questionnaire was developed in collaboration with an expert in cognitive psychology and was based on experience gained in previous projects (cf. for example [14]). Question 1 aimed at eliciting reasons for using a comprehensive SE-DSS (motivation). Question 2 aimed at identifying benefits of a comprehensive SE-DSS for improvement management on the organizational level. Question 3 aimed at getting an idea of the amount of user interaction that could be expected. Question 4 aimed at identifying the types of information that users need for comprehensive SE-DS. Question 5 aimed at prioritizing the different types of information needed by the users. Question 6 aimed at getting a better understanding about how query results should be presented to the user. Question 7 aimed at identifying other application areas (not mentioned in the scenarios) of comprehensive SE-DSS.

The questions not only aimed at eliciting requirements from potential future users of a comprehensive SE-DSS, but also to substantiate the validity of the scenarios offered to the interviewees.

The interviews were conducted as follows. Interviewees received the common part of the scenario description and two role-specific scenario descriptions a couple of days prior to the interview. When the interview started, first the role-specific scenario was presented to the interviewee. Then, the interviewee was asked to answer the questions from the perspective of the first role. When all questions related to the first role had been answered, the second role-specific scenario was presented to the interviewee, and the interviewee was asked to assume the second role and think about differences in the requirements for that role. ¾ of the time were assigned to the first role, ¼ to the second role. Eight of the nine interviews were recorded with an MP3 stick. In one case, a scribe recorded the interview on paper. All interviews lasted between 25 and 35 minutes.

Each interviewee was randomly assigned to two of the three specific roles. The set of questions was not sent to the interviewees in advance. Also, there was no communication between interviewees about the content of the interviews while the study was conducted.

Table 1 shows the random assignment of role-specific scenarios to interviewees (1 = quality manager; 2 = project manager; 3 = product manager). The purpose of having three scenarios for three different management roles was to find out whether these differ in their user requirements. If that was the case, the development of a comprehensive SE-DSS had to take these differences into consideration.

Interviewee	A	B	C	D	E	F	G	H	J
Role	1	1	2	1	1	2	1	1	2
Role	2	3	3	2	3	3	2	3	3

Table 1. Scenario assignments to interviewees

The procedure we used to aggregate and synthesize the answers given by the interviewees was inspired by the grounded theory approach [15]. We started the transcription with the first interview and the first question. Then we took the next interview and tried to find communalities and differences related to the first answer of the first question. If a similar answer was found, the counter of the first answer to the first question was set from 1 to 2. If no sufficient similarity was encountered, then the new answer from the second interview was added to the list of answers related to the first question. When all interviews were checked for question one, we repeated this procedure for question two, starting with the first interview. When an answer that was given to a question was found to be more related to another question, then this answer was re-assigned to that more relevant question, again following the procedure described above. After having processed all answers related to all questions, we double-checked that the aggregated and synthesized answers still represented sufficiently well the set of answers originally provided by the interviewees.

In addition to counting the occurrence of similar answers, a binary ranking was made: the interviewee (H) explicitly or intuitively expressed high importance of the response to the question, (M) either explicitly ranked it as medium important or did not clearly rank it as highly important. The process of aggregation and ranking resulted in Tables 2 to 6.

Why would you use a DSS in the given situation?			
		H	M
1.1	To get faster, broader, independent and empirically validated information about effectiveness and efficiency of a particular SE technology.	2	3
1.2	To answer the question: Which SE technique is most efficient / effective in a particular context (organization, process, product, documents)?	1	2
1.3	To get an overview on the existing techniques.	1	1
1.4	To get quantitative information (costs, quality level, defect reduction rate) about effectiveness and efficiency of a SE technique; people often tend to deliver qualitative information.	3	
1.5	Access to external information, which are otherwise not easy to get	2	2

Table 2. Motivation for DSS usage (question 1)

The fifth question was used to prioritize different types of information obtainable from controlled experiments. Answers to this question were aggregated with responses given to the fourth question (cf. Table 5).

How could a DSS contribute to organizational improvement management?		H	M
2.1	By connecting it with the internal software improvement management (One interviewee gave the hint that this connection will only work in one direction, i.e. data will be imported from the DSS into the organizational improvement management but not vice versa)	3 1	
2.2	By enhancing experience management	1	1
2.4	By benefiting from experience of others	3	

Table 3. Benefits for organizational improvement management (question 2)

Two alternative interaction strategies. 1. Similar to a search engine but more specialized. 2. Iterative refinement of the solution area by user model based interaction. Which strategy would you prefer, and why?		H	M
3.1	A combination of the alternatives is preferable	3	4
3.2	Transparency is important: Why did I get this result set? Access to the full set should be possible	4	
3.3	Not answering lots of questions, but fill in a template with checkboxes	2	
3.4	Especially in case of a huge result set, the second alternative becomes more attractive	3	1
3.5	Guidance for reducing the result set (e.g., use the context to reduce result set)	3	
3.6	Interaction has to be goal/problem oriented	1	

Table 4. Interaction preferences (question 3)

Results from empirical studies can be described and aggregated differently. Which information should be provided by the DSS?		H	M
4.1	Which techniques are available (information on a highly aggregated level)?	3	1
4.2	How effective/efficient is a certain technique with respect to which quality aspect?	3	2
4.3	Description of the process in which a SE technique shall be applied	1	
4.4	Costs for introducing/applying the SE technique	2	2
4.5	To get information about the impact a single SE technique has on the whole development process	1	
4.6	Information that allows for conclusions about the validity of empirical results associated with a particular SE technique	3	2
4.7	Context information (kind of system, programming language, process step)	2	1
4.8	Preconditions that have to be fulfilled prior to the application of the SE technique (e.g., skills, kind of documents available)	1	1

Table 5. Types of information needed (questions 4+5)

The seventh question was not intended to elicit new requirements but to confirm the relevance of our scenarios, and to identify new/other application areas for a comprehensive SE-DSS. Since the answers were not used for requirement elicitation, we omit the related table here. The relevance of the scenarios was confirmed. In addition, the answers confirm findings from question two, but on a more general level. For example, it was mentioned that a comprehensive SE-DSS could be used to educate new employees, or store (and maintain) project experience. Additionally, the available information might be used to focus future studies on SE technology effectiveness/efficiency, and thus help improve the coordination of empirical research.

How should the information be presented?			
		H	M
6.1	Profile for each SE technique (details on request)	1	1
6.2	Aggregated information in multiple graphical presentation	5	1
6.3	Easy-to-understand, self-explaining diagram	6	1
6.4	Easy-to-understand, self-explaining table	7	1
6.5	Executive management summary		1

Table 6. Presentation preferences (question 6)

In the next step, based on the aggregated and ranked answers of the interviewees, we sketched use cases from which an initial set of requirements could be derived.

3.2 Additional Requirements

As mentioned earlier, the pilot study was used to elicit requirements of potential end users of a comprehensive SE-DSS, i.e., software managers. Additional requirements that relate to the needs of content (data) contributors, administrators, and the sponsor of the comprehensive SE-DSS were derived from lessons learned we gained with setting-up and running web-based repositories. Table 7 lists these additional requirements. Requirements that emerge from the envisioned comprehensive DS method will also impact the internal system functionality, but are not considered here.

#	
AR1	Support for distributed contribution
AR2	Support for distributed quality management
AR3	Multi-role management
AR4	Multiple cross-linking of content items

Table 7. Additional Requirements

(AR1) Support for distributed contributions: It must be possible for the research community to contribute with new studies on SE technologies.

(AR2) Support for distributed quality assurance: AR1 requires at least some degree of quality assurance (QA). This could be organized, e.g., by establishing a QA assurance board that is in charge of approving new contributions. Only after approval is a new contribution integrated into the body of knowledge and made available to SE-DSS users.

(AR3) Multi-role management: AR1 and AR2 lead to two more different roles, i.e., contributor and QA. Contributors might be part of the QA board, but are not allowed to approve their own submission. Other possible roles are administrator, user, power user, guest, paying user, sponsor.

(AR4) Cross-linking of experience items: To enable the drawing of a landscape that visualizes the relationship between available empirical studies on the effectiveness and efficiency of SE technologies, it is necessary to cross-link repository items.

4 Structured List of Requirements

Based on the two sources of requirements, described above, we order the requirements according to the standard three-tier architecture, and according to Ruhe's generic requirements categories.

Table 8 lists the requirements derived from the pilot study and combines them with the additional requirements. The set of requirements is grouped into five categories: user interface (UI), presentation (PR), content (CO), experience management (EM), and repository (RE). The first two categories correspond to the first layer of the standard three-tier architecture, the third category corresponds to the second layer, and the fourth and fifth categories correspond to the third layer. Column three of Table 8 provides for each requirement the reference to related aggregated answers or additional requirements.

Table 9 shows the mapping of the requirements for the comprehensive SE-DSS to the framework "idealized" requirements (R1-R9) suggested by Ruhe [1]. The instantiation depends on our concrete problem topic, i.e., comprehensive SE technology selection, and usage scenarios, i.e., on-line, individual and strategic decision support for project, quality, and product management. One lesson we learned was that the framework was sufficiently generic to incorporate all of our specific requirements. The following remarks are helpful for interpreting Table 9.

(R1) We assumed that this generic requirement exclusively focuses on the management of the content. We moved requirement related to retrieval, presentation, and explanation – which are often associated with knowledge and experience management – to other generic categories.

(R2) Since we aim at a web-based implementation of the comprehensive SE-DSS, export interfaces have to be provided to allow for integration into an organization-specific improvement management infrastructure.

(R3) In the envisioned SE-DSS, the decision process will be supported by goal-oriented and problem-oriented interaction facilities.

(R4) Since we do not plan to support simulation, all listed requirements relate to process modeling.

(R5) Since the scenarios offered to interviewees aimed at individual decision-making, no requirements related to negotiation-support functionality were provided.

(R6) All requirements listed in this category reflect the needs of our target SE-DSS users.

(R7) Since the underlying comprehensive SE-DS was not presented to the interviewees, no specific requirements were given. Nevertheless, we listed those requirements that potentially have an impact on the underlying method.

(R8) Idem.

(R9) The listed requirements stress the distributed character of web-based comprehensive SE-DSS.

#	**User Interface**	Reference
UI1	Support for several kinds of graphical / textual presentations	6.1, 6.2, 6.3, 4
UI2	Low interaction, easy access	3.1, 3.4, 1.4
UI3	Goal-oriented interaction support	3.6, 3.5
UI4	Alternative interaction modes	3.1, 3.4, 3.3
	Presentation	
PR1	Transparency of decision process (reduction of alternatives, priorities)	2.1
PR2	Goal/problem-oriented aggregation of information	6.1, 6.2, 4.1, 4.3, 4.5, 4.6, 6.5
PR3	Understandable, self-explaining	6.3, 6.4
PR4	Presentation in diagrams, tables, text	6.1-6.5
	Content	
CO1	Effectiveness/efficiency with respect to quality aspect	4.2
CO2	Costs for introduction/applying the technique	1.4, 4.4
CO3	Preconditions that have to be fulfilled prior to the application of the technique	4.8
CO4	Context information	1.2
CO5	Structured meta information for the content	2.3, 4
	Experience Management	
EM1	Support for distributed contribution	AR1
EM2	Support for distributed quality assurance (distributed content management)	AR2
EM3	Support for export of repository data to organizational improvement management systems	2.1, 2.2
EM4	Multi-role management	AR3
	Repository	
RE1	Cross-linking of experience items	AR4
RE2	Case-oriented storing	4.5, 4.6, 2.4

Table 8 Requirements

Ruhe's Framework [1]	**Specific user requirements for comprehensive SE-DSS**
(R1) Knowledge, model and experience management	EM1-EM3, CO1-CO5
(R2) Integration into organization	EM3-EM4, RE1
(R3) Process orientation	CO3-CO5, PR1-PR2, EM4
(R4) Process modeling and simulation	CO3-CO5
(R5) Negotiation	--
(R6) Presentation and explanation	PR1-PR5
(R7) Analysis and decision	PR1-PR2, RE1-RE2
(R8) Intelligence	RE1-RE2
(R9) Group facilities	EM1, EM2

Table 9. Mapping to Ruhe's idealized requirements

5 Summary and Future Work

In this paper we have presented the requirements of a web-based tool for comprehensive decision-making in support of SE technology selection. The requirements were collected from a literature survey and from structured interviews with research managers.

The main contribution of our work is the empirical grounding of the requirements collection, by conducting scenario-based structured interviews with nine Fraunhofer IESE managers. Besides the identification of requirements, the research yielded the following results: All of the interviewees accepted the predefined scenarios as being relevant and practical, none had difficulties with understanding. We interpret this finding to support the construct validity of our measurement instrument (scenario-based structured interviews).

Surprisingly, we did not find much difference between management roles. Apart from prioritization of content presentation (question 5), the answers given were very similar, no matter which specific role was assigned to an interviewee. At the moment, it is not fully clear whether this indicates that differences between roles are not as large as we originally expected, or whether the answers given by the interviewees were too strongly influenced by the way role-specific scenarios were presented to them. Also, the subjects might not be fully representative for the specified roles due to the nature of their work in research environments, which is probably not as strongly focused on actual (and mostly short-term) decision-making within software projects.

Some of the subjects were skeptical if they would use such a web-based DSS, as they felt unable to estimate the actual power of such a system in supporting them in their job. This partly seems to reflect the common fear that web-based information sources potentially create information overload.

Future work is dedicated to the incremental development of the comprehensive SE-DSS. At each stage, the underlying method and the resulting tool will be evaluated through controlled experiments and surveys among experts from academia and industry. Issues to be evaluated include effectiveness and efficiency of the method and tool support, as well as validity of the delivered information and completeness of the database.

References

[1] Ruhe, G.: "Software Engineering Decision Support – A new paradigm for Learning Software Organizations". In: *Proc. Workshop. Learning Software Organizations*, Springer, 2003.

[2] Ruhe, G.: "Software Engineering Decision Support: Methodology and Applications". In: *Innovations in Decision Support Systems* (Ed. by Tonfoni and Jain), International Series on Advanced Intelligence Volume 3, 2003, pp 143-174.

[3] Wohlin, C.; Petersson, H.; Aurum, A.: "Combining Data from reading Experiments in Software Inspections". In: *Lecture Notes on Empirical Software Engineering*, World Scientific Publishing Co. Pte. Ltd, Singapore, 2000.

[4] Pickard, L.M.; Kitchenham, B.A.; Jones, P.W.:"Combining empirical results in software engineering", *Information and Software Technology*, 40(14):811-821,1998.

[5] Runeson, P.; Thelin, T.: "Prospects and Limitations for Cross-Study Analyses – A Study on an Experiment Series". In Jedlitschka, A. & Ciolkowski, M. (Eds) *Proc. of 2nd Workshop on Empirical Software Engineering*, Roman Castles, 2003, pp. 136-145.

[6] Jedlitschka, A.; Pfahl, D. and Bomarius, F.: "A Framework for Comprehensive Experience-based Decision Support for Software Engineering Technology Selection"; In *Proc. of Intern. Conf. SEKE 2004*. Banff, Canada, 2004

[7] Christie, A. M.: Simulation: "An Enabling Technology in Software Engineering", *CROSSTALK – The Journal of Defense Software Engineering*, pp. 2-7, April 1999.

[8] Pfahl, D.: *An Integrated Approach to Simulation-Based Learning in Support of Strategic and Project Management in Software Organisations*, University of Kaiserslautern, Germany; Stuttgart: Fraunhofer IRB Verlag; 2001.

[9] Raffo, D. M.: *Modeling Software Processes Quantitatively and Assessing the Impact of Potential Process Changes on Process Performance*, Carnegie Mellon University, Pittsburgh, PA, UMI Dissertation Services #9622438, 1996.

[10] Jedlitschka, A.; Ciolkowski, M.: "Towards Evidence in Software Engineering"; In *Proc. of ACM/IEEE ISESE 2004*, Redondo Beach, California, August 2004, IEEE CS, 2004.

[11] Li; Ruhe, G.: "Web-Based Decision Support for Software Release Planning", in: *Proc. of WI/IAT 2003 Workshop on Applications, Products and Services of Web-based Support Systems*, Halifax, 2003, pp 13-20.

[12] Biffl, S.; Halling, M.: "A Knowledge Management Framework to Support Software Inspection Planning", in [16]

[13] Conradi, R.; Wang, A.I.: *Empirical Methods and Studies in Software Engineering – Experiences from ESERNET*; Springer LNCS 2765, 2003.

[14] Jedlitschka, A.; Nick, M.: "Software Engineering Knowledge Repositories"; in [13] pp.55-80

[15] Strauss, A. & Corbin, J.: *Basics of Qualitative Research. Techniques and Procedures for Developing Grounded Theory*. 2nd ed. Thousand Oaks: Sage, 1998.

[16] Aurum, A.;Jeffery, R.; Wohlin, C.; Handzic, M. (Eds): *Managing Software Engineering Knowledge*; Springer-Verlag; Berlin 2003

Document Information

Title:	Requirements for a Tool in Support of SE Technology Selection
Date:	February 2004
Report:	IESE-017.04/E
Status:	Final
Distribution:	Public