OSLO UNIVERSITY
Department of Informatics

The background information on subjects in program comprehension studies

Rolf Vassdokken

Master of Science Thesis

May 2005

ABSTRACT

Program comprehension is a very important skill a software engineer need. Many researchers conduct experiments on program comprehension in order to improve tools, documentation, and maintenance guidelines supporting program comprehension. Individual programmers' productivity might vary significantly even though they have similar background. Thus, the subjects' background is very important when conducting and analyzing experiments on program comprehension. The survey presented in this short Master thesis identifies subjects background information reported in software experiments on program comprehension. The background information reported in 24 articles was systematically analyzed in order to answer what kind of background information is reported and how the background information was used in the analysis.

The articles reports many different background variables, but the overall impression of the background information reported in program comprehension experiments is that it is rather arbitrary and small. The analysis shows that there is a need for standards and guidelines of how to collect and report subjects' background information. The survey shows also that almost no background information of the subjects is used in the experiments' analysis. The articles in this survey provide so little information about the subjects' background that it is difficult to perform replications and meta-analysis. This thesis aims to make researchers more aware of the subjects' background in their experiments and reports.

On the basis of the results of the analysis I have suggested background variables that should be collected in comprehension studies and proposed a background questionnaire. The questionnaire was used in an experiment with 24 subjects from the industry. I report here experiences with the questionnaire.

ACKNOWLEDGEMENTS

This thesis was written as a part of a controlled experiment conducted at Simula Research Laboratory for my Master of Science degree at the Department of Informatics, University of Oslo.

First of all I want to thank my advisor Amela Karahasanović for taking me into her project and helping me in the job writing this thesis. She has encouraged and supported me during this semester writing this short thesis, come with valuable contributions and reading my thesis. I will also like to thank Gunnar Carelius for technical assistance and usage of the web-tool SESE, and Johan Almqvist for the collaboration implementing the experiment into SESE. I will also thank the Simula Research Laboratory for supported facilities during my studying and writing.

Finally I want to thank my family and all others who supported me during my time studying for my master degree, and specially Gunn for always having faith, confidence and believe in me.

Oslo, May 2005
Rolf Vassdokken

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

1.1 Motivation

One of the core software engineering activities is to comprehend programs. When you maintain, reengineer, inspect, reuse, migrate, or enhance software systems you need program comprehension. Program comprehension is the process of acquiring knowledge about a computer program, and is very important because the majority of the software development effort is spent on maintaining existing software systems. Studies show that after the implementation of a software system, the programmers use more than 50% of their working time on changes. (Zelkowitz 1978; Lientz 1983; Lehman and Belady 1985; Pfleeger 1987; Nosek and Prashant 1990; Coleman, Ash et al. 1994; Holgeid, Krogstie et al. 2000). Program comprehension is also becoming more important because the software programs tends to become larger and more complex, and studies shows that program comprehension is taking up to 60% of total time devoted to maintenance (Lucia, Fasolino et al. 1996; Dunsmore, Roper et al. 2000).

Researchers have conducted several empirical studies in order to understand program comprehension (Mayrhauser and Vans 1996; 1997; Ramalingam and Wiedenbeck 1997; Mayrhauser, Vans et al. 1998; Ramalingam and Wiedenbeck 1999; Corritore and Wiedenbeck 2000; 2001; Wiedenbeck and Engebretson 2002; Parkin 2004; Wiedenbeck and Engebretson 2004). Different comprehension models have been studied: direction and breadth. The direction of comprehension model is divided into a top-down (Soloway, Ehrlich et al. 1982; Brooks 1983), a bottom-up strategy (Schneiderman and Mayer 1979; Pennington 1987), or a mixture of them both (Letovsky 1986; Mayrhauser and Vans 1995; 1996; 1997). Littman, Pinto et al. (1986) discusses the scope or breadth of comprehension where a systematic or as-needed strategy is used. Many of the experiments on program comprehension have been performed and analyzed with the purpose of aiding the development of tools, documentation, maintenance guidelines and training routines that can help simplify program comprehension tasks, and thereby improve software engineers' program comprehension. Analyses from these kinds of experiments show a focus on the programming effort and comprehension. It is important to look at how well the tasks have been solved by each individual subject, but the results need to be carefully evaluated in context with the subjects' background and experience. To be able to perform adequate meta-analysis and replications the subjects' background information is important. How is the subjects' background information used in program comprehension experiments? Is the subjects' background taken into consideration in the analysis of program comprehension experiments, and how is it done?

The aspects in software engineering can be divided into people, process and technology (Runeson 2003). Research and experiments are complicated due to that people are quite heterogeneous in contradiction to technology and processes that can be controlled. The productivity between individual programmers with similar background might vary significantly (Brooks 1980). Brooks (1983) initially created his comprehension model to explain among others the individual differences between persons' abilities to comprehend a

program's purpose. Why does one person find a program easier to comprehend than does other? This is a question researchers try to figure out.

The subjects in program comprehension experiments are usually described as a homogeneous group of people categorized as novices, students, professionals or experts, and recruited from the industry and/or universities. But are they really homogeneous? Even if the subjects are categorized as e.g. students they will have taken different courses and some will perhaps have work experience also. Thus, the background information needs to be taken into consideration when performing the analysis. What kind of background information should be collected in program comprehension studies? Sjøberg et al. (2004) concludes in their survey on controlled experiments that the software engineering community does not know which background variables are the important ones, thus no template of which data to collect exists.

The focus on subjects in software experiments needs more attention, and this is among others what I want to address in my survey by looking at program comprehension experiments. In my survey I wanted to find out which data has been collected in program comprehension studies and how the data is used in the analysis results. To be able to do adequate analysis, meta-analysis and replications when studying program comprehension the subjects' background need to be thoroughly documented.

The motivation for this research is to help researchers collect the most relevant background information from the subjects in software experiments on program comprehension, and make them focus more on the subjects' individual background when performing the experiment analysis. If a framework or a standard questionnaire was accepted by the software community it would improve their analysis results and reporting, and meta-analysis and replications would be easier to conduct. I will focus my work on experiments and articles related to program comprehension.

1.2 Objective

The objective of this research was to explore what background information has been collected and reported in controlled experiments studying program comprehension, and how this information was used when the experiment results was analyzed. The survey of the articles reporting such experiments in this research will address the following questions:

- What background information is collected and reported from participants in software experiments studying program comprehension?
- How the subjects' background data were used in the result analysis?

I have developed a background questionnaire on the basis of the findings in my survey. The questionnaire was used in a controlled software experiment with professional developers. This survey and questionnaire could be a step towards developing a standard questionnaire for collecting background information from the subjects participating in program comprehension experiments.

1.3 Research context

This master thesis is a part of the project: “Research Methods and Support Tools for Conducting Empirical Research in Software Engineering Document Actions”.

The purpose of the project is to advance the state-of-the art of empirical software engineering research. The research problem to be addressed is how to develop infrastructures, apparatus and methods for conducting experiments and other empirical studies in software engineering that will significantly advance the state of the art.

My work related to this project was to come with suggestions of a background questionnaire that should be used in the experiment.

1.4 Structure

The document is further organized as follows:

Chapter 2 – Related work: Describes the related work.

Chapter 3 – Research method: Describes the research method.

Chapter 4 – Result and discussion: Gives a detailed description of the result in the survey, and relevant discussion.

Chapter 5 – Pre-test: Presents suggestions for a background questionnaire and an experiment where a background questionnaire was used.

Chapter 6 – Validity: Discusses the most important threats to validity of this survey.

Chapter 7 – Conclusion and future work: Presents the conclusions of this thesis, and suggests implications for future work.

Bibliography

Appendix A – Background questionnaire

2 Related Work

This chapter presents related work with focus on subjects in software engineering experiments and their background information. Chapter 2.1 describes the identification of related work, and chapter 2.2 describes the related work found.

2.1 Identification of related work

I have conducted searches in digital databases and libraries to related work. The libraries that have been searched include the ACM Digital Library, INSPEC, IEEE Xplore, and publications and technical reports published at Simula Research Laboratory. The search engine Google has also been used. Keywords used in the search were:

- participants background
- subjects background
- background information
- background questionnaire

The search was performed in February 2005. The initial search gave 3079 hits. To narrow the search even more the sub-keyword "information" were used on the first two keywords, and "subjects" and "participants" on the two last keywords. This resulted in a new hit rate of 1193 articles. Only article titles, keywords and the abstract chapter were used to find articles with relevance. If the title and abstract had relevant information the article was studied more closely. The reference list from articles found and primary studies were also used to find relevant articles.

2.2 Surveys and articles

The only research I know that has been done with the focus on subjects in controlled software experiments is Sjøberg et al. (2004). Hansen (2004) has written his thesis based on the same survey.

Sjøberg et al. (2004) made the survey with an attempt to systematize all controlled experiments reported in leading software engineering journals and conferences in the decade 1993 - 2002. 113 articles of 5453 reported controlled experiment, and were used in the study. In the survey they analyzed the experiments in detail giving an overview on how controlled experiments are reported, and with the focus on subjects participating in the experiments. Sjøberg et al. (2004) concludes from the survey that:

"*There is no generally accepted set of background variables for guiding data collection in a given type of study, simply because the software engineering community does not know which variables are the important ones.*" (Sjøberg, Hannay et al. 2004)(Page 13)

Sjøberg et al. (2004) focused in their survey on subject background variables like gender, age, education, experience and task-related training in the articles, which also are variables related to my survey. The information and level of detail reported about the subjects in these articles varied a lot. 14 of the 113 articles did not report anything about the subjects at all. The background information reported in the controlled experiments conducted in the articles was different if the participants were experts/professionals or students/novices. For the students (91 experiments) the following information was given: gender, age, grades, programming experience, work experience in industry, task-related experience and task-related training. For the professionals (27 experiments) more details were given: reviewers, analysts, programmers, managers, degree, gender, age, language, nationality, programming experience/language, work experience, task-related experience and task-related training. Sjøberg et al. (2004) concludes that the reporting is relatively low and arbitrary, and that this is a hindrance for meta-studies.

Sjøberg et al. (2004) suggests that researchers should collect background information about competence, productivity, education, experience (including domains), task-related training and experience, age, gender, culture, etc., but all depending on what to study and if the subjects are students or professionals. They also suggest that future work should research on the variation in performance related to the subjects' background.

3 Research Method

This section describes how this survey was conducted, the kind of experiments considered in this survey, and the procedure for identifying and analyzing the relevant articles. Chapter 3.1 describes the selection of articles in this survey and chapter 3.2 describes the analysis used.

3.1 Selection and identification of articles

The scope of this research was limited because of the short time available. Levine (2005) performed a controlled experiment where the main goal was to identify the comprehension strategies and difficulties by novice software programmers while understanding and performing maintenance tasks on a medium sized program. I made a search of articles similar to Levine's identification of related work. The search was performed in digital libraries and databases, such as ACM Digital Library, IEEE Xplore and INSPEC. The search engine Google was also used. The main keywords used in the searches were:

- Software comprehension
- Software maintenance
- Program comprehension
- Object-oriented comprehension
- Comprehension strategies
- Problem solving strategies

The initial search resulted in more than 13000 articles. To find the relevant articles for my survey, a sub-search with the keywords "experiment" and "subjects" was conducted with the result of the initial search as basis. This narrowed the number of articles down to 1569.
The articles title, keywords and abstract were studied, in the same order, to select articles with relevance to the research questions. The total number of articles used in this survey became 24. The search was performed in the spring 2005.

3.2 Analyzing articles

A systematic survey of the articles found was now performed. The data extracted from these articles was:

- what kind of background information that has been collected
- how has it been collected
- has it been used in the analysis part
- how the experiment subjects were classified into novice/student, intermediate or expert/professional.

The procedures for data collection and analysis consisted of several steps, and were performed in relation to the questions above. First, the relevant data about the participants'

background data and category classification was extracted from the various articles and listed in a table (see Table 1 and Table 2 on page 20 and 22). I had to make some adjustments about the way information variables was reported when listing them in the table due to that similar kind of data was reported in different ways. The different background data from the different articles was now merged, and the data was analyzed and discussed to reveal what background information is needed to be collected in software experiments studying program comprehension. Secondly, information about how the background information is collected and how the participants were categorized into expertise were analyzed. Finally the results presented in the articles were analyzed with the purpose of finding out how the subjects' background was used in the analysis.

Some of the articles used in this survey provided just the main findings of the experiments, so more data might be given in the full reports.

4 Results and discussion

In this chapter I will present the results of this survey and discuss the information found related to the research questions. Chapter 4.1 gives an overview of the results from the survey. Chapter 4.2 discusses the results.

4.1 Results

In this chapter relevant information from the different articles used in this survey will be summarized. A total of 24 articles were used in the survey.

The different experiments reported in articles used in this survey had all collected background information about the participants. A questionnaire was presented to the participants in some articles, but most of them reported background information without telling how the data was collected. The amount of data reported in these articles differed a lot. The results are presented in Table 1 and Table 2 shown below. The background variables in the tables are based on variables found in the articles. I had to make some generalizations of the background variables because the authors reported one and the same thing in different ways.

		Articles used in the survey									
	Background information variables	(Corritore and Wiedenbeck 2000)	(Corritore and Wiedenbeck 2001)	(Mayrhauser and Vans 1996)	(Koenemann and Robertson 1991)	(Mayrhauser and Vans 1997)	(Jørgensen and Sjøberg 2002)	(Mayrhauser, Vans et al. 1998)	(Burkhardt, Détienne et al. 1998)	(Davis 2000)	(Fix, Wiedenbeck et al. 1993)
Category	Category (Novice, Student, Intermediate, Expert, Professional)	P	E	P	E	P		E	EN	EN	EN
	Sub-category S / N									X	X
	Sub-category P / E	X	X			X	X				
Demo-graphic	Number of participants	30	30	1	12	4	54	5	49	32	40
	Gender		X	X							
	Age (range, average, etc.)		X								
	Mandatory or volunteer										X
	Subjects paid or rewarded				X						
Education	Years of education									X	
	Education degree	X	X							X	
	Recruited from										
	Line of education				X				X	X	
	Number of credits/courses in higher education										

	Background information variables	Articles used in the survey									
		(Corritore and Wiedenbeck 2000)	(Corritore and Wiedenbeck 2001)	(Mayrhauser and Vans 1996)	(Koenemann and Robertson 1991)	(Mayrhauser and Vans 1997)	(Jørgensen and Sjøberg 2002)	(Mayrhauser, Vans et al. 1998)	(Burkhardt, Détienne et al. 1998)	(Davis 2000)	(Fix, Wiedenbeck et al. 1993)
	Number of credits/courses in task relevant programming language										
	Number of credits/courses in programming total										
	Number of courses (no credits)										
	Relevant course type taken									X	X
Experience	Working title/position/area	X	X				X				
	Work experience (yes/no)				X	X					
	Years of work experience	X	X	X	X		X			X	X
	Work experience in a task relevant area			X	X	X		X	X	X	
	Programming experience	X	X		X						
	Knowledge/experience in OO programming		X								
	Number of programming languages familiar with		X		X						
	Number of programming languages (can write simple programs)		X		X						
	Number of programs written										
	Lines of code										
	Specific tool knowledge										
	Knowledge or no knowledge of the program to maintain					X		X			
	Knowledge or no knowledge of the task environment					X		X			
	Platform knowledge		X			X					
	Operating system knowledge		X			X					
	Referred to another article with more background information of the subjects										
	Used background information in the analysis										

Table 1 – Background information with professionals and mixed group of subjects

	Background information variables	Articles used in the survey													
		(Tegarden and Sheetz 2001)	(Levine 2005)	(Hinkel 2005)	(Ramalingam and Wiedenbeck 1999)	(Wiedenbeck and Engebretson 2002)	(Wiedenbeck and Engebretson 2004)	(Hendrix, Cross et al. 2000)	(Prechelt, Unger-Lamprecht et al. 2002)	(Binkley 2002)	(Verth, Bakalik et al. 1989)	(Parkin 2004)	(Karahasanović, Hinkel et al. 2004)	(Ramalingam and Wiedenbeck 1997)	(Mosemann and Wiedenbeck 2001)
Category	Category (Novice, Student, Intermediate, Expert, Professional)	S	S	S	S	N		S	S	S	NS	S	S	NS	NS
	Sub-category: Student or Novice				X	X	X	X	X	X				X	
	Sub-category: Professional or Expert						X								
Demo-graphic	Number of participants	6	39	39	41	32	10	39	74	63	12	29	39	75	101
	Gender		X	X									X	X	X
	Age (range, average, etc.)		X	X	X	X	X						X	X	X
	Mandatory or volunteer		X	X	X			X					X	X	X
	Subjects paid or rewarded		X	X	X			X			X	X	X	X	
Education	Years of education		X	X	X					X		X	X	X	X
	Education degree	X							X			X		X	
	Recruited from		X	X									X		
	Line of education	X	X	X	X			X		X		X	X		
	Number of credits/courses in higher education			X											
	Number of credits/courses in task relevant programming language			X										X	
	Number of credits/courses in programming total													X	
	Number of courses (no credits)														
	Relevant course type taken	X	X	X		X	X	X	X	X	X		X	X	X
Experience	Working title/position/area					X	X								
	Work experience (yes/no)	X					X					X			
	Years of work experience											X			
	Work experience in a task relevant area		X	X									X		
	Programming experience								X			X			
	Knowledge/experience in OO programming		X		X				X				X	X	
	Number of programming languages familiar with				X		X		X					X	
	Number of programming languages (can write simple programs)														
	Number of programs written				X									X	X
	Lines of code				X		X		X					X	X
	Specific tool knowledge			X		X									
	Knowledge or no knowledge of the program to maintain											X			
	Knowledge or no knowledge of the task environment											X			
	Platform knowledge														
	Operating system knowledge														

	Background information variables	Articles used in the survey													
		(Tegarden and Sheetz 2001)	(Levine 2005)	(Hinkel 2005)	(Ramalingam and Wiedenbeck 1999)	(Wiedenbeck and Engebretson 2002)	(Wiedenbeck and Engebretson 2004)	(Hendrix, Cross et al. 2000)	(Prechelt, Unger-Lamprecht et al. 2002)	(Binkley 2002)	(Verth, Bakalik et al. 1989)	(Parkin 2004)	(Karahasanović, Hinkel et al. 2004)	(Ramalingam and Wiedenbeck 1997)	(Mosemann and Wiedenbeck 2001)
	Referred to another article with more background information of the subjects		X										X		
	Used background information in the analysis														

Table 2 – Background information with students and novices as subjects

Figure 1 below shows statistic about the number of background variables reported from the 24 articles in this survey.

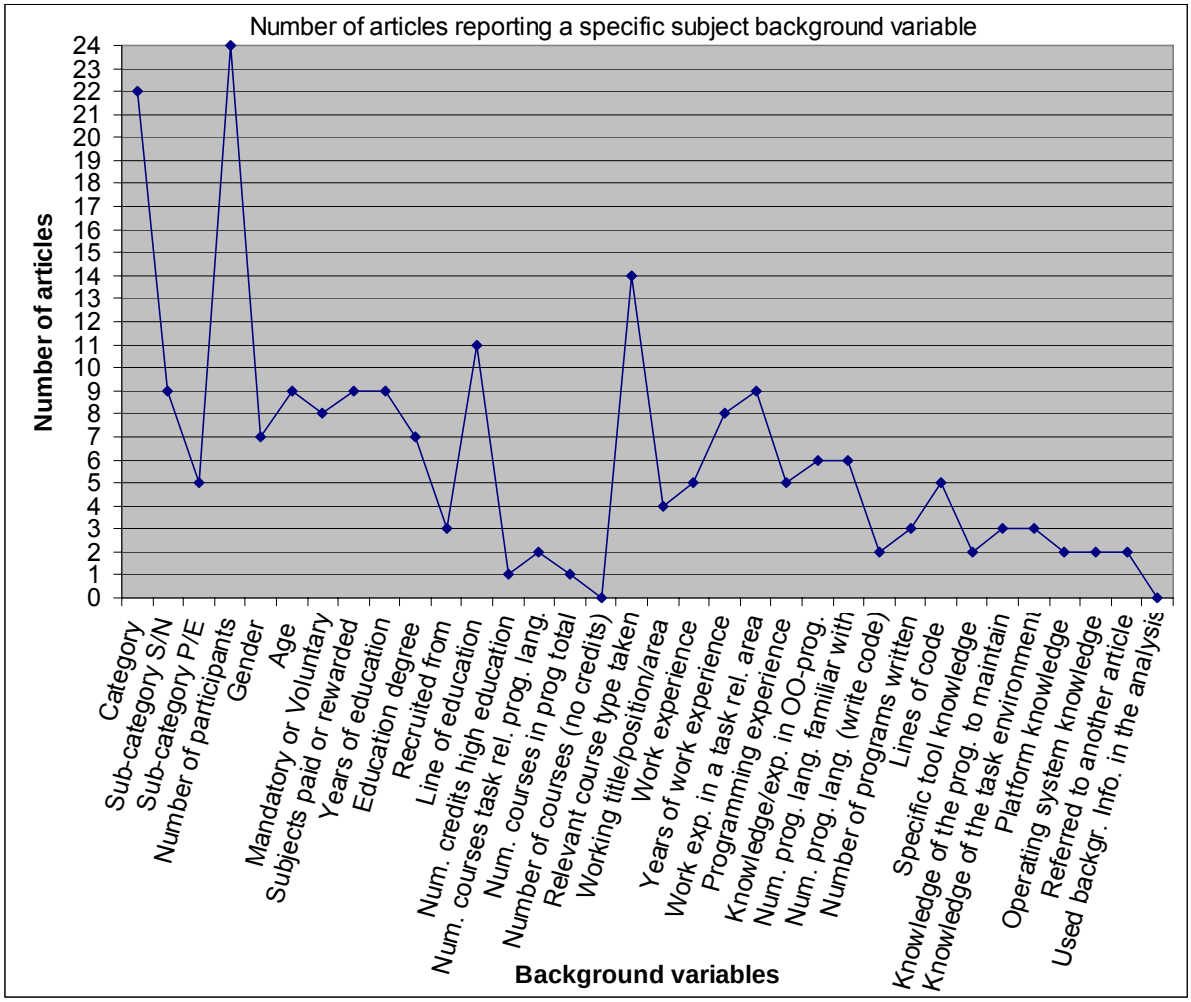

Figure 1 – Graph showing number of articles reporting background variables

4.1.1 Categorization

All articles had made a categorization of the participants, except two where one article called the participants just teachers (Wiedenbeck and Engebretson 2004), and in another the participants were called software maintainers (Jørgensen and Sjøberg 2002). In Mayrhauser et al. (1997) the participants were ranked by levels of expertise and the amount of accumulated knowledge subjects had acquired prior to the start of each observation. Four subjects attended this experiment. Overall the participants were classified as either students/novices or professionals/experts. None of the articles had classified participants as intermediate. The different categories are grouped in two tables shown above. Table 1 shows experiments with just professionals/experts and experiments with a mixed group of both professionals/experts and novices/students, and Table 2 shows experiments with only novices and students. 14 articles involved just students or novice (included Wiedenbeck et al. 2004), seven articles involved experts or professionals (included Jørgensen et al. 2002), and three articles had a mixture of both groups. Seven of the 14 articles with students/novice, four of the seven articles with experts/professionals and two of three articles with mixed subjects had also specified the participants into sub-categories. By sub-categories here I mean such as undergraduate, freshmen, graduate, bachelor, master/MSc, PhD, year of study, etc. for students/novices (S/N), and working title, field of expertise, working position etc. for professionals/experts (P/E). How the different subjects have been categorized into the three main categories (student/novice, intermediate and professional/expert) is not mentioned in any article. Novices are usually students, but in Wiedenbeck (2002) the participants are school teachers, school administrators and teaching assistants, and in Wiedenbeck (2004) the novices are teachers. They are all categorized as novice in the experiments' task area.

4.1.2 Demographic data

The demographic variables mentioned in the articles were only gender and age. Gender was mentioned in seven of the articles, and age in nine. The age was reported as range and/or average. All articles had also reported the number of participants in their experiments where the number of subjects ranged from one up to 101.

4.1.3 Mandatory ness and rewarding

If subjects were participating as volunteers or if the tasks were mandatory where given in eight articles. Tasks in a specific course were seen as mandatory even tough it wasn't mentioned in the report. Six of these articles were also reporting that subjects were paid or rewarded for their participating. In addition to these six, another three also reported about payment and reward. Only one of the experiments with professionals/experts reported about this.

4.1.4 Education

Information about the subjects education or if they where taking a task relevant course where given in 20 of the 24 articles. The four not reporting this were experiments with professionals/experts. Nine reported how many years of education the participants had. This could be information like "third year of education" etc. Seven reported about the education

degrees. Where the participants were recruited from to participate in the experiments where given in only three articles, and these studies were with students as subjects. The information given was the school the subjects studied at. The line of education was given in eleven articles. Some mentioned exactly how many subjects having a specific line of education, but mostly only the different lines of education was mentioned. When it comes to more specific education information little information is given. None of the articles said anything about the grades or grade level of the subjects. Information reported about credits and courses were very low. Only one article gave information about credits/courses in higher education, one informed about the average number of programming courses taken, and two informed about credits/courses taken in the task relevant programming language. 14 of the 24 articles gave information about task relevant courses the participants were taking. Twelve of these were experiments with students. In experiments with both novice and experts (two experiments) this information were just given about the students. For the professionals only information that they had experience was given.

4.1.5 Experience

Work experience information was mostly given in experiments involving professionals. Four articles reported some kind of working title, and five articles mentioned that the subjects have had some work experience, but not in which area or in what position. The years of work experience was given in eight of the articles, and was given as an average and/or range. Work experience in the relevant task area was given in nine reports. A total of 17 articles mentioned something about work experience.

All articles involved some kind of programming task, but information about the subjects programming experience and knowledge was rather low. One could argue that work experience is the same as programming experience, but as long as some articles reported some information related to programming experience I chose to have this as a single variable. Only five reported that the subjects have had programming experience, one of them involving students. This information was given as average and/or range number of years. Six articles reported about experience or knowledge about object-oriented programming. How many programming languages the subjects were familiar with was reported in six articles. Two of these said that the subjects could write small programs in these programming languages. The number of programs written was given in three articles, all with students, and the number of lines written was reported in two articles in addition to the previous three articles, i.e. five articles.

Few details were provided about relevant tools, programs and tasks. Only two articles reported that the subjects had knowledge or experience with tools to be used in the experiments, three informed that the subjects had some kind of or no kind of knowledge related to the program and/or task environment in the experiments. Two articles reported about platform and operating system knowledge even if it was not directly related to the experimental tasks.

The amount of background data reported in these articles varies a lot and relatively few details were provided. Two articles referred to another report for more information about the subjects.

The result and analysis of the experiments are of course the most important part of the article. The background information about the subjects in these experiments was not used in the result and analysis part. In Corritore et al. (2000; 2001) the subjects were all experts, but they were grouped into procedural and object-oriented programmers. The purpose of the research was to examine two dimensions of program comprehension and compare the two groups of subjects. Wiedenbeck et al. (1999) made also grouping of procedural and object-oriented programmers, but background information were not used in the analysis. Mayrhauser et al. (1997; 1998) have only four and five subjects in the experiments where all are compared with each other, but the background data is not discussed in the analysis. Davies (2000), Fix et al. (1993) and Burkhardt et al. (1998) have both novice and experts in their experiments. The two groups were compared, but the background data was not used. Parkin (2004) compared subjects' task time with demographic information. He also made t-test statistics for each demographic characteristic which compared corrective and enhancement samples containing subject's values of that characteristic, to discount the influence demographic data later in the experiment. How this test is done is not documented. None of the other articles focus on subjects' background information in their results and analysis.

4.2 Discussion

The amount of the reported background information varies substantially as shown in the tables and figure above, and I will in the following chapter discuss the findings. Chapter 4.2.1 discusses categorization of subjects. In chapter 4.2.2 the demographic data is discussed. In chapter 4.2.3 the subjects experience and education is discussed. Chapter 4.2.4 discusses the usage of background information in analysis. Chapter 4.2.5 summarizes all.

4.2.1 Categorization

The participants in experiments are usually categorized as expert/professional or novice/student. The survey shows that all authors, except one, had categorized their participants as either expert/professional or student/novice. It seems that the category novice is used when the subjects have very little or no knowledge or experience with the tasks in the experiments. Expert/professional is used when the subjects come from the industry with some years of work experience. The term "intermediate" is not mentioned at all, except from Hinkel (2005) where the students were categorized as intermediate and novice programmers. I think this term is convenient to cover the difficult gap between novice and expert, but also when sub-categorizing the subjects.

Over half of the articles had grouped the subjects into sub-groups. These were mostly involving students. Hansen (2004) shows in his thesis that most experiments are performed with students due to that they are more easily accessible than professionals. The experiments with experts had grouped the subjects related to their field of work. Sjøberg et al. (2004) also registered two main categories and several sub-groups in their survey. Intermediate subjects were not mentioned here either. It is hard to do replications and meta-analysis when the subjects are just called e.g. students or experts. One thing making categorization difficult is that all have different level of expertise. Runeson (2003) discovered big differences between freshmen students and undergraduate students in his experiment. Even students in the same course differ in level of expertise, but they can be distinguished on the basis of their grades. Experts from the industry also differ a lot in expertise. Even with the same education and

work experience. A programmer's level of expertise in a domain greatly affects program comprehension. Hansen (2004) mention that the use of "student" and "professional" in many cases should be exchanged with "novice" and "experts" due to the differences internally in these categories.

What kind of category of the population researchers want to conduct their research on seems to be determined before subjects are recruited to the experiment, but how this is done is usually not reported. The exception is experiments with students where the tasks in a course are compulsory, because here no special form of recruitment is necessary. It is interesting to know how the subjects are recruited, but this is not the scope of this survey.

How are the subjects categorized? How should they be categorized? How detailed should one be in categorizing? Should only the researcher do the categorization? These are all very difficult questions. Hærem (2002) developed his own set of criteria for identifying experts, intermediates and novices. This was done in corporation with the respondents, their managers and the corporation's education center, and the expertise was based on the domain from which the experimental task was developed. He also used questionnaires where the subjects evaluated themselves. The subjects were asked to rate their degree of expertise in their domain. This is something that would be valuable to have in a background questionnaire both before and after the experiment.

When asking if a person looks at himself as an expert, intermediate or novice, students would usually say they are novice unless they have several years working experience in the industry. A student is a person going to school, but he isn't necessarily a novice, because many students today have many years of work experience. When it comes to people with long work experience it is in my perception that they all differs in expertise depending on their self confidence. Some people brag about their skills, some don't, but when a person has many years of experience in e.g. a programming language he usually calls himself an expert. Hærem (2002) says that a person see on himself as an expert until he meets an obstacle that he don't know how to handle. He calls this "not analyzable exceptions". Hærem (2002) suggests that both demographic data, nominating and characteristic value should define a person's expertise. To be called an expert a combination of education and work experience both generally and task related should be considered together with self evaluation questions before and after the tasks.

What is the relation between the different categories and expertise? Long experience does not necessary give good expertise. All depends on the task performance and domain area. Maybe one could categorize the different groups into students and professionals, and internally in these groups categorize the subjects as either novice, intermediate or expert depending on education, work experience, task related experience and knowledge, and task performance? It is important to keep the sub-grouping. The different categories need to be standardized because then meta-analysis, replications etc. can be conducted more precisely. Hærem (2002) refers to other articles and agrees that expertise is domain specific, and that general demographic data are poor proxies of expertise. Standard subject categorizations and background questionnaire could ease this categorization of expertise.

Subjects participating in experiments should be grouped such that they have as similar background as possible. Experts with 5-10 years of work experience can have very different background. The background questionnaire could have questions that sorted out and grouped

the subjects into smaller groups depending on programming experience, programming language, line of code etc. As far as I know there is no model for doing this. The results of a background questionnaire could clarify if the subjects were i.e. experts or not. The subjects could also be classified at the end of the experiment depending on their performances.

Hansen (2004) reported that for students there were lots of different subject categories. There is and should be a difference between undergraduate and graduate students when it comes to program comprehension, and therefore is the suggested categorizing of students by Hansen (2004) quite logical. The problem, mentioned by Hansen (2004), is the different school systems in different countries. These could be made as a standard mapping in a standard background questionnaire. The same is with the professional/expert category. The subjects could just give information in which year of study they are and/or how many years of work experience they have, and from this they could be categorized.

In the reports by Hinkel (2005), Levine (2005) and Karahasanović (Karahasanović, Hinkel et al. 2004) the students participating were divided into two groups because some students had industrial experience. Could these be called intermediate? It is obvious that an intermediate have some kind of education and work experience, but the question is where should the boundary between novice, intermediate and expert be? Many might say this have to be different for the respective experiments. A standard background questionnaire could have questions that made it easier to put subjects into the "correct" categories related to education, general work experience and specific work experience relevant for the tasks conducted.

4.2.2 Mandatory or volunteer

Just a third of the articles reports if the subjects' partitioning is volunteer or mandatory. Volunteers could bias the experiment results by being extra motivated for the tasks and particularly interested in the topic. These subjects would not be representative for the population. The subjects in an experiment should therefore be all either volunteer or mandatory. Each individual participant's motivation could also have influence on the performance and thereby also the results. An expert with very bad motivation would maybe perform as a novice in some cases, but also vice versa. Participants are just normal people and will as everybody else have good or bad days. A background questionnaire could document issues related to this by making the subjects mark of if they volunteer or not, if they have special interests in the task domain area, etc. It is not possible to draw any inference from the results of an experiment if the population is not well defined (Sjøberg, Hannay et al. 2004). A background questionnaire could be used to get a well-defined population. Researchers could make several people fill in the questionnaire ahead of the experiment, and from the results of the answers given the researchers could pick out subjects with the most similar background. The results from the questionnaire could also group the subjects after the experiment if a mixed group of subjects were conducting the experiment. The only thing here is the problem recruiting subjects. I don't think researchers have the luxury to pick out the subjects they want. Subjects are not that easy to get.

Sjøberg et al. (2004) says that volunteers may bias the experiment results because they are more motivated. Experiments that are mandatory to the subjects can make subjects sabotage the experiments by answering wrong answers to questions. This is always a risk, but by rewarding the subjects, this risk is limited. This issue may apply to the background questionnaire also if the subjects brag about their skills. I don't think neither of these issues is

a big problem, because the subjects are usually serious and have themselves interests in getting better program comprehension and software engineering skills.

4.2.3 Rewarding

If the subjects are paid or rewarded for their participation it might have influence on their performance. Nine of the articles in this survey reported about some kind of reward or payment. It is of my perception that subjects are rewarded as long as the tasks are not mandatory. Money or other kind of reward as motivation for the subjects making them take the experiment serious is often used. For compulsory tasks the performance might not be representative unless the subjects gain something from it.

4.2.4 Demographic data

The demographic data reported in the articles were information like number of participants, age and gender. This is data that is easy to collect and report. All reported the number of participants, but under half of the articles reported anything about other demographic data. Sjøberg et al. (2004) also reports that the number of participants is reported in all the 113 articles in their survey, but here also information about demographic data is very shallow. Gender and age was reported, respectively, in only seven and six articles of 91 experiments with students, and two and three of 27 experiments with professionals (Sjøberg, Hannay et al. 2004). The demographic data, except the number of participants, seems not important in the experiments. I think the age will have a certain value when evaluating program comprehension together with the subjects' education and work experience. It is not obvious that novice or students are people between 18 and 25 of age. It is not uncommon today that people that have been working for many years go back to school to either take more education to update his/hers knowledge, or to start all over with something new. Do the subjects with higher age perform better when having the same education and work experience, or vice versa? Gender is also interesting when it comes to comparing program comprehension. Is the program comprehension different between male and female? Demographic data is also always interesting when it comes to statistical data. Thus I mean these data should be in a background questionnaire, and I agree with Sjøberg et al. (2004) that these data also should be reported to make meta-analysis and replications easier.

4.2.5 Education and experience

Some kind of information about the participants' education was not given in only four of the articles. All these four articles had professionals/experts as subjects in their experiment. Information about relevant courses and line of education was most frequently reported. The other variables found in the survey were not reported in many of the articles. Only three articles did not report anything about experience.

What is experience? How can it be measured? Jørgensen and Sjøberg (2002) could not find any guidelines on how to measure or interpret experience. The author refers to a dictionary when saying that it is two valid interpretations of experience: (a) "*event or activity that effects on in some way"*; (b) "*knowledge or skill acquired from seeing and doing things*". The subjects in their research used both interpretations individually and sometime as a combination. Experience is usually measured in years with a specific field of work, but the

subjects in Jørgensen and Sjøberg (2002) stated: "*two maintainers experiencing similar events and activities could reach the same experience level at very different point in time.*" (Jørgensen and Sjøberg 2002)(Page 126). This is a very important observation, and indicates that both the number of years and level of skill must be considered when determining experience.

This also reflects to students taking courses. Here the course grades are a measure of their knowledge or "experience". Experience might be measured in productivity? How many lines of code (LOC) has the subject made lately, total and in a specific programming language? This could be an estimation of the experience together with number of years of relevant activity or event. If two subjects had both e.g. three years of work experience in Java where one of them had worked with Java in the last three years and the other had work with Java four years ago, the subject with most recently work experience probably would perform best. The questionnaire should therefore ask about the total year of experience, and years of experience with specific field areas and when, together with LOC in different programming languages.

One can ask what relevance does education have on task performance when a subject has at least 10 years of experience, but to understand program comprehension this may be important. It also may be important when performing meta-analysis and replication of experiments.

Both education and work experience in the background questionnaire is necessary when having both students and professionals in experiments because, as mentioned earlier, it is not uncommon today to go back to school after several years of work experience.

4.2.6 Usage of background information in analysis

How is the background information referred to in the analysis and result? How does the results relate to the classification and comprehension? Is it possible to use the background information to better understand program comprehension? How? How realistic is the experiments when the background information reported in articles are relatively low and arbitrary?

Most of the articles report some information about the subjects and categorizes them, but this is rarely used in the analysis. This might be due to that it is out of the scope for the research objective. It is logical that researchers report just the information relevant for the scope of the experiment, but to be able to make meta-analysis and replications more details of the subjects' background is necessary. It is in my perception that the researchers have collected more background information than they report, and hopefully the background data is collected and stored for future analysis. This is an important issue a standard background questionnaire could make easier together with a uniform way of reporting experiments (Sjøberg, Hannay et al. 2004).

"*The heterogeneity of the subjects is generally not paid much attention to in the papers analysed. Most of them do not seem to focus on the diversity in subject backgrounds, and only a few of them report on differences between the individual subjects or between categories of subjects.*" (Hansen 2004)(Page 46). This is also findings I find in the article in my survey. Subjects individual background was not used in the analysis at all. In the

experiments by Mayrhauser and Vans (1997) and Mayrhauser, Vans et al. (1998) it was only four and five subjects respectively where the reader could study each subject in the analysis, but the background data was not directly used in the analysis. The number of subjects in the different experiments might be one reason why diversity among subjects is not paid much attention. This is of course a very large and difficult job, but I must agree with Hansen (2004) when he says that it must be paid more attention to who the subjects in experiments are. Not just as groups and categories, but also as individuals. This could be solved with a standard background questionnaire and a uniform way of reporting the data.

4.2.7 Summary

Sjøberg et al. (2004) recommend that researchers should report the following regarding subjects' background: the type and number of subjects, context variables such a general software engineering experience and experience specific to the tasks in the experiments, and how the subjects were recruited.

The background information collected can of course vary depending on the type of experiments, but in program comprehension experiments this information is vital and important to the results and analysis. Some background information should be mandatory to collect and report, but specific background information relevant for the specific experiment should also be collected and reported.

My study show that the amount of background reported is very arbitrary, but it is in my perception that more background information is collected than reported in these articles. Because of confidentiality agreements some of the background data might not be possible to publish. One could also do as Karahasanović (Karahasanović, Hinkel et al. 2004) and Levine (2005) by referring to a paper with more background information. It is then up to the reviewer to get this data if interested.

All the different background variables reported in the articles in this survey are very important to conduct meta-analysis, replications and research on the variations in performance related to subjects' background.

5 Background Questionnaire

A controlled experiment on program comprehension was performed at Simula Research Laboratory in May 2005. The background questionnaire I have developed was used in this experiment. This section proposes a background questionnaire, describes the experiment and my experience with the questionnaire.

5.1 Background questionnaire

Below I come with suggestions of variables that I mean should be in a background questionnaire for software experiments studying program comprehension. The variables are based on my survey and my own thoughts.

- Demographic data (age, gender, number of participants)
- Education (where, when, degree, credits, grades in a task specific course)
- Task relevant courses (when, credits, grades)
- Work experience (when, with what, num. of projects, project size, etc.)
- Work position/function (when)
- Programming knowledge (language, LOC, task specific knowledge)
- Design and patterns experience
- Task relevant experience or/and training
- Tool experience
- Self evaluating of expertise in related things
- Mandatory or volunteer?
- Paid/rewarded?
- Area of special interest
- Motivation degree (range)

Based on this list I developed a proposal for a background information questionnaire that was used in an experiment explained in chapter 5.2. The proposed background questionnaire is presented in Appendix A.

The questionnaire to be used on students and professionals does not need to be much different. Both should have mostly the same questions, but maybe in more detail about courses taken, course recruited from, credits and grades for students. For experts with many years of experience grades and credits are maybe not that relevant, but the courses taken and at which university/collage is interesting. Work experience, self evaluation of expertise and task relevant experience and knowledge should be described thoroughly by all kind of subjects. The same is for programming knowledge. Experts with years of work experience need to estimate more about their productivity than students. I don't think it is necessary to have different questionnaires depending on what categorization the subjects are from. If a question is not applicable for a subject, the subject just need to answer exactly "not applicable". If different questionnaires for students and experts are necessary the background

questionnaire could be automated such that the questions changed on the basis of what was answered.

The different participants in an experiment will always have different perceptions of the experiment, but also on the background questionnaire. Participants might not want to or can not answer all the questions due to personal reasons or that they just do not know the answer. One particular question here is the number of lines of code programmed in a specific programming language. This is a number you might not go around remembering, even though the most used way to measure a programmers productivity is to count lines of code made. In the background questionnaire used in the pre-test the subjects were asked to give an estimated number themselves, but the best would perhaps be to give suggestions where the subjects tick off the best suited answer. This would again simplify the analysis work.

Some participants might have solved similar tasks earlier in their education or work. The background information is usually collected before the experiment, but information if the task was known from before could be collected either right after the task or after the experiment, and taken into consideration when the result is being analyzed.

If subjects have some kind of relevant work experience, the company they come from might not be of any relevance, but the domain area and working area should be given where the subjects do self evaluations of their expertise. Besides these questions and ratings, questionnaires could also be answered by the subjects' manager and the subjects' education center (Hærem 2002). Together with the task results each subject could then be categorized more precisely. The questions and ratings should be as specific to the task they perform as possible. Unfortunately this makes it very difficult to develop a standard background questionnaire.

How is the background questionnaire performed? On paper or electronically. By interview? The different articles do not tell anything about this, and I think it is a big mixture of them all. With a tool like Simula Engineering Supporting Environment (SESE) (Arisholm, Sjøberg et al. 2002), which is a web-based support environment for software engineering experiments, it is easy to make this questionnaire on the web. SESE also generate statistics that is easy to extract and use in the analysis. My background questionnaire was implemented using SESE.

The background information should be as thorough as possible because the data can be used in the analysis, and it would also confirm if the participants in the experiments were classified in the correct category; novice, intermediate or expert. When researchers recruit participants to their controlled experiments they usually try to find a homogenous group of people. The different subjects could be selected out from people's curriculum vitae if intermediate or experts are wanted. If novices/students are wanted for the experiment the researchers contact schools and universities. Novices/students are always easiest to find because they are at school, maybe attending some relevant course. They are also easiest to get because they are cheap and usually have time to participate in experiments. Experts and intermediate are usually in a job situation, and cost a lot because they must be taken out of their regular work when performing the experiment. Therefore most experiments are performed with students.

The programming skills subjects have must be collected. This can be collected by a rating scale where the subjects tick of. The subjects need to be objective about their own perception and expertise regarding their programming skills, and not think what other people say about them.

The background questionnaire I made has text fields where the subjects should give a number as answer. The reason a text field was used is because we encouraged subjects to give a more detailed answer. For example that they told us the time period they programmed using a specific programming language, how big the project was, etc. This could of course have been divided into several more detailed questions.

Making a standardized background questionnaire is not an easy task due to the categorization problem of the participations in experiments, disagreements between researchers, different kind of research areas etc. Researchers, students, experts and scientists need to agree on the classification and what kind of information that should be collected and reported.

5.2 The experiment

In this chapter I will tell about the experiment where my proposed background questionnaire was used.

5.2.1 Data collection and supporting tools

SESE is a web-based tool supporting logistics in controlled software experiments (Arisholm, Sjøberg et al. 2002). The participants used this tool to answer the background questionnaire, to download documentations and code, to up-load their task solutions and to give feedback during solving the tasks (feedback-collection). The tool recorded start-time and end-time for each task. Keystrokes, mouse-clicks and window focus events were logged with timestamps in milliseconds by the GRUMPS-Lite software (Thomas and Kennedy 2003).

5.2.2 Participants

A request to some Norwegian industry companies, having Java developers, was performed to recruit subjects to the experiment. They participant were paid for their time. The subjects in the pre-test were 24 professionals from the Norwegian industry who conducted change tasks on Java applications. All of them had good knowledge of Java. The experiment was conducted in five separate sessions on five separate days, and with maximum eight subjects each day.

5.2.3 The treatments and tasks

The experiment was conducted at Simula Research Laboratory's facilities at IT-Fornebu where all the participants were given a PC-terminal.

The background questionnaire and the experiment were two separate treatments in SESE, where the experiment depended on the background questionnaire, meaning that it had to be filled in before the experiment could be started. The background questionnaire is a fairly

straight forward scheme to fill in and is given in Appendix A. The subjects spent about 15 minutes on this.

The participants of the pre-test was introduced the background questionnaire and to unknown applications written in Java. The applications was a Mini-bank (Arisholm and Sjøberg 2001) and a Library application system (Ericsson and Penker 1998) where the subjects added new functionalities. When solving tasks on the Library application a feed-back collection screen (Karahasanović and Sjøberg 2001) appeared every 15 minutes with the text: "What are you thinking now?" The subjects were instructed to describe what they were thinking just before the screen appeared. The time available for writing comments was limited to two minutes.

Online documentation (the Library application system description, tasks, Java documentation) was provided. The subjects were not allowed talking to each other about the tasks during or after the experiments. Confidentiality had to be accepted before the tasks could be downloaded.

5.3 Experience with the questionnaire

The subjects in the experiment came with some questions because of uncertainty on some questions. Se appendix A for the questions related to the different issues.

The biggest problem was related to the number of lines of code (LOC) implemented in the specific programming languages. This was something that subjects had problems to remember. They also asked if they should count in LOC programmed in school too. My intention of the question about the LOC in different programming languages was to document all coding, so the LOC from school should be given. This can of course be discussed. The subjects should write down the number of LOC, but as suggested earlier, the best would probably be if they had suggestions to tick of.

Another topic was if programming in writing their MSc. thesis should count as work experience. This is a difficult question and it is two approaches to this such as I see it. The thesis is a part of the education and would therefore be a double up if also used as work experience. However, a thesis can be done in a company and could therefore count as work experience. This question should be specified and standardized in the future. My suggestion is to have this as part of the education, but that the thesis could be specified in another question.

At the end of the questionnaire there are some questions about OO-projects. The subjects asked if they should count in projects from school also. My intention of these questions is that only work related projects should be given because the OO-projects in school are usually not full projects due to the time available in a semester.

The general experience with the questions is that they are not clear enough in their formulations. All questions could be more precise to avoid misunderstandings, and should be evaluated more in the future.

6 Validity

The main threats to validity for this study were article limited scope of the survey, selection bias, inaccuracy in data extraction and misclassification.

This research has been conducted over a short time of period, and a limited number of articles were analyzed. It is therefore difficult to generalize the results of this study. However it can be a pointing pin for further research on this area where a larger amount of articles involving experiments should be used as a basis. The articles were selected on the basis of analysis of titles, keywords and abstracts; hence the selection could have been more thorough.

Extracting data from papers is a non-trivial task, and the lack of common terminology complicates this even more. Data that may seem obvious to me may thus be misinterpreted. Some data have not been stated explicitly enough, making approximations and best educated guesses necessary, while other data have been extracted between the lines. My goal has been to analyze articles in an objective manner, but I have been forced to use more or less subjective opinions on several occasions. I have tried to meet this challenge by reading the articles several times.

The lack of a common terminology imposes a threat to validity also regarding classification of data. I know that the terminology is applied differently by different authors, thus making it easy to categorize data wrong. Classification is difficult when there are no standardized labels to sort elements into. I have in many cases had to come up with my own variables based on the type of information contained in the analyzed papers. As this information is not standardized, I have extracted and classified it according to our own interpretations of what is written in clear text and between the lines.

Although I'm aware that there are threats to the validity of this survey, I feel that I have addressed these issues, and more important - taken actions to minimize them. I have had to use subjective opinions when selecting, extracting and analysing the material, but this has been subject to careful evaluations. Others may disagree with me on singular categorizations, but in the big picture I feel confident that the overall results will remain the same.

7 Conclusions and Future Work

In this chapter I will draw some conclusions from the results of this survey (chapter 7.1) and outline some possibilities for future work (chapter 7.2).

7.1 Conclusions

Computer program maintenance is one of the core engineering activities programmers do, and to make a good maintenance job you need good program comprehension. It is widely accepted that software maintenance absorbs a significant amount of the effort used in software development, and that the major time consuming process in software maintenance is program comprehension. Programmers use different strategies to comprehend programs, and many experiments have been conducted to address this. Research on how programmers comprehend programs is important to be able to improve software tools, documentation, maintenance guidelines and education of programmers that support their cognitive processes in an appropriate manner, and thus improve maintenance and program comprehension. The subjects in the experiments are usually heterogeneous, and the productivity between individual programmers with similar background might vary significantly. So to be able to conduct adequate analysis each individual subjects' background must be taken into consideration. Thus, the purpose of this survey was to find what kind of background information that is reported in published articles on program comprehension experiments. The importance of a survey like this is to make researchers focus more on the subjects' background when conducting experiments on program comprehension. Not just when collecting background information, but also when conducting the analysis of the experiment and when writing the research articles and reports. This survey could also be a step towards a standard background questionnaire and of what to report in the research articles.

If researchers focus too little on the individual subjects' background in their analysis, the results would not necessarily represent all subjects in the experiment because of the individual differences in background and strategies the subjects use. The strategies should be analyzed on the basis of the background. If the subjects background is not used in the analysis it is very difficult for other readers to make a validation of the analysis when little and arbitrary information about the subjects are given. Detailed background information about subjects in program comprehension experiments reported in articles makes it easier to perform meta-analysis and replications. The level of detail might depend on what the research focuses on, but within program comprehension some standard information should be collected. A standard questionnaire could be a step towards a standard or at least a uniform way of reporting subjects' background information, which again would ease the reading, validation and comparing of reports. A standard questionnaire would also ease the work for the researchers when collecting background information.

The different background variables reported in the articles in this survey all together cover most of the variables researchers should collect from the subjects. Many of the articles also reports about relevant courses subjects are taking, which I find very important when looking

at program comprehension. The experiments with professionals report mostly about work experience and the reports with students report mostly about education. Even though many different background variables were reported in the articles, the survey showed that the amount of background data reported in program comprehension studies is rather arbitrary. I can‘t see that any article uses any form of standard when reporting about subjects background, indicating that no standard is used when collecting this background information either. A few articles reports that a questionnaire had been used, but not what kind of variables that had been collected. My perception is that questionnaires about subjects' background are often very simple and have very little information, and when the result of the experiment is analyzed the researchers use very little of the background information collected. Researchers write something about the subjects, but not how it is collected and used in the analysis.

Results show that the background information reported differs between experiments with students and experiments with experts. Much more information about education is given in experiments with students, and much more information about work experience is given in experiments with experts. This is in some way naturally, but the information about e.g. students work experience should not be neglected, and the same is about experts' education. The most frequent variables reported about the subjects in my survey were categorization, line of education and relevant course type taken.

Analysis in the experiments in this survey shows a focus on the programming effort and comprehension. It is very important to look at how well the tasks have been solved, but results must carefully be evaluated in context with the subjects' background and experience. In most of these experiments the participants' background is more or less neglected in the result analysis. The few exceptions is in studies where the number of participants is very small (Mayrhauser and Vans 1996; Mayrhauser and Vans 1997; Mayrhauser, Vans et al. 1998). In those studies it is easier to perform analysis looking at differences between the subjects, but here also the amount of data reported is too little and varies too much. Analyzing data collected from large experiments with many subjects is very work intensive and error prone, and might be a reason why the subjects' background is not taken into consideration in the analysis. Because the subjects' background information reported varies too much from article to article, it makes it difficult to conduct meta-analysis and replications. The research also shows that the same authors differ in background information given in their articles. This indicates that the researchers don't collect the same background information from experiment to experiment, which again indicates that no standard is used.

Many of the articles analyzed in this survey are a summary of the experiments conducted; meaning more data about the subjects in the experiments probably or might exist. These data can probably be given when contacting the authors, but the more data collected and reported; the easier it is for other researchers to perform similar researches, replications and meta-analysis. The analysis depends of course of the specific scope of the research, but the subjects' background is a part of how program comprehension should be analyzed.

Hansen (2004) made a survey on controlled software engineering experiments in the decade 1993–2002 and concludes in his research that the way experiments are reported does not adhere to any standards, which is the same conclusion I had drawn from my survey. The background information reported varies too much. Hansen (2004) reports that the main categorization of information reported was programming experience, work experience, task

experience and task related experience. Sjøberg et al. (2004) comes with recommendations of what to report in articles, and says that a more uniform way of reporting experiments would help to improve the review of articles, replications of experiments, meta-analysis and theory building.

The results from my survey can be used as a template for further research in this area, and my questionnaire can be a template for developing a standard background questionnaire. I would suggest that even more detailed information should be reported about the subjects' background to improve the ability to perform meta-analysis and replications, but also improve research on the variations of comprehension between individual subjects. A standard background questionnaire could help making a standard or uniform way of reporting about subjects participating in software experiments, making it easier to study the varieties between subjects in program comprehension studies, but also in other kind of software experiments.

With a standardized background questionnaire it would be easier to compare different studies researching on program comprehension. Due to the lack of and difference in background information in earlier studies the comparison and meta-analysis can be difficult. Background data from subjects in earlier experiments might be very difficult to collect since researchers will not or don't have the time to dig up such data. This was a problem in the survey made by Hansen (2004) and Sjøberg et al. (2004).

Researchers categorize subjects mainly into two categorizations; students/novice and experts/professionals. Students were used in most of the articles used in this survey. I think it is far too easy just to group the participants into novice and experts when performing analysis. More detailed categorization is needed to perform more detailed studies on each subject's program comprehension. One could make a model for grouping the subjects into smaller groups depending on each subject's expertise, and have main groups like novice, intermediate and expert. Hærem (2002) made a research on expertise, and similar research should be conducted where suggestions for standard categorization could be made.

The ideal questionnaire could be hierarchical where the next question depends on the answer given in the previous question. In this way you would not have several questionnaires and you did not have to care about the background of the subject when assigning them the experiment. The data collected from the background questionnaire could also be included in the analysis tool used to get the result from the experiment itself. This is a major and difficult job, and it might not be possible at all, but this should be investigated further. The analysis should be performed with respect to each individual's background, and then specially education, work experience and task relevant experience.

A survey like this could make the researchers more aware of collecting background information from the subjects and use it in the analysis even though no standard questionnaire existed. How the analysis protocols work is not the scope of this thesis. I have no suggestions of how to integrate the subjects' background data in the analysis, but if the results from a background questionnaire could be a part of an analysis protocol it would ease and improve the analysis.

Researchers might judge me for criticizing the authors of the articles for not describing the background information about the experiment participants thorough enough, and I'm in no

position to criticize either. But the researchers must agree that the documentation about the subjects vary quite a lot and is not of any kind of standard. If a standard questionnaire could be developed it would reduce the "overhead" of collecting and documenting this background information, thus ease the researchers work on this point.

Making a standard background questionnaire is very difficult, and maybe also impossible due to the different experiments and objective in the research. I think a standard questionnaire can be made, but the individual research must of course add questions that are more relevant to each specific research. A standard of what should be reported in research articles could also be made in addition to the standard background questionnaire. Sjøberg et al. (2004) recommend to report accurately the following: "*the type and number of subjects, including the mortality rate; context variables such as general software engineering experience and experience specific to the tasks of the experiments; how the subjects were recruited; the application areas and type of tasks; the duration of the tasks; and external validity of the experiments, including being specific about the sample and target population of the experiment.*" (Sjøberg, Hannay et al. 2004)(Page 31). The combination of these standards would ease the gathering and reporting of subjects' background information, the review of articles, meta-analysis, replication of experiments and theory building.

7.2 Future work

Due to my research time was very short, I had to focus just on a particular group of experiments; program comprehension. The amount of articles read and data collected is therefore rather small. A similar survey should performed in a much larger scale where data from many different types of software experiments was collected and analyzed to make suggestions for a standard background questionnaire. Further research on subjects' categorization of expertise should also be conducted. The suggested questionnaire should be presented to other researchers for evaluation, and should be tested in different experiments.

Bibliography

Arisholm, E. and D. I. K. Sjøberg (2001). "Assessing the Changeability of two Object-Oriented Design Alternatives - a Controlled Experiment." Empirical Software Engineering **6**(3): 231-277.

Arisholm, E., D. I. K. Sjøberg, et al. (2002). "A Web-based Support Environment for Software Engineering Experiments." Nordic Journal of Computing **9**(4): 231-247.

Binkley, D. (2002). "An Empirical Study of the Effect of Semantic Differences on Program Comprehension." Proceedings of the 10th International Workshop on Program Comprehension (IWPC'02).

Brooks, R. (1983). "Towards a Theory of the Comprehension of Computer Programs." International Journal of Man-Machine Studies: 543-554.

Brooks, R. E. (1980). "Studying programmer behavior experimentally: the problems of proper methodology." Communications of the ACM **2**(4): 207-213.

Burkhardt, J. M., F. Détienne, et al. (1998). "The Effect of Object-Oriented Programming Expertise in Several Dimensions of Comprehension Strategies." Proceedings of the 6th International Workshop on Program Comprehension (IWPC+98): 82-89.

Coleman, D., D. Ash, et al. (1994). "Using Metrics to Evaluate Software System Maintainability." IEEE Computer: 44–49.

Corritore, C. L. and S. Wiedenbeck (2000). "Direction and Scope of Comprehension-Related Activities by Procedural and Object-Oriented Programmers: An Empirical Study."

Corritore, C. L. and S. Wiedenbeck (2001). "An exploratory study of program comprehension strategies of procedural and object-oriented programme."

Davis, S. P. (2000). "Expertise and the program comprehension of object-oriented programs." 12th Workshop of the Psychology of Programming Interest Group, Cozenza Italy.

Dunsmore, A., M. Roper, et al. (2000). "Object-oriented inspection in the face of delocalisation." Proceedings of the 22nd international conference on Software engineering.

Ericsson, H.-E. and M. Penker (1998). "Case study. UML Toolkit." John Wiley and Sons, Inc.

Fix, V., S. Wiedenbeck, et al. (1993). "Mental Representation of Programs by Novices and Experts." INTERCHI'93: 74-79.

Hansen, O. (2004). Survey of controlled software engineering with focus on subjects. Department of Informatics. Oslo, University of Oslo.

Hendrix, T. D., J. H. Cross, et al. (2000). "Do Visualization Improve Program Comprehensibility? Experiments With Control Structure Diagrams for Java."

Hinkel, U. N. (2005). Evaluating Methods for Data Collection during Software Engineering Experiments. Department of Informatics. Oslo, University of Oslo.

Holgeid, K. K., J. Krogstie, et al. (2000). "A Study of Development and Maintenance in Norway: Assessing the Efficiency of Information Systems Support Using Functional Maintenance." Information and Software Technology **42**(10): 687–700.

Hærem, T. (2002). "Task Complexity and Expertise as Determinants of Task Perception and Performance: Why Technology-Structure Research has been unreliable and inconclusive." Series of Dissertations.

Jørgensen, M. and D. I. K. Sjøberg (2002). "Impact of experience on maintenance skills." Journal of software maintenance and evolution: Research and practice **14**: 123-146.

Karahasanović, A., U. N. Hinkel, et al. (2004). "Feedback Collection versus Think-Aloud in Software Engineering Research: A Controlled Experiment."

Karahasanović, A. and D. I. K. Sjøberg (2001). "Data Collection in Software Engineering Experiments." Information Resources Management Association International Conference.

Koenemann, J. and S. P. Robertson (1991). "Expert problem solving strategies for program comprehension." Conference on Human Factors in Computing Systems.

Lehman, M. and L. A. Belady (1985). "Program Evolution — Processes of Software Change." Academic Press.

Letovsky, S. (1986). "Cognitive processes in program comprehension." Empirical Studies of programmers: 80-98.

Levine, A. K. (2005). A study of comprehension strategies and difficults by novice programmers performing maintenance tasks of object-oriented systems. Department of Informatics. Oslo, University of Oslo.

Lientz, B. P. (1983). "Issues in Software Maintenance." **15**: 271–278.

Littman, D. C., J. Pinto, et al. (1986). "Mental models and software maintenance." The first workshop on empirical studies of programmers on Empirical studies of programmers.

Lucia, A. D., A. R. Fasolino, et al. (1996). "Understanding function behaviors through program slicing." Proceedings of 4th IEEE Workshop on Program Compre-hension: 9-18.

Mayrhauser, A. v. and A. M. Vans (1995). "Program understanding: models and experiments." Advances in Computers **40**: 1-38.

Mayrhauser, A. v. and A. M. Vans (1996). "Identification of Dynamic Comprehension Processes During Large Scale Maintenance." IEEE Transactions on Software Engineering **22**: 424 - 437.

Mayrhauser, A. v. and A. M. Vans (1997). "Program understanding behaviour during debugging of large scale software." 157-178.

Mayrhauser, A. v., A. M. Vans, et al. (1998). "Program Comprehension and Enhancement of Software."

Mosemann, R. and S. Wiedenbeck (2001). "Navigation and Comprehension of Programs by Novice Programmers." Proceedings of the Ninth International Workshop on Program Comprehension (IWPC'01).

Nosek, J. T. and P. Prashant (1990). "Software Maintenance Management: Change in the Last Decade." Journal of Software Maintenance Research and Practice **2**(3): 157–174.

Parkin, P. (2004). "An Exploratory Study of Code and Document Interactions during Task-directed Program Comprehension." 2004 Australian Software Engineering Conference (ASWEC'04).

Pennington, N. (1987). "Comprehension strategies in programming." Empirical studies of programmers: second workshop.

Pfleeger, S. L. (1987). "Software Engineering — The Production of Quality Software." Macmillan.

Prechelt, L., B. Unger-Lamprecht, et al. (2002). "Two Controlled Experiments Assigning the Usefulness of Design Pattern Documentation in Program Maintenance." IEEE Transaction on Software Engineering **28**(6): 595-606.

Ramalingam, V. and S. Wiedenbeck (1997). "An empirical study of novice program comprehension in the imperative and object-oriented styles."

Ramalingam, V. and S. Wiedenbeck (1999). "Novice comprehension of small programs written in the procedural and object-oriented styles." Int. J. Human-Computer Studies **51**: 71-87.

Runeson, P. (2003). "Using students as Experiment Subjects – An analysis on Graduate and Freshmen Student Data."

Schneiderman, B. and R. Mayer (1979). "Syntactic/semantic interactions in programmer behavior: a model and experimental results." International Journal of Computer and Information Science **8**: 219-238.

Sjøberg, D. I. K., J. Hannay, et al. (2004). "A Survey of Controlled Experiments in Software Engineering."

Soloway, E., K. Ehrlich, et al. (1982). "What do novices know about programming?" Directions in Human-Computer Interaction: 27-54.

Tegarden, D. P. and S. D. Sheetz (2001). "Cognitive activities in OO development." Int. J. Human-Computer Studies **54**: 779-798.

Thomas, R. and G. Kennedy (2003). Generic Usage Monitoring of Programming Students. ASCILITE 2003 Conference, University of Adelaide, Australia.

Verth, P. B. V., L. Bakalik, et al. (1989). "Use of the Cloze Procedure in Testing a Model of Complexity." 156-160.

Wiedenbeck, S. and A. Engebretson (2002). "Novice Comprehension of Programs Using Task-Specific and Non-Task-Specific Constructs." Proceedings of the IEEE 2002 Symposia on Human Centric Computing Languages and Environments (HCC'02).

Wiedenbeck, S. and A. Engebretson (2004). "Comprehension Strategies of End-User Programmers in an Event-Driven Application." Proceedings of the 2004 IEEE Symposium of Visual Languages and Human Centric Computing (VLHCC'04).

Zelkowitz, M. V. (1978). "Perspectives on Software Engineering." ACM Computing Surveys **10**(2): 197–216.

Appendix A: Background Questionnaire

This background questionnaire is in Norwegian due to that the participants in the pre-test were all Norwegians. The questionnaire is not shown in the web-based form with text fields, radio buttons, etc., but in plain text:

Thinkaloud-replikeringseksperiment: Bakgrunnsskjema
Velkommen til Feedback Collection eksperimentet!

Dette eksperimentet er delt inn i to hoveddeler:

- Bakgrunnsinformasjonsskjema
- Feedback Collection eksperimentet

Formålet med bakgrunnsinformasjonsskjemaet er å hente inn informasjon som er relevant for selve analysen av eksperimentet.

Formålet med eksperimentet er å utforske forståelsen av objektorienterte konsepter.
Formålet med eksperimentet er ikke å evaluere hvor flink du er til å programmere.

I dette eksperimentet skal du løse endringsoppgaver på små applikasjoner i Java ved hjelp av JBuilder. For å forstå hvordan du løser oppgavene, må vi "titte" litt i tankene dine. Du vil bli bedt om å skrive hvordan du har tenkt i løpet av eksperimentet ved at du noterer ned tankene dine inn i et eget vindu, som vil dukke opp med jevne mellomrom. Dette kommer vi nærmere inn på senere.

Selve eksperimentet består av fem deler:

1. En øvelsesoppgave.
2. En oppgave som du skal gjennomføre på vanlig måte.
3. Oppvarmingsøvelser hvor du skal øve på å bruke Feedback-collection-vinduet sammen med en instruktør.
4. Tre oppgaver til. Du skal skrive kommentarene i Feedback-collection-vinduet mens du jobber.
5. Et spørsmålsskjema.

Før du starter på eksperimentet må du fylle ut informasjon om din bakgrunn.

Innledning bakgrunnsskjema

I dette spørreskjemaet skal du svare på spørsmål og påstander om din erfaring, utdanning og kompetanse.

Det er viktig at du svarer ut fra din egen mening og ikke ut fra hva andre måtte mene, eller det du tror er "riktig" svar. Det er ingen riktige eller gale svar. Det riktige svaret er det du selv mener passer best.

Ikke tenk for mye på hver påstand, men velg det som virker umiddelbart riktig. Det første innfallet er oftest det man egentlig mener.

Opplysningene vil utelukkende bli brukt som en del av eksperimentet, og vil bli behandlet konfidensielt.

Dette skjemaet må være ferdig utfylt før du kan begynne med selve eksperimentet.
Spørsmål merket med * må besvares!

Erfaringsskjema

3.1) **Personinformasjon**
3.1.1) *Fødselsår
Angi hvilket år du er født. (åååå)

3.1.2) *Kjønn

Mann Kvinne

3.2) **Arbeidserfaring**
3.2.1) *Hvor mange års arbeidserfaring med programmering/systemutvikling har du ?
Beskriv det gjerne utfyllende ved å ta med hvilket språk og når.

3.2.2) *Hvor mange års arbeidserfaring har du totalt?
Angi med ca. desimaltall.
Beskriv også gjerne hvilken stilling og årsperiode.

3.3) **Utdanning**
3.3.1) *Hvor mange vekttall har du totalt (20 vekttall = 1 år fulltids utdanning; 1 vekttall = 3 studiepoeng)?

3.3.2) *Hvor mange vekttall programmeringsrelatert informatikk har du?

3.4) **Kurs**
Kurs du har tatt privat eller interne/eksterne kurs gjennom jobben.

3.4.1) *Angi hvor mange datarelaterte kurs du har tatt (som ikke har gitt vekttall).
Husker du ikke eksakt så angi et cirka tall. Antallet trenger strengt tatt ikke å stemme med antall kurs du beskriver i neste spørsmål.

3.4.2) *Beskriv kurs du har tatt innenfor programmeringsrelatert informatikk og systemutvikling, og når (årstall).
Ikke ta med fag/kurs fra skoler som gir vekttall. Dette skal du beskrive senere.
Beskriv hva kurset gikk ut på, når og om det ble tatt internt/eksternt på jobben eller på privat basis.

Programmeringskompetanse

4.1) Generell programmeringskompetanse
4.1.1) *Hva er din vurdering av hvor dyktig du er som programmerer?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

4.2) Spesifikk programmeringskompetanse - Java
4.2.1) *Hvor mange måneders arbeidserfaring har du totalt med dette programmeringsspråket?

4.2.2) *Estimer omtrentlig hvor mange linjer kode du har programmert i dette språket:

4.2.3) *Hva er din vurdering av hvor godt du kan dette programmeringsspråket?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

4.3) Spesifikk programmeringskompetanse - C++
4.3.1) *Hvor mange måneders arbeidserfaring har du totalt med dette programmeringsspråket?

4.3.2) *Estimer omtrentlig hvor mange linjer kode du har programmert i dette språket:

4.3.3) *Hva er din vurdering av hvor godt du kan dette programmeringsspråket?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

4.4) Spesifikk programmeringskompetanse - C#
4.4.1) *Hvor mange måneders arbeidserfaring har du totalt med dette programmeringsspråket?

4.4.2) *Estimer omtrentlig hvor mange linjer kode du har programmert i dette språket:

4.4.3) *Hva er din vurdering av hvor godt du kan dette programmeringsspråket?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

4.5) Spesifikk programmeringskompetanse - Simula
4.5.1) *Hvor mange måneders arbeidserfaring har du totalt med dette programmeringsspråket?

4.5.2) *Estimer omtrentlig hvor mange linjer kode du har programmert i dette språket:

4.5.3) *Hva er din vurdering av hvor godt du kan dette programmeringsspråket?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

4.6) Spesifikk programmeringskompetanse - SmallTalk

4.6.1) *Hvor mange måneders arbeidserfaring har du totalt med dette programmeringsspråket?

4.6.2) *Estimer omtrentlig hvor mange linjer kode du har programmert i dette språket:

4.6.3) *Hva er din vurdering av hvor godt du kan dette programmeringsspråket?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

4.7) Spesifikk programmeringskompetanse - Pascal
4.7.1) *Hvor mange måneders arbeidserfaring har du totalt med dette programmeringsspråket?

4.7.2) *Estimer omtrentlig hvor mange linjer kode du har programmert i dette språket:

4.7.3) *Hva er din vurdering av hvor godt du kan dette programmeringsspråket?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

4.8) Spesifikk programmeringskompetanse - Python
4.8.1) *Hvor mange måneders arbeidserfaring har du totalt med dette programmeringsspråket?

4.8.2) *Estimer omtrentlig hvor mange linjer kode du har programmert i dette språket:

4.8.3) *Hva er din vurdering av hvor godt du kan dette programmeringsspråket?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

4.9) Spesifikk programmeringskompetanse - C
4.9.1) *Hvor mange måneders arbeidserfaring har du totalt med dette programmeringsspråket?

4.9.2) *Estimer omtrentlig hvor mange linjer kode du har programmert i dette språket:

4.9.3) *Hva er din vurdering av hvor godt du kan dette programmeringsspråket?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

4.10) Spesifikk programmeringskompetanse - Annet språk I

4.10.1) Angi navnet på språket

4.10.2) Hvor mange måneders arbeidserfaring har du totalt med dette programmeringsspråket?

4.10.3) Estimer omtrentlig hvor mange linjer kode du har programmert i dette språket:

4.10.4) Hva er din vurdering av hvor godt du kan dette programmeringsspråket?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

4.11) Spesifikk programmeringskompetanse - Annet språk II

4.11.1) Angi navnet på språket

4.11.2) Hvor mange måneders arbeidserfaring har du totalt med dette programmeringsspråket?

4.11.3) Estimer omtrentlig hvor mange linjer kode du har programmert i dette språket:

4.11.4) Hva er din vurdering av hvor godt du kan dette programmeringsspråket?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

Verktøykjennskap

5.1) Erfaring med JBuilder
5.1.1) *Hvor mange års erfaring har du totalt med JBuilder?

5.1.2) *Hvor godt mener du at du kan JBuilder?
1 = Kan svært lite/ingenting, 5 = Ekspert

1 2 3 4 5

5.2) Erfaring med annet verktøy.
5.2.1) *Hvilke andre verktøy har du brukt og kjenner?
(Kryss av flere alternativer)

Eclipce
JBoss
Visual Cafe
Visual Age
Together
IBM Rational
JEdit
Visual J++
JCreator
SUN J2SDK
GNU Emacs
OptimalJ
Textpad

Annet, spesifiser:

5.2.2) *Skriv ned de tre verktøyene du har brukt mest og kjenner best, og hvor lang erfaringstid du har med hver enkel?
(Grader erfaringen din med verktøyet ved å angi et tall mellom 1-5 hvor 1 er lite og 5 er mye)

Designmetoder/ -notasjoner og pattern
6.1) Erfaringsskjema - Hvilke designmetoder/-notasjoner kjenner du til? (gradert fra 1-5)
(1 = Kan svært lite/ingenting, 5 = Ekspert)

6.1.1) *UML

1 2 3 4 5

6.1.2) *Data-driven design (relasjonsdatabaser eller lignende).

1 2 3 4 5

6.1.3) *OMT (Object modelling technique)

1 2 3 4 5

6.1.4) *Responsibility-driven design

1 2 3 4 5

6.1.5) *Strukturert analyse og/eller strukturert design.

1 2 3 4 5

6.1.6) *Rollemodellering

1 2 3 4 5

6.1.7) *Design patterns

1 2 3 4 5

6.1.8) *Arkitektur patterns

1 2 3 4 5

6.2) Andre designmetoder/-notasjoner du kjenner til II
6.2.1) Navn:

6.2.2) Kjennskapsgradering
(1 = Kan svært lite/ingenting, 5 = Ekspert)

1 2 3 4 5

Jobbfunksjon

7.1) *Hva er din primære jobbfunksjon i dag?
(Kryss av flere alternativer dersom du har mer enn én primærfunksjon)

Kravhåndtering / kravanalyse
Arkitektur og design
Koding / programmering
Testing
Kvalitet- og prosessutvikling
Vedlikehold
Prosjekt- /linjeledelse
Integrering (deployment)
Annet

7.2) Hvis du svarte "Annet på forrige spørsmål, spesifiser:

7.3) *Hva mener du er dine styrker innen systemutvikling?
(Kryss av for flere alternativer dersom dette er ønskelig)

Kravhåndtering / kravanalyse
Arkitektur og design
Koding / programmering
Testing
Kvalitet- og prosessutvikling
Vedlikehold
Prosjekt- /linjeledelse
Integrering (deployment)
Annet

7.4) Hvis du svarte "Annet på forrige spørsmål, spesifiser:

7.5) *Hvor mange OO-prosjekter har du deltatt i?

7.6) Hvis du har vært leder for OO prosjekter, angi hvor mange måneder du har vært dette totalt, og for hvert enkelt prosjekt.

7.7) Hvor mange ansatte hadde du under deg i hvert av prosjektene?
(List opp prosjektene og angi cirka antall ansatte i prosjektene du listet opp i forrige spørsmål).

Avslutning

Takk!
Bakgrunnsskjemaet er nå ferdig utfylt.

Trykk på "Fullfør"-knappen for å avslutte denne sesjonen og vent på en bekreftelse. Når du har mottatt bekreftelsen trykker du på linken "Eksperiment" på menyen øverst (ved siden av "Logg av"-knappen) eller skriver adressen til SESE på nytt i nettleseren (www.sese.no) for å komme til neste del av eksperimentet.