From Design Rationale to Reengineering Rationale: Lessons Learned in a Maintenance Pilot Case Study

Maria Istela Cagnin *[1], Débora M. B. Paiva[1♦], José Carlos Maldonado[1], Rosângela Dellosso Penteado[2], Renata P. M. Fortes[1], Fernão Stella R. Germano[1]

[1]Universidade de São Paulo – USP
Instituto de Ciências Matemáticas e de Computação
São Carlos-SP, Brazil, CEP 13560-970
{istela, debora, jcmaldon, renata, fernao}@icmc.usp.br

[2]Universidade Federal de São Carlos – UFSCar
Departamento de Computação
São Carlos-SP, Brazil, CEP 13.565-905
rosangel@dc.ufscar.br

Abstract. *Software can be considered an organizations asset, because it evolves and incorporates value as business rules change. So, it is important that good techniques be used in software development, in order to assure that its lifecyle is extended. One of these techniques is Design Rationale, which documents all the project decisions made during software development in order to ease revision, maintenance, documentation, evaluation and project learning. Design Rationale has been used in this paper in a context different than software development, in the reengineering of an electronic repair shop control legacy system, and the term Reengineering Rationale (RR) is used. In order to observe the behaviour of that technique in this context, a pilot case study has been conducted for analysing its importance to perfective maintenance of systems, resulting of reengineering. For this pilot case study two hypotheses have been formulated, one concerning the support provided by RRs to perfective maintenance and the other to maintenance time reduction. This paper discusses the results obtained and the lessons learned of a pilot case study conducted in academic environment.*

Key words: Design Rationale, Maintenance, Pilot Case Study.

1. Introduction

Software is a product that evolves continuously to satisfy its user needs. Furthermore, it is considered as an organizations asset, because it evolves and incorporates value as the business rules change. So, it is necessary that good techniques be used in its development or reengineering, in order to ease maintenance activities and, consequently, extend its lifecyle, assuring the return of investiments done by the organization. *Design Rationale* (DR) is one of the techniques that have to be used as, according to Gruber & Russel [13] e Moran & Carroll [16], it explains "how" and "why" an artifact has been designed in a certain way. So, DR can describe: (1) the reasoning used to obtain a certain final design, for example, how the system architecture meets the desired functionalities; (2) why certain architecture has been

* Financial support by FAPESP, # 00/10881-4

♦ Financial support by Capes

choosen over the other alternatives; and (3) which system behaviour is expected and under which operational conditions. Under the same perspective, Lee [15] considers that DRs are important because they include the reasons behind a design decision and their justification, the other alternatives considered and all the argumentation that led to the decision.

As DR offers help in conducting revision, maintenance, documentation, evaluation and project learning activities [6], it can be extremely useful for reuse of previous projects [1, 20], for coordination of people that belong to a team work [9], for promotion of critical reflection during project development [11] and for artifact maintenance [2]. DR concepts are extended in this paper to the reengineering context and the *Reengineering Rationale* (RR) term is used.

The objective of this paper is to show the planning, execution, results and lessons learned of a pilot case study in order to evaluate the importance of RRs in the perfective maintenance of an electronic repair shop control system, resulting of a reengineering process application. For this pilot case study two hypotheses have been formulated, one concerning the support provided by RRs to perfective maintenance and the other concerning that maintenance time reduction.

In Section 2, the related work is discussed. In Section 3 the pilot case study definition and planning is presented, based on Wholim et al. purpose [22]. In Section 4 the pilot case study execution is reported. In Section 5 the results obtained and lessons learned are discussed. In Section 6 the final remarks and suggestion for future works are presented.

2. Related Work

According to Tervonen [21], Rittel [18] was the first to advocate systematic documentation of DR as part of design, and the origin of recorded software design rationales can be traced to Freeman's paper [12] in which he explains how these could improve design review.

Recent research has the tendency to combine DR systems with design support tools in different areas, such as, Mechanical Engineering and Civil Engineering. Furthermore, different forms of DR capture have been investigated with the objective of making that activity more automatic, less intrusive and less expensive.

Empirical work has been done in recent years with the objective of investigating how DR can be applied in project development. The results obtained show that usually the efforts spent in DR capture are worth doing, considering the benefits that can be obtained when maintenance, reuse and project learning activities are conducted. Such results show the value of carrying out empirical studies with respect to using DR information.

Karsenty [14] evaluated the use of DR documents in mechanical engineering design. The goal of that study was to evaluate how useful DR documents are. Six experienced professional designers were asked to understand and to assess a previous design. These tasks were chosen because they are considered by the designers as very important when they need to work on the results of a previous study. These designers were provided with documents that described the solution and other documents describing DR. They were free to use blueprints and DR as they chose. To determine the usefulness of DR documents, the author attempts to answer the three following questions: (1) Do designers confronted with an unknown design need to know the design rationales? (2) How designers use design rationale documents? (3) Do we succeed in capturing the rationales looked for by designers? It was concluded that DR is useful for some designers who use it as support to their reasoning, but it is not sufficient to completely answer the designers questions.

Conklin and Burgess-Yakemovic [10] did field trials for software development using extensions of a DR scheme, named IBIS [9]. They studied planning design meetings in an industrial setting for eighteen months. This case study identified key points during the technology transfer issues, when introducing rationale methods, such as the presence of a team leader and a clear argument supporting data relative to the costs and benefits of the approach. They found that capturing DR is useful during the requirements analysis and design stages.

Bratthall et al [5] carried out a controlled experiment where the value of having access to a retrospective DR is evaluated both quantitatively and qualitatively. The hypothesis was that changes would be faster and more correct if such information was available during change impact analysis. Realistic change tasks were applied by seventeen subjects from both industry and academia on two complex systems from the domain of embedded real-time systems. The results from the quantitative analysis shown that, for one of the systems, there was a significant improvement in correctness and speed when subjects have access to a DR document. In the qualitative analysis, DR was considered helpful for speeding up changes and improving correctness.

Shull et al. [19] introduce an empirical methodology, based on experiences gathered over more than two decades of work, for transferring development processes from the conceptual phase to industry. The methodology presents a series of questions that should be addressed, and that evaluate and provide feedback for the four steps of the methodology (feasibility study, observational study, case study: use in real lifecycle and, case study: use in industry). This paper pilot case study complains with the methodology two first steps.

3. Pilot Case Study Definition and Planning

Pilot Case Study Definition:

Object of study: Reengineering *Rationale* (RR).

Purpose: Piloty case study to validate an experiment design, which aims to evaluate the importance of RRs in the perfective maintenance of an electronic repair shop control system, resulting of a reengineering process application.

Quality focus: Importance of RRs usage to support maintenance.

Perspective: Related to software engineers interested in RRs usage to support system maintenance.

Context: The pilot case study planned for twelve graduate (master and doctor degrees) students. The following artifacts are available for the pilot case study conduction: requirements document, system class diagram, use case diagram, business rules documentation, system user manual, system source code in Smalltalk language with about 2.2 KLOC. These artifacts have been created in a reengineering process application using the PARFAIT agile process [7]. The pilot case study has two treatments (that is, comparison of the maintenance activity with and without RRs usage).

<u>Pilot Case Study Planning</u>:

Context Selection: The system to be maintained is the result of a reengineering process application to a legacy system that controls entry and exit of electronic appliances in a repair shop. Legacy system, originally developed in Clipper, has been migrated to Smalltalk and MySQL DBMS [17]. The resulting system has about 2.2 KLOC, considered of medium size.

The RRs have been collected by a software engineer, immediately after the reenginnering end and are documented in IBIS format, which records questions arosen during the project as well as all the positions and argumentations of designers about each question. An example of RR is presented in Figure 1. The argumentation considered by the software engineer has a (✓) mark. The pilot case study results aim to supply important information to other software engineers about the possibility to use RRs to support systems maintenance.

...			
Question			
Name	How long has the repair shop to wait for the appliance to be retrieved before discarding it?		
Status	Ready		
Date	March 10, 2003		
Position			
Title		The repair shop must wait six months before discarding the appliance, in case the owner does not retrieve it within that period.	
Key words		-	
Content		-	
Against Argumentation		**Favor Argumentation ✓**	
Title	Too long	Title	Reasonable Time
Key words	-	Key words	-
Content	Too long waiting time. Could be a maximum of one month. In that case, it would not be necessary for the repair shop to have physical space available to store non-retrieved appliances.	Content	Reasonable time for waiting the appliance retrieval by its owner without judicial damage to the repair shop.

Figure 1. RR Example

The activities that are part of the training phase for the pilot case study conduction refer to learning of Smalltalk language programming (four hours), of the VisualWorks 5i.4 environment (one hour), of the MySQL DBMS (one hour), of the *Design Rationale* concept and rationale representation schema, named IBIS (four hours). For completing these activities a twelve hours time is estimated.

The maintenance activity is composed of two phases. The first refers to a new functionality inclusion related to the consultation of electronic appliances that have been repaired and were not retrieved in a certain period of time. The second refers to a new functionality inclusion related to the destination storage (that is, donation, sale or auction) that will be given by the system to appliances not retrieved (first phase). For completing that activity forty hours time is estimated. Thus, for the pilot case study conduction a fifty-two hours total time is estimated. The time spent in each maintenance phase to be recorded in a sheet, according to a template supplied. The students could estipulate their own time schedule to conduct the proposed maintenance activity.

This pilot case study is inserted in the Maintenance context of the Software Engineering domain.

Hypothesis definition: During planning the following hypotheses were formulated.

Null Hypothesis, H1: The *RRs* has not supported the system understanding as information useful to conduct the perfective maintenance has not been found.
Null Hypothesis, H2: The time to do the perfective maintenance is larger when *RRs* are available.

Alternative Hypothesis, H1: The *RRs* supported the system understanding as information useful to conduct the perfective maintenance has been found.
Alternative Hypothesis, H2: The time to do the perfective maintenance is smaller when *RRs* are available.

Data Collection: To obtain the case study results and, consequently, to justify the hypotheses, different information is collected before, during and after the study, as presented in Table 1.

Table 1. Data collected about the pilot case study conducted

Data collection period	**Data collected**
before case study	Personal data (school level, specialty); OO paradigm, Smalltalk, application domain, MySQL DBMS, DR skill; overall and specific knowledge about the software maintenance activity.
during case study	Maintenance time (in minutes); Which doubt led the maintainer to consult RRs? Which RRs were used to answer the doubt raised? Which questions/doubts arouse during the maintenance activity? Of these, which have been answered by RRs? Which approach has been used to consult the RRs (previous reading or reading according to the raise of doubts during the maintenance activity)? Has the available system documentation been sufficient to support the maintenance activity? In the negative case, which other types of document or of information in the available documents could be necessary? Brief description of the steps followed to do the maintenance.
after case study	Difficulties found; Whether the available system documentation has been sufficient; Other documents suggested; Observations; Suggestions; Consults to customers about DR.

Variable selection: The variables selected in the pilot case study planning are the following: **independent** – OO paradigm, Smalltalk language, VisualWorks 5i.4 environment and application domain student skill; school level; student specialty in computer areas; and amount of information useful for maintenance conduction, found in the available RR set; **dependent** – total time spent to do the maintenance activity.

Selection of subjects: The technique chosen to conduct the pilot case study has been stratified random sampling (that is, the population is divided in a number of groups with a non distribution among groups), as the subject groups have been created from a profile analysis, obtained with the application of a specific questionnaire.

Pilot case study design: The pilot case study has been designed to be conducted by twelve subjects, divided in four groups of three subjects, as shown in Table 2, considering that the first two used RRs and other two did not. The maintenance activity has to be conducted effectively with the participation of all group members. In case that has not been possible during the whole pilot case study, it has to be recorded in which phase it has not been possible and why, according to the template supplied.

Table 2. Pilot case study design

Group	Maintenance with RRs support	Maintenance without RRs support (ad hoc)
G1	X	
G2	X	
G3		X
G4		X

Instrumentation: The following instruments have been used by the subjects during the pilot case study conduction: requirements document, business rules documentation, system user manual, system class diagram, use case diagram, electronic repair shop control system source code (all produced during the reengineering using the PARFAIT[1] agile process [7]), RR set (only for the subject groups that conduct the maintenance with its support – G1 and G2), maintenance activity plan, pilot case study data collection form and material used in training.

Validity evaluation: To verify the validity of the results obtained in the pilot case study, the following validity threats have been established:

- *Conclusion validity*: Some data collected are subjective (as for example: the system available documentation is or is not sufficient; the RRs description is or is not clear, sufficient or not sufficient, etc) and depend on the skill of the software engineer that is conducting the maintenance;

[1] PARFAIT is the acronym for *Processo Ágil de Reengenharia baseado em FrAmework no domínio de sistemas de Informação com VV&T* (in Portuguese), which means "Framework-based Agile Reengineering Process in the Information System Domain with VV&T".

- *Internal validity:* the pilot case study has to be conducted by the subject without any restriction of day, time and place, that is, he/she can conduct it in the moment he considers more proper.The subjects selected are not volunteers. That can influence the results. There is a possibility of communication between subjects (groups) participating in the pilot case study, but the groups are motivated not to publish data related to the pilot case study;
- *Construct validity*: as the pilot case study has to be conducted by more than one subject it is necessary to know how many years of experience they have so as to classify data obtained in different categories. The RRs have been recorded by only one individual (that conducted the reengineering). That record has been made only after the reengineering completion;
- *External validity*: this pilot case study has to be conducted in the academic environment using a system resulting of a reengineering process application. So, there is difficulty to generalize results for the industrial environment.

4. Pilot Case Study Conduction

The pilot case study has been conducted during the second semester of 2003 by twelve graduate students (ten master degree and two doctor degree). All students had already coursed Software Engineering disciplines. The ten master students were enrolled a discipline Software Engineering Special Topics of the Graduate Program in Computer Science at the Federal University of São Carlos. The two doctor students were enrolled in the Graduate Program in Computer Science at the ICMC-USP.

The pilot case study execution has been conducted in three phases. In the first phase, the twelve hours training have been conducted on the techniques envolved in the study, according to what has been planned. In the second phase, a presentation of the plan and of the pilot study case objective has been done for the participants. In the third phase, the maintenance activity plan has been distributed to the participants together with the source code and system documentation, RRs (only for Groups G1 and G2), data collection form of the pilot study case and the material used in training, so as to allow the participants to use it in case of doubts about the techniques. Each group delivered, after two month time, the electronic repair shop control system with the maintenance conducted and the data collection form filled. Then, a meeting has been conducted in which all the participants of each group reported: a) the steps followed to conduct each maintanance activity phase; b) results of the maintenance activity (screen and reports demonstration, etc); c) whether the training given has been sufficient; d) how the work has been articulated when some group member was not present (via e-mail, ICQ, etc), e) main doubts arose; f) main difficulties faced; g) whether there was been interaction between groups, in the positive case, how it has been done.

5. Results Obtained and Lessons Learned

Several lessons have been learned with the conduction of the pilot case study and will cooperate for the conduction of an experiment, aimed to obtain the desired statistical significance, as they have raised several problems, both before the pilot case study (planning and training) and during its conduction. These lessons learned have been based on the data collected during and after the pilot case study. At the end of this section, a summary of the data colleted after the pilot case study is presented (Table 5).

A specific training in software engineering has not been offered to the subjects participating of the pilot case study, as it has been believed that they had sufficient skill in software engineering, according to the results of the profile questionnaire that has been submitted to the subjects. However, during the pilot case study conduction, it has been observed that some of the questionnaire results have not portraited reality, so, it should be stressed that the questionnaire subjectivity has to be considered in the validity threats. Another form to evaluate the subject skill has to be conducted or software engineering basic concepts training should be supplied so as to level the subjects knowledge.

For the groups creation, the answers given individually to the subjects profile questionnaire have been considered, as mentioned before. Trying to obtain homogeneity among groups, the students have been allocated according to their skill level. But, as all questions pertaining the subject knowledge were subjective, the absence of quantitative information has been observed (for example, undergraduate marks average, graduate marks average (common disciplines), undergraduate course name, course emphasis, course institution, etc), so as to make group creation more adequate and homogeneous. Even so, the questionnaire questions may not review the subject skill level, so some profile questionnaire answers must be considered in the validity threats.

The subjects have had great difficulty in the Smalltalk language, so, according to the feedback questionnaire (Table 5) applied after the pilot case study, the need to dedicate more time to the practical training in the Smalltalk language has been observed. Another point observed was that the groups without RRs support have not followed correctly the pilot case study conduction, as they have not consulted the customers. This should have been done when the information necessary to the maintenance activity conduction was not available. A more rigorous follow-up by groups has to be done to avoid that deviation.

Group 1 conducted the pilot case study with only two components. One of these had enough knowledge of the GREN framework (that is, **G**estão de **RE**cursos de **N**egócios, in portuguese), [3, 4]), done some instantiations and even framework modifications, totaling about thirty-six hours of experience. The authors were not aware of these component experiences with the GREN framework, as the subjects profile questionnaire did not have a question about it. That information is important, as the system reengineering has been based on GREN, so all the system classes structure is based on the framework class hierarchy and its knowledge may influence the analysis of the data collected. The Smalltalk language and the GREN framework knowledge of the other participants has occurred only after the training given. No participant had DR knowledge.

The time in hours spent by the groups in the maintenance activity is presented in Table 3. Due to the visible time discrepancy among groups, the need to obtain more information about subjects profile of each group has been felt, that is, undergraduate marks average, graduate marks average (common disciplines), in order to better analyse the data. Group 1 members have studied in a Computing Engineering undergraduate course, whereas the others have had Computer Science. Furthermore, as already mentioned, one of the members was already experienced in Smalltalk language and GREN framework. That explains the significative difference among Groups 1 and 2, even though Group 1 had one component less. The difference among Groups 2 and 3 is acceptable and, analysing other data of the students profile, it has been observed that almost all these groups components have taken an undergraduate course in the same institution, with the exception of only one Group 2 component. The time spent by Group 4 has been very different from the others. This can be explained by its components disinterest, noted during the pilot case study conduction and by the exactness of time intervals spent annotated in the template of the supplied sheet.

Table 3. Time spent by groups on the maintenance activity

Group	Subject	Maintenance with RRs support	Maintenance without RRs support (ad hoc)
G1	Component 1, Component 2	**10:30 hs**	-
G2	Component 4, Component 5, Component 6	**17: 30 hs**	-
G3	Component 7, Component 8, Component 9	-	**19:50 hs**
G4	Component 10, Component 11, Component 12	-	**32 hs**

With relation to Hypothesis 1, the RRs supported partially the system understanding for doing perfective maintenance. Group 1 stated during the pilot case study data collection that the RRs supported about 30% of the maintenance activity. However, Group 2 stated that has not had support. On the other side, during the results oral presentation, the Group stated that consulted the RRs as doubts arose and that they have helped to answer some of the existing doubts. So, the RRs supported, even partially, the maintenance activity conducted. About Hypothesis 2, it has been observed, as shown in Table 3, that the time spent to do the perfective maintenance is lower when RRs are available.

In Table 4 the planning parts are presented, identified during the pilot case study results evaluation, that have to be altered for the experiment conduction.

Table 4. Summary of planning alterations

Planning parts altered	Content altered
Construct validity	- consider the subjectivity of the software engineer profile questionnaire. - consider the answers related to the software engineer knowledge, as they may not reveal the true subject knowledge level.
Profile questionnaire	- obtain quantitative data about the knowledge, that is, undergraduate average marks level, graduation average marks level (common disciplines), undergraduate course name, course emphasis and course institution. - include a question about the GREN framework skill.
Training	- provide software engineering basic concepts training to level the subjects knowledge. - dedicate more time to Smalltalk language practical training.
Execution	- do more precise follow up of the experiment responsible persons, so as to avoid deviation from what has been planned.

In Table 5, a summary of the answers given by the groups after the pilot case study conduction is presented.

Table 5. Summary of feedback questionnaire

Questions	G1	G2	G3	G4
Difficulties found	wrong understanding of the maintenance that should have been conducted, implied in GREN modification, that has been done thanks to skill of one group member.	group members did not know the information that should have been presented in the report (doubts have been answered via e-mail).	GREN and Smalltalk language skills.	Smalltalk skill.
System documentation has sufficed?	Yes	No	No	No
Other documents suggested	-	GREN documentation.	DB details, GREN manual, maintenance details.	-
Suggestions	Number the RRs	-	more classes using GREN.	more Smalltalk training

continued

Questions	G1	G2	G3	G4
Observations	have not previously read the RRs.	have previously read the RRs.	code has been extensively studied.	-
Consult to customers about RRs	-	-	no	no

6. Final Remarks and Future Work

The result of the pilot case study allowed: a) improvement in the plan initially defined, mainly with respect to construct validity, profile questionnaire, training, pilot case study execution, as presented in Table 4; b) lessons learned and, mainly, traps that could invalidate the experiment statistical significance; c) observation of pilot case studies importance before the experiment, in order to identify problems both in planning and those arose during the experiment conduction, so as to avoid resources and associated costs waste.

Even being a pilot case study, with problems identified both in the planning and in the pilot case study conduction, it has been observed that RRs documented during the system reengineering, have supported its understanding to conduct perfective maintenance and the time spent in the perfective maintenance activity is lower when RRs are available. It has also shown the importance of the DR technique applicability also in the reengineering context.

Results, found in the literature, demonstrate the value of carrying out empirical studies with respect to using DR information. Our pilot case study has also supplied indications of that fact.

As presented in Section 2, Conklin and Burgess-Yakemovic [10] found that capturing DR is useful during the requirements analysis and design stages. With the pilot case study reported in this paper it has been observed that DR capture is useful also in reengineering to ease fratture maintenance.

The hypothesis of Bratthall et al [5], that changes would be faster and more correct if DR was available during impact change analysis, was confirmed by the pilot case study. In this paper it has been observed, by quantitative analyses, that the maintenance time with RRs has been lower than the maintenance time without RRs.

With the results obtained and lessons learned in the pilot case study conducted, the authors will do the experiment to obtain results that can be analysed with statistical significance. With that, it will be possible to confirm the results obtained in the pilot case study. Besides the conduction of the experiment in the academic environment, it is also intended to apply it in the industrial environment. That will allow attendance of the last two steps of the Shull et al. methodology [19], that is, case study: use in real lifecycle and, case study: use in industry.

References

[1] Ball, L., Lambell, N., Ormerod, T., Slavin, S., and Mariani, J. (1999). *Representing Design Rationale to Support Innovative Design Reuse: A Minimalist Approach*. In Proceedings of the 4th Annual Design Research Thinking Symposium - MIT.

[2] Boy, G. (1995). *Supportability-based Design Rationale*. In Proceedings of the 6th IFCA Symposium on Analysis, Design and Evaluation of Man-Machine Systems, Boston, MA, USA.

[3] Braga, R.T.V. *A Process for Construction and Instantiation of Frameworks based on a Domain-Specific Patterns Language*. Sc.D. Thesis – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos-SP, Brazil, 224 p., 2003. (in portuguese).

[4] Braga, R.T.V.; Masiero, P.C. *A Process for Framework Construction Based on a Pattern Language*. In: 26th Annual International Computer Software and Applications Conference (COMPSAC'2002), IEEE, Oxford, England, p.615-620, August, 2002.

[5] Bratthall L., Johansson, E., Regnell, B. *Is a Design Rationale Vital when Predicting Change Impact? A Controlled Experiment on Software Architecture Evolution*. In: Proceedings of the Second International Conference on Product Focused Software Process Improvement, Springer-Verlag, 2000, pp. 126–139.

[6] Burge, J. and Brown, D. C. (2000). *Reasoning with Design Rationale*. In Proceedings of Artificial Intelligence Design Conference.

[7] Cagnin, M.I.; Maldonado, J.C.; Penteado, R.; Germano, F. *PARFAIT: Towards a Framework-based Agile Reengineering Process.* In: Agile Development Conference, Salt Lake City, Utha, EUA, 25-29 June, 2003.

[8] Cincom Systems Inc (2001). *Smalltalk Visual Works NonCommercial*, 5i.4 release. Version 5i.4. URL: http://cincom.br/visualworks/. Accessed: March, 2004.

[9] Conklin, J. and Begeman, M. L. (1988a). *gIBIS: A Hypertext Tool for Exploratory Policy Discussion*. ACM Transactions on Office Information Systems, pages 303–331.

[10] Conklin, J.; Burgess-Yakemovic, K. *A Process-Oriented Approach to Design Rationale*. In: Design Rationale Concepts, Techniques and Use, T. Moran and J. Carrol (editors), Lawrence Erlbaum Associates, p. 293–428, 1995.

[11] Fischer, G., McCall, R., and Morch, A. (1989). *Design Environments for Constructive and Argumentative Design*. In Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'89), pages 269–275.

[12] Freeman P. Toward improved review of software designs. Proceedings of the National Computer Conference, 1975.

[13] Gruber, T. R.; Russel, D. M. *Design Knowledge and Design Rationale: A Framework for Representation, Capture, and Use*. Technical Report KSL 90-45, Knowledge Systems Laboratory, Standford, California, 40 p., 1991.

[14] Karsenty, L. (1996). *An Empirical Evaluation of Design Rationale Documents*. In Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'96), pages 13–18, Vancouver, BC.

[15] Lee, J. (1997). *Design Rationale Systems: Understanding the Issues*. IEEE Expert, pages 78–84.

[16] Moran, T. P.; Carroll, J. M. *Design Rationale: Concepts, Techniques, and Use Computers, Cognition, and Work*. New Jersey: Lawrence Erlbaum Associates, 1996. 659 pages.

[17] MySQL (2002). *MySQL Reference Manual for Version 3.23*. URL: http://web.mysql.com/. Accessed: February, 2002.

[18] Rittel, W.H.J. On the planning crisis: Systems analysis of the first and second generations. Bedriftsøkonomen, vol.34, n.8, 1972.

[19] Shull, F.; Carver, J.; Travassos G. H. *An Empirical Methodology for Introducing Software*. In: 8th European Software Engineering Conference (ESEC'2001). Vienna University of Technology, Austria, September 10-14, 2001.

[20] Stutt, A. and Motta, E. (1995). *Recording the Design Decisions of a Knowledge Engineering Community to Facilitate Re-use of Design Models*. In Proceedings of the 9th Knowledge Acquisition for Knowledge-Based System Workshop (KAW'95), Banff, Canada.

[21] Tervonen, I. Quality Derivation, Refinement and Generalization in Object-Oriented Software Construction, IEEE, 1993.

[22] Wholim, C.; Runeson, P.; Höst, M.; Ohlsson, M.; Regnell, B.; Wesslén, A. *Experimentation in Software Engineering*. Kluwer, 2000.