Effort and Schedule Estimation of Software Development Projects

Kjetil Johan Moløkken-Østvold

Thesis submitted for the degree of PhD.

Department of Informatics
Faculty of Mathematics and Natural Sciences
University of Oslo

August 2004.

Abstract

This thesis explores different topics related to effort and schedule estimation of software development projects. The contributions presented here, in the form of observations and recommendations, should be of interest to a diverse audience involved in various aspects of software development projects.

For practitioners:

- Use of flexible development methods (e.g. agile, evolutionary and incremental) may reduce the magnitude of effort overruns.
- Software projects with a public client are likely to have greater effort overruns than comparable private projects.
- An estimation process that combines knowledge from experts with different backgrounds may facilitate increased estimation accuracy.

For researchers:

- There are considerable weaknesses in how surveys on software estimation accuracy are performed, both related to sampling, terminology and measurement.
- Software projects encounter, on average, a 30-40% effort overrun.
- Surveys reveal that expert judgment is by far the preferred estimation approach in the software industry.

For clients:

- Price should not be the only selection criterion when evaluating proposals from software providers.
- The clients should encourage use of flexible development methods in order to establish as close a dialogue as possible with the software providers.

For the political community:

- Procurement of software is still a major challenge in the public sector, and scandals surface frequently in a wide range of institutions.
- There is often an unbalanced mix of professionals in the public sector, with few resources available in the form of employees with skills in IT and project management.

The thesis consists of three sections, which address 1) *state of practice* in the software industry, 2) different methods for *improving estimation accuracy*, and 3) general *methodological aspects* in research on software estimation.

Related to international estimation practices and performance, a review of previous research performed as part of this thesis revealed that a majority of software projects (60-80%) had encountered effort overruns. The average project had effort overruns of 30-40%. The frequency (65-80%) and magnitude (20-25%) of schedule overruns was similar. The dominant approach to estimation was expert judgement. Similarly, a survey of software projects in Norway undertaken as part of this thesis found a frequency of 76% effort overruns, with an average magnitude of 41%. The frequency and magnitude of schedule overruns was 62% and 25% respectively. Expert estimation was by far the preferred estimation method. It was also observed that the type of client had an impact on the magnitude of effort overruns. Public projects had an average effort overrun of 67%, as opposed to the 21% average in private projects. This observed difference appears to be caused by systematic differences between private and public organizations found at the political, organizational, and individual levels. In an experiment on expert judgment, the results indicated that professionals in technical roles (project managers and developers) were significantly more optimistic, and less realistic, than professionals in non-technical roles (sales managers and user analysts) when estimating project effort.

Regarding the improvement of estimation practices, two different approaches were investigated. A controlled experiment showed that the combination of expert estimates through unstructured group discussion can reduce the existing bias towards estimates that are overly optimistic. In addition, it appears that the use of flexible development models (e.g. incremental, iterative, or agile) leads to a lesser magnitude of effort overruns when compared to the traditional sequential waterfall model.

It was also found that results in some previous studies on estimation accuracy may have been affected by methodological shortcomings, and that there is little attention directed towards ethical issues in software engineering research in general.

Acknowledgements

My gratitude extends first and foremost to supervisor Magne Jørgensen, for guidance and inspiration during the past years, and co-supervisor Tore Dybå for valuable input and interesting discussions.

In addition, thanks to:

- Dag Sjøberg and Erik Arisholm for their research input.
- Aslak Tveito for his helpfulness and positive attitude.
- Stein Grimstad for his decision to join the estimation team and for providing critical and insightful comments.
- Pål Sørgaard for all contributions.
- Sinan S. Tanilkan, Hans Gallis, Anette C. Rekdal and Siw E. Hove for their contributions to the BEST-Pro survey.
- Marek Vokáč, James Dzidek and Gunnar Carelius for stimulating discussions and contributions to previous, and ongoing, studies.
- Jo Hannay, Vigdis By Kampenes, Bente Anda and Amela Karahasanovic for their contributions on controlled experiments and research ethics.
- Hans Christian Benestad and Glen Farley for contributions to our studies.
- Tanja Gruscke, Ragnfrid Sjøberg and Håkon Ursin Steen for help on the BEST project.
- Chris Wright for proofreading.
- Reidar Conradi and all other members of the INCO project for their contributions.
- The Simula Research Laboratory administration for creating an excellent work atmosphere and a dynamic organization.
- The IfI-library employees for excellent help.
- Anonymous referees and adjudication committees for input on research papers.
- Nestor Alexander Haddaway for motivational contributions.

This research would not have been possible without the participation of the software industry. Many thanks go to all the professionals who helped with organization and contributions. This research was funded by the Norwegian Research Council under the project INCO as part of the IKT2010 program.

Thanks also to Aftenposten, Computerworld, Sintef, Abelia, the Auditor General of Norway, Steria, Software Innovation and Simula Research Laboratory for their interest in this research, invitations to seminars and contributions to an ongoing debate.

Finally, thanks to Anette and my family for supporting me throughout these years.

Contents

1. Introduction

"Overruns in time and money are usual. In fact, underruns are highly unusual. On the surface, those problems arise from the problems of specification and estimation. Loose and instable specifications certainly prevent timely development. But the programming estimation problem is difficult, even with good specifications for a new capability or a new development environment."

- Harlan D. Mills, 1976 [1]

Mastering the art of software project estimation is a major challenge for engineers and managers. In this thesis, software project estimation is defined as the process of predicting the effort (costs) and schedule (time) required for implementing a software solution, based on a requirement specification.

It seems that the task of accurately estimating software projects is as challenging today as it was thirty years ago [1-3]. Several important issues in research on software estimation remain unsolved. How frequent and large are actual effort and schedule overruns? Are there reasonable and cost-efficient approaches to improve the situation? What kinds of research methods are appropriate for investigating software project estimation? These unresolved issues were the main motivation for the work undertaken as part of this thesis.

Frequently cited surveys on software estimation have reported average effort overruns ranging from 89% [4] (or 189%, discussed in paper VIII [5] of this thesis) to 18% [6]. However, closer inspection reveals that such surveys often have methodological shortcomings that may have affected the results. Since effort estimation is an important research area, and has been the topic of many papers, books and seminars, there is a need for a repository of unbiased knowledge.

In addition, there appears to be a discrepancy between the focus of the research community and the practice of professional developers and managers. In order to improve estimation accuracy, much research has been directed at developing and adapting rigorous estimation models and frameworks. Formal estimation models have been based on a variety of measures of development size (such as lines of code [7] and function-points [8]), and a variety of model development approaches (such as linear regression [9]). However, all industry surveys show that these models are not frequently used [10-13]. Proponents of such

models have still not managed to provide empirical evidence that their employment leads to increased estimation accuracy when compared to the more widely used expert judgment approach [14]. The reason for the lack of use of formal models is probably a combination of the facts that they do not increase accuracy, are more complicated to use, and are less flexible than expert judgment.

The research efforts directed towards aiding and improving the expert judgment approach have focused primarily on work breakdown structures (WBS) [15, 16], checklists [17, 18], experience databases [19], and group-based estimates [18, 20, 21]. These supporting tools may lead to improvements in some cases, but there is no conclusive evidence that they solve the problem of inaccurate estimates.

There are also those who suggest that other aspects of the software project, such as choice of development model [22-25], are important for reducing effort and schedule overruns. Such proponents favour the use of more flexible (i.e. incremental, agile or evolutionary) development models over the traditional sequential models.

Interestingly, the results from other research areas, such as transport infrastructure projects, reveal the same frequency and magnitude of effort overruns [26] as reported in software projects. In addition, researchers have observed that overruns are independent of region and appear to be stable over time. As suggested, both within software engineering [27, 28] and other research areas [26], it is possible that issues not directly related to the shortcomings of the estimation process itself may account for most of the overruns. Such issues include the influence of politics and economics on estimates.

As long as estimates are not deliberately wrong or deemed unimportant, the approaches described in this thesis can be used as simple and cost-efficient tools for analyzing and improving software estimates.

Independent of which aspect of the software estimation problem being studied, there is also a clear limitation that there is neither a coherent research framework nor an established terminology [29]. Such shortcomings have contributed to the limited validity of several previous studies.

In order to explore the different topics related to software estimation presented in this introduction, this thesis investigates three general research questions:

RQ1: What is the state, in the industry, of the practice and level of performance related to software project estimation?

RQ2: Are there sensible and cost-efficient ways to improve the software process so as to improve estimation accuracy?

RQ3: Are there current methodological or ethical challenges related to how research on software estimation is conducted?

These research questions are reflected in the organization of this thesis, which has three main parts, and a short summary. Sections 1 to 4, inclusive, of the summary provide an overview of the problem area and the research conducted as part of this thesis. In addition, the individual papers and research results are described briefly. All papers are included in full in the main sections after the summary.

2. Research Method

The studies conducted as part of this thesis all involved professional software practitioners as participants. This increases the external validity of the results. However, it also poses challenges, such as the importance of having a rigorous research method and an awareness of ethical issues. This section briefly outlines our research method, while ethical concerns are discussed in paper VII [30].

2.1. Description of Studies

Three main studies contributed to this thesis: a review of previous international surveys on effort estimation, a survey of the Norwegian software industry, and a controlled experiment.

Study 1 (S1): As part of the preparation for our research on software estimation practices and performance in the Norwegian software industry, we performed a review of previous surveys. This approach appeared to be the best and most cost-efficient way of acquiring knowledge about international practices and performance related to software estimation.

A common problem with reviews of previous surveys is that of determining which studies to include. However, in this review, this was not an issue because the amount of previous surveys available was limited. All surveys printed in international journals and conference proceedings were considered. Included in the review were all identified surveys that focused on one or more of the following topics: 1) choice of estimation approach, 2) level of estimation accuracy and bias, 3) the perceived importance of estimation accuracy, and 4) causes of estimation inaccuracy.

The review provided us with: 1) a methodological framework for our own survey, 2) an overview of state of the practice, and 3) benchmark results for our survey. The results are presented in paper I [31]. An updated summary of the results was subsequently used in the introduction of a paper describing our survey of the Norwegian software industry (paper II [32]).

Study 2 (S2): Between February and November 2003, we conducted an in-depth survey (BEST-Pro) on software estimation practices and performance in the Norwegian software industry. The purpose of the survey was to compare results with those reported from other countries. In addition, we wanted to address topics omitted in previous surveys, such as the

impact of development method and type of client on estimates. Results related to the various topics are discussed in papers II, III, VI and VIII [32-35].

The rationale for using a survey, instead of, for example, a (multiple) case study, was to ensure that we included a wide range of companies, in terms of size, experience and product type.

The main reason for choosing an interview-based survey, rather than mailed questionnaires, was that we were aware of the shortcomings of surveys based on mailed questionnaires. In surveys based on personal interviews, there is a higher response rate, participants are more dedicated and it is possible to resolve any ambiguities that might arise [36]. The last of these is especially important in our area, since there are no uniform standards regarding terminology [29]. In return for their participation, the companies could choose between financial compensation and attending a seminar on estimation hosted by Simula Research Laboratory. Almost all companies choose the latter option, and we hosted a seminar with over sixty project managers, senior managers and executives from the Norwegian software industry.

Study 3 (S3): As part of previous work, we had conducted a controlled two-phase experiment with participants from a Norwegian web development company. The research aimed at determining whether the company role of software practitioners influenced their estimation performance. In addition, we investigated whether increased estimation accuracy could be achieved by combining expert estimates through unstructured group discussion. The results from this experiment were so interesting that, as part of this thesis, they were further analyzed and elaborated on in papers IV and VI [37, 38].

The use of a controlled experiment, as opposed to a more realistic action research approach, was based on the fact that we wanted to isolate and control two elements (individual opinion and group discussion) from the noise created by the work environment in a software project. We aspired to balance realism and control, and feel that this was achieved to a sufficient extent.

The company received financial compensation for their efforts, with participants being allowed to count the experiment as normal billable hours. Hence, a smooth and unbiased recruitment procedure was ensured.

However, the use of participants from only one company entails that generalizations on the basis of the results of the experiment should be limited to companies with similar properties, in terms of types of product, maturity and processes.

The other minor studies that contributed to this thesis are described in the respective papers. In sum, the balance of the studies appears satisfactory, with different methods employed to explore different aspects of the software estimation problem.

2.2. Terminology and Measures

When conducting research on estimation accuracy, it is problematic that there exist no agreed upon standards for terminology and measurement in the software engineering community. This problem is independent of the type of study, and includes such issues as uncertainty about what an estimate represents, the stage in the development process at which the estimate is provided, and how to calculate estimation accuracy.

At present, there exists no precise definition of what an estimate is [29]. Neither in the research community nor among professional practitioners is the term "estimate" used in a concise manner. In addition, much research on software estimation tends to treat a software estimate as a single fixed value. During the course of our research, however, we have noticed that software projects often have several effort estimates. This problem, and how this poses challenges to estimation models, has also been addressed by Edwards and Moores [39]. The term *effort estimate* is frequently used for different purposes, such as:

- Most likely effort – the number of man-hours the estimator(s) believe is the most likely required workload for completion of the project requirements.
- 50-50 estimate – An estimate that is just as likely to be optimistic as pessimistic.
- Planned effort – A project plan, perhaps optimistic in order to promote efficiency.
- Budgeted effort – The budget allocated (in costs/man-hours), often with a risk buffer, in order to complete a project.
- Project bid – The bid communicated to a potential client in order to win a contract.

In our studies, we are most often interested in measuring and improving estimation capability. Therefore, we compared the outcome (actual effort and schedule), with the *most likely* effort and schedule estimates. The most likely estimate is not subject to as much pressure as other estimates, e.g. the project bid, which often is affected by internal political pressure and "price-to-win".

If a project has multiple most likely estimates calculated during the course of the project, we use the estimate provided at the stage when the decision to start the project was made

(i.e. the planning phase or when an agreement with a client is reached). This research approach is in accordance with studies on project estimation in other areas of research [26].

There exists several different ways of calculating estimation accuracy. The most common is the MRE (Magnitude of Relative Error) measure [40], which is calculated as:

$$MRE = \frac{|x - y|}{x}, \quad \text{x = actual and y = estimated.} \tag{1}$$

Even though the MRE is the most widely used measure of estimation accuracy [41], one must be aware that it has unfortunate properties. The main concern is, perhaps, the fact that underestimated and overestimated projects are weighted unevenly, with underestimation not weighted sufficiently [42]. In addition, it does not seem sensible from a practitioner's point of view, since estimation performance in the software industry is often based on the estimated effort [43-46].

Estimation accuracy can also be measured according to the method used by Bergeron and St-Arnaud [13]. They used a formula recommended by Conte et al. [40]. This is similar to the measure that is labelled as MER (Magnitude of Error Relative to the estimate) elsewhere [42], and is calculated as:

$$MER = \frac{|x - y|}{y}, \quad \text{x = actual and y = estimate.} \tag{2}$$

While MRE measures estimation error relative to the actual effort, MER measures it relative to the estimate. It is argued that this measure is more meaningful, since profit or loss should be calculated on the basis of expected cost by most project managers.

The BRE (Balanced Relative Error) is, as its name indicates, a more balanced measure [47] than MRE and MER. It is calculated as:

$$BRE = \frac{|x - y|}{\min(x, y)}, \quad \text{x = actual and y = estimate.} \tag{3}$$

Assume, for example, that two projects estimate the required effort to be 1000 work-hours. Project A spends 500 work-hours, while project B spends 2000 work-hours. The MRE of Project A is 1.00, while the MRE of Project B is only 0.50. Employing the MER,

Project A's result is 0.50, while Project B's result is 1.00. The BRE evenly balances overestimation and underestimation, leading to a result of 1.00 for both projects.

In research on forecasting, a similar measure, termed modified MAPE (Mean Absolute Percentage Error), has been proposed [48] to balance overestimation and underestimation:

$$\mathrm{mod}\,ifiedMAPE = \left| \frac{x - y}{(x + y)/2} \right|, \quad \text{x = actual and y = estimate.} \qquad (4)$$

In the example above, the modified MAPE would be 0.67 for both projects. This number may be difficult to relate to for practitioners than BRE.

Two other common prediction measures are *residuals* and *pred(m)*. The residuals are calculated straightforwardly as: residual = actual – estimate (5), while the pred(m) measure (6) concerns the amount of project estimates that fall within m percent of the actual value. Typically, m is set to 25.

These are the measures most frequently encountered in research on software estimation and related areas. However, as all of these have shortcomings, more complex measures, such as SD, RSD and LSD are also used [42].

It is possible for the choice of accuracy measure to have a huge impact on results, as has been shown in other studies [42, 46]. The same could well be true of the studies in this thesis.

In the review of previous surveys (S1), we did not choose evaluation criteria, but observed the choices of others. The previous surveys relied mainly on the MER and MRE/BRE measures but without the absolute values for each project, in order to observe the direction of the effect. It is however, often difficult to discern the evaluation criteria, since the research method is often not described precisely.

For our own survey (S2), it was important to be able to choose the most appropriate evaluation criteria, as well as being able to compare with previous studies, and make our findings accessible for practitioners. More complicated measures such as SD, RSD and LSD were therefore rejected, even though these are more robust for other purposes, such as model evaluation [42]. Use of residuals was inappropriate, since we were interested in the relative project overruns. Pred(m) could have been used to describe our data, but this does not provide as much information about the magnitude of the overruns. However, we did use a variant of pred(m) in a report [49] on our survey, in order to provide additional information. The modified MAPE has not been used by others in the Software Engineering

community. We do not consider such lack of use to be an excuse for not employing it. Rather, we consider that, for most purposes, analyses of the BRE and the modified MAPE will provide the same results. However, the use of the modified MAPE will be difficult for practitioners to relate to.

Therefore, the choice was between MRE, MER and BRE. We decided to use measures based on BRE, since this places an even emphasis on overestimation and underestimation. It is robust and sensible, from both theoretical and practical points of view. However, the main results from statistical analyses would have been similar, whether MRE, BRE, MER or the modified MAPE had been used, due to the nature of the data.

In addition, whether using BRE, MRE or some of the other measures there are further factors to address. Are we interested only in the accuracy of the estimates, or in the direction of the effect, or both? Absolute values, whether BRE, MRE or MER, do not capture estimation bias, because they are not concerned with the direction of the estimation inaccuracy.

BREbias measures both the size and the direction of the estimation error, i.e., whether there is a bias towards optimism or pessimism:

$$BREbias = \frac{(x - y)}{\min(x, y)}, \quad \text{x = actual and y = estimate.} \tag{7}$$

We were most interested in the BREbias, to be able to analyze the direction of the effect.

In addition, for a set of observations, we will probably have differences if we compare the mean and the median of the observations, e.g. MMRE and MdMRE, as illustrated by Foss et al. [42]. When aggregating the results, we decided to report both the mean and median BREbias. As described in the relevant papers, the statistical analyses are either on the mean or medians based on properties of the samples.

If we had been comparing two similar and competing estimation methods, we would have been more cautious about using BRE, because it may not always reveal the "best" method [42]. However, in our survey this was not the objective, and we believe that BRE and BREbias, presented with both means and medians were appropriate.

In papers on our controlled experiment (S3), we simply compared the estimates in statistical analyses, since there were no actual values.

3. Summary of Research Papers

This thesis is composed of eight papers (I – VIII), roughly divided into three topics of interest. A brief description of each paper, and how they are related to each other, is presented in this section. An overview of the relationship between topics and papers is presented in Table 1.

	I	*II*	*III*	*IV*	*V*	*VI*	*VII*	*VIII*
T1. International estimation practices	x	(x)	(x)					(x)
T1. Norwegian estimation practices		x	x			x		
T1. The impact of type of client			x					
T1. The impact of estimator background				x				
T2. Combination of estimates					x			
T2. Impact of development model						x		
T3. Research ethics in software engineering							x	
T3. Sampling and measurement		(x)				(x)		x

Table 1: Topics and Papers

The studies described in the previous section contributed to the different papers as shown in Table 2.

	I	*II*	*III*	*IV*	*V*	*VI*	*VII*	*VIII*
S1. Review of previous surveys	x	x						(x)
S2. BEST-Pro survey		x	x			x		(x)
S3. Controlled experiment				x	x			
Other studies			x				x	x

Table 2: Studies and Papers

3.1. Papers on State of the Practice (T1)

Most of the papers in this thesis are concerned with analyzing the state of the practice related to software estimation in the industry. Topics of interest are how software projects are estimated, what the level of accuracy is, and possible differences based on estimation approach, role of the estimator, or type of client. In order to address this matter, we have performed systematic literature studies, an in-depth survey, and a controlled experiment.

Paper I: A Review of Surveys on Software Effort Estimation

Kjetil Johan Moløkken-Østvold and Magne Jørgensen.

IEEE International Symposium on Empirical Software Engineering (ISESE 2003), 2003. September 30 - October 1, Rome, Italy. Page 223-230. IEEE Computer Society. ISBN 0-7695-2002-2.

This paper summarizes estimation knowledge through a review of surveys on software effort estimation. Main findings were that: (1) Most projects (60-80%) encounter effort and/or schedule overruns. The overruns, however, seem to be lower than the overruns reported by some consultancy companies. For example, Standish Group's 'Chaos Report' describes an average cost overrun of 89%, which is much higher than the average overruns found in other surveys, i.e., 30-40%. (2) The estimation methods in most frequent use are expert judgment-based. A possible reason for the frequent use of expert judgment is that there is no evidence that formal estimation models lead to more accurate estimates. (3) There is a lack of surveys including extensive analyses of the reasons for effort and schedule overruns.

Paper II: A Survey on Software Estimation in the Norwegian Industry

Kjetil Johan Moløkken-Østvold, Magne Jørgensen, Sinan S. Tanilkan, Hans Gallis, Anette C. Lien and Siw Elisabeth Hove.

10th International Symposium on Software Metrics. 2004. Chicago, Illinois, USA: IEEE Computer Society. pp. 208-219.

This paper provides an overview of the estimation methods that software companies apply to estimate their projects, why those methods are chosen, and how accurate they are.

In order to improve estimation accuracy, such knowledge is essential. We conducted an in-depth survey, where information was collected through structured interviews with senior managers from 18 different companies and project managers of 52 different projects. We analyzed information about estimation approach, effort estimation accuracy and bias, schedule estimation accuracy and bias, delivered functionality and other estimation related information. Our results suggest, for example, that average effort overruns are 41%, that the estimation performance has not changed much the last 10-20 years, that expert estimation is the dominating estimation method, that estimation accuracy is not much impacted by use of formal estimation models, and that software managers tend to believe that the estimation accuracy of their company is better than it actually is.

Paper III: Project Management of Public Software Projects: Avoiding Effort Overruns

Kjetil Johan Moløkken-Østvold, Magne Jørgensen, Pål Sørgaard and Stein Grimstad.
Submitted to Information and Software Technology.

Effort overruns, abandonment, lawsuits, system breakdowns, and other "scandals" appear to be the rule, rather than the exception, where public software projects are concerned. This has been reported in the United States, The United Kingdom, Norway and several other OECD countries. It is important to note that this is not only the opinion of scandal-seeking tabloids, but also that of the more serious technical press. In addition, in the past five years, public officials in several countries have hosted conferences on the topic, and several reports addressing this problem have been written.

However, is it just the transparency of public projects that make them easily accessible by the media and the general public? Do public projects really face larger effort overruns than private projects? Or is this just a myth? In order to address this problem, we conducted a survey that compared effort overruns, and other factors relevant for software engineering project managers, of public and private software projects in Norway. We found that there are, indeed, causes for concern for those involved in public projects. These projects had effort overruns of a significantly greater magnitude than private projects.

Depending on the type of contract, effort overruns are either paid for by the client, written off as losses by the contractors, or there is a shared responsibility. In addition, project managers who face problems may be tempted to cut back on testing or functionality in order to reduce potential overruns, thus delivering lesser value to the clients. Therefore,

we present an overview of the problem, and offer advice that is relevant for both software providers and public clients that seek to reduce effort overruns.

Paper IV: Expert Estimation of the Effort of Web-Development Projects: Are Software Professionals in Technical Roles More Optimistic Than Those in Non-Technical Roles?

Kjetil Johan Moløkken-Østvold and Magne Jørgensen.

Empirical Software Engineering, 2005, Volume 10, Issue 1, pp 7-29.

Estimating the effort required to complete web-development projects involves input from people in both technical (e.g., programmers), and non-technical roles (e.g., user interaction designers). This paper examines how the role and type of competence impact the estimation strategies and performance. An analysis of actual web-development project data and results from an experiment suggest that people with technical competence provided less realistic project effort estimates than those with less technical competence. This means that more knowledge about how to implement a requirement specification does not always lead to better estimation performance. We discuss, amongst others, two possible reasons for this observation: (1) Technical competence induces a bottom-up, construction-based estimation strategy, while lack of this competence induces a more "outside" view of the project, using a top-down estimation strategy. An "outside" view may induce more use of the history and reduce the bias towards over-optimism. (2) The software professionals in technical roles perceive that they are evaluated as more skilled when providing low effort estimates. A consequence of our findings is that the choice of estimation strategy, estimation evaluation criteria and feedback are important to avoid underestimation.

3.2. Papers on Improving Software Estimation Accuracy (T2)

The section on improving software estimation accuracy suggests ways in which practitioners may enhance their performance. The first paper investigates how combining estimates from different experts may reduce individual biases, while the second investigates whether choice of development model affects estimation accuracy.

Paper V: Group Processes in Software Effort Estimation.

Kjetil Johan Moløkken-Østvold and Magne Jørgensen.

Journal of Empirical Software Engineering, 2004, Volume 9, Issue 4, pp 315-334.

The effort required to complete software projects is often estimated, completely or partially, using the judgment of experts, whose assessment may be biased. In general, such bias as there is seems to be towards estimates that are overly optimistic. The degree of bias varies from expert to expert, and seems to depend on both conscious and unconscious processes. One possible approach to reduce this bias towards over-optimism is to combine the judgments of several experts. This paper describes an experiment in which experts with different backgrounds combined their estimates in group discussion. First, twenty software professionals were asked to provide individual estimates of the effort required for a software development project. Subsequently, they formed five estimation groups, each consisting of four experts. Each of these groups agreed on a project effort estimate via the pooling of knowledge in discussion. We found that the groups submitted less optimistic estimates than the individuals. Interestingly, the group discussion-based estimates were closer to the effort expended on the actual project than the average of the individual expert estimates were, i.e., the group discussions led to better estimates than a mechanical averaging of the individual estimates. The groups' ability to identify a greater number of the activities required by the project is among the possible explanations for this reduction of bias.

This paper is an expansion of a previous paper [50], which has selected for a special issue of Empirical Software Engineering as best paper from the co-located EASE/PPIG 2003 conferences.

Paper VI: The Impact of Development Model on Estimation Accuracy in Software Projects

Kjetil Johan Moløkken-Østvold and Magne Jørgensen.

Submitted to IEEE Transactions on Software Engineering.

Flexible software development models, e.g., evolutionary and incremental, have become increasingly popular. Advocates of these models claim that among the benefits is improved estimation accuracy, which is one of the main challenges of software project management. This paper describes an in-depth survey of software development projects. The results support the claim that estimation accuracy improves when a flexible development model is

applied. The reason for the improvement is not obvious. We found, for example, no difference in project size, estimation process, or delivered proportion of planned functionality between projects applying different types of development model. However, we did find that the type of client had a strong impact on the estimation accuracy when applying flexible development models. This suggests that a better client relationship, which is facilitated by flexible development models, is an important reason for the observed improvement in estimation accuracy.

3.3. Papers on Methodological Aspects (T3)

These papers explore two different methodological issues related to research on software estimation (and software engineering in general). One discusses ethical concerns when using industrial participants, while the other discusses problems related to measurement and (lack of) random samples.

Paper VII: Ethical Concerns when Increasing Realism in Controlled Experiments with Industrial Participants.

Kjetil Johan Moløkken-Østvold.

Accepted for HICSS38 (Hawaii International Conference on System Sciences), 2005.

The emerging interest for realistic controlled experiments in computer science has created a need for focus on related research ethics. Increased realism and scale in experimental studies pose new challenges which have not been debated to a sufficient extent. Specifically, there can be conflicts between the ethical principles of scientific value and informed consent. This paper provides an account of related previous work in computer science research ethics. To illustrate, two large scale software engineering experiments with industrial participants are described. Challenges and solutions in these experiments are discussed in light of current ethical guidelines. Interviews and debriefing sessions with industrial participants from these, and other, experiments are also provided. These reveal that there not necessarily will be ethical problems with increased realism, given that the researchers respect the principles of informed consent, benefice and confidentiality.

Paper VIII: How Large Are Software Cost Overruns? Critical Comments on the Standish Group's CHAOS Reports.

Magne Jørgensen and Kjetil Johan Moløkken-Østvold.

Submitted to Information and Software Technology.

The Standish Group (www.standishgroup.com) claims that the results of their CHAOS research, i.e., their large-scaled surveys conducted in 1994, 1996, 1998, 2000 and 2002, are the most widely quoted statistics in the IT industry. This may very well be true. Quoted with particular frequency are the results described in the 1994 CHAOS report, probably because the 1994 CHAOS report is free and easily can be downloaded from the web. The results of that report have been used in several recent governmental reports, project reviews, and research studies. Examples are the PITAC 1999 report and cost estimation studies. An important result from the 1994 CHAOS research is the reported 189% average cost overrun of so-called challenged projects, i.e., projects not on time, on cost, and with all specified functionality. In this paper we argue that the 189% average cost overrun number, as it is commonly interpreted, is not consistent with results of other cost accuracy surveys and probably far too high to reflect the average cost overrun in that period. The measures and the research method of the CHAOS survey are insufficiently described to evaluate the quality of the results, e.g., there are many possible interpretations of what is meant by 'cost overrun' in the CHAOS reports. We should therefore cease to trust the 189% average cost overrun as a reference point for performance of software projects until such time as the Standish Group disclose how they measure cost overrun and how they conduct their research.

3.4. Identification of the Work the Author

All papers in this thesis, except one, were co-authored with other researchers. It is, therefore, necessary to highlight the work done by the author of this thesis. All major studies, presented in subsection 2.1, that form the basis of this thesis were initiated and managed by the author. The review of previous surveys (S1), and the controlled experiment (S3) was executed by the author. The BEST-Pro survey (S2) was planned and executed with the help of others, mainly supervisor Magne Jørgensen, Sinan S. Tanilkan and Hans Gallis, with the author as project manager. All data analysis, except on some parts of the BEST-Pro survey, was conducted by the author.

The papers presented here were mainly written by the author, with input from Magne Jørgensen and other co-authors. The author is a co-author of one paper (VIII [5]).

As part of the PhD work, the author also contributed to the following papers that were not included in this thesis:

1. Magne Jørgensen, Teigen K. H. and Kjetil Johan Moløkken-Østvold, *Better sure than safe? Overconfidence in judgment based software development effort prediction intervals,* Journal of Systems and Software, vol. 70, no 1-2, p 79-93, 2004.

2. Magne Jørgensen and Kjetil Johan Moløkken-Østvold, *Understanding Reasons for Errors in Software Effort Estimates,* IEEE Transactions on Software Engineering, vol. 30, no 12, 2004.

3. Stein Grimstad, Magne Jørgensen and Kjetil Johan Moløkken-Østvold, *Software Effort Estimation Terminology: The Tower of Babel,* Submitted to Information and Software Technology, 2004.

4. Kjetil Johan Moløkken-Østvold and Magne Jørgensen, *Software Effort Estimation: Unstructured Group Discussion as a Method to Reduce Individual Biases,* The 15th Annual Workshop of the Psychology of Programming Interest Group (PPIG 2003). 8-10 April 2003. Keele University, UK, pp. 285-296. (Selected as best paper for appearance in a special issue of Empirical Software Engineering.)

5. Kjetil Johan Moløkken-Østvold, Anette C. Lien, Magne Jørgensen, Sinan S. Tanilkan, Hans Gallis and Siw Elisabeth Hove, *Does Use of Development Model Affect Estimation Accuracy and Bias?,* Product Focused Software Process Improvement: 5th International Conference, PROFES 2004, Kansai Science City, Japan, April 5-8, 2004. ISBN: 3-540-21421-6. pp. 17-29. (LNCS 3009, Springer Verlag).

6. Magne Jørgensen and Kjetil Johan Moløkken-Østvold, *Eliminating Over-Confidence in Software Development Effort Estimates,* Product Focused Software Process Improvement: 5th International Conference, PROFES 2004, Kansai Science City,

Japan, April 5-8, 2004. ISBN: 3-540-21421-6. pp 174-184. (LNCS 3009, Springer Verlag).

7. Magne Jørgensen and Kjetil Johan Moløkken-Østvold, *A Preliminary Checklist for Software Cost Management,* IEEE International Conference on Quality Software, Dallas, USA, p 134-140, November 2003.

8. Magne Jørgensen and Kjetil Johan Moløkken-Østvold, *Situational and Task Characteristics Systematically Associated With Accuracy of Software Development Effort Estimates,* Information Resources Management Association Conference, Philadelphia, USA, May, 2003, p 824-826.

9. Magne Jørgensen and Kjetil Johan Moløkken-Østvold, *Combination of software development effort prediction intervals: Why, when and how?,* Fourteenth IEEE Conference on Software Engineering and Knowledge Engineering (SEKE'02), July 15-19, 2002, Ischia, Italy, pp. 425-428.

10. Kjetil Johan Moløkken-Østvold, *Software Effort Estimation: Planning XP Guidelines Compared to Research on Traditional Software Development,* Presentation at the Ph.D. Symposium in Fourth International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP 2003), May 25-29, 2003, Genova, Italy. pp 441-442. Lecture Notes in Computer Science, Springer-Verlag.

11. Kjetil Johan Moløkken-Østvold, Magne Jørgensen, Sinan S. Tanilkan, Hans Gallis, Anette C. Lien and Siw Elisabeth Hove, *Project Estimation in the Norwegian Software Industry – A Summary,* Simula Report 2004-03. Results from the BEST-Pro (Better Estimation of Software Tasks and Process Improvement) survey.

Some of the papers have recently been submitted, while others are still in press. An updated list of publications can be found at:
http://www.simula.no/people_publication.php?people_id=69&internal_people=y.

3.5. Notes

Two of the papers (IV and V) were based on the results of an experiment conducted while the author was a Masters student. However, the material has been further analyzed and significantly expanded, and none of the papers were included in the Masters thesis.

There have been minor changes in formatting and updated references in papers included in this thesis compared to previously published versions, in order to accommodate the readers.

4. Main Research Contributions

The research conducted as part of this thesis has provided contributions in the form of observations and suggestions for improvement that should be relevant for researchers, practitioners, clients and the political community.

For practitioners:

- **Use of flexible development methods (e.g. agile, evolutionary and incremental) may reduce the magnitude of effort overruns.** Software projects with a flexible development model (e.g. incremental and evolutionary) appear to have several benefits, including increased estimation accuracy, when compared to traditional sequential models. In order to reduce overruns, software professionals should explore flexible development models, and assess whether the use of such models would benefit their organization.
- **Software projects with a public client are likely to have a larger magnitude of effort overruns than do comparable private projects.** Our contribution also includes an in-depth analysis of possible reasons for this observation, and guidelines on how to improve projects with public clients. When dealing with public clients, software practitioners are advised to be mindful of possible hazards related to 1) political, 2) organizational, and 3) individual factors. It is, however, essential to note that it is the responsibility of the software providers to provide *realistic* estimates, independent of the capabilities of the client.
- **An estimation process that combines knowledge from experts with different backgrounds may facilitate increased estimation accuracy.** Software Managers should focus on assessing and improving their estimation process. We found that the company role of the estimators plays an important part in estimation work. In our experiment. people in non-technical roles provided less optimistic estimates than those in technical roles. In order to reduce individual biases, a simple and cost-efficient method for improving estimation accuracy is to combine, through an informal group meeting, the opinions of several experts with diverse backgrounds.

For researchers:

- **There are considerable weaknesses in how surveys on software estimation accuracy are performed, related to sampling, terminology and measurement.** In particular, we show that the famous CHAOS report by the Standish Group has

methodological shortcomings that place in question the validity of the results. Researchers should try, to a greater extent, to establish a common framework for terminology and measurement, and address problematic issues pertaining to sampling when reporting research results.

- **Software projects on average encounter a 30-40% effort overrun.** It seems that the frequency and magnitude of effort and schedule overruns are similar in different regions, and our survey of estimation performance found results similar to those of studies conducted during the past twenty years. Further research should seek to explore the underlying reasons for these observations.
- **Surveys reveal that expert judgment is by far the most preferred estimation approach in the software industry.** It seems that previous and ongoing efforts to develop generic estimation models, using measures such as lines of code, are futile. Such models are difficult and time consuming to employ, and cannot capture all the knowledge needed to address a complicated problem. In addition, there are many input parameters in such models that it is impossible to know at the start of the project. More research should be directed at addressing the needs of software practitioners, e.g. through the development of checklists, experience databases and procedures for reducing estimation bias. In addition, more attention should be paid to political and economical aspects of the software development process, because estimation is not isolated from such factors.

For clients:

- **Price should not be the only selection criterion when evaluating proposals from software providers**. Software clients should act more systematically when procuring software. They should seek to investigate the previous track-record of the providers, especially related to performance in similar projects. If the clients themselves are not able to assess the capabilities of the providers accurately, hiring a neutral outside party for assistance should be considered.
- **The clients should encourage the use of flexible development methods in order to establish a close a dialogue as possible with the software providers.** A close dialogue is facilitated when flexible development methods are used. This can reduce the chance of deliveries that does not fulfil the clients' needs, and identify problems already at an early stage. In addition, flexible development frequently allows for

delivery of the most critical parts of a system at an early stage, which will add value for the clients.

For the political community:

- **Procurement of software ia still a major challenge in the public sector, and scandals surface frequently in a wide range of institutions.** Intriguingly, the findings from our BEST-Pro survey contributed to a debate in the Norwegian media. The results made the headlines in Norway's leading newspaper, Aftenposten. This led to an interest in public project overruns, which in turn generated a discussion in the Norwegian Parliament. We were invited to present our results to several government departments, the Office of the Auditor General of Norway, and several seminars and interested software companies. The debate continued in Aftenposten, Computerworld and other media, and this has generated an increased focus on software estimation research in Norway. However, public awareness alone is not sufficient for action; such problems have been identified previously, and professional advice has been repeatedly ignored. On a larger scale, the decision makers should seek to implement policies that stimulate the improvement of processes internally, and enable mechanisms to operate that will reduce the focus on price in software procurement.
- **There is often an unbalanced mix of professionals in the public sector, with few resources available in the form of employees with skills in IT and project management.** Officials should encourage the employment and education of professionals proficient in software engineering and project management. There should also be internal career paths and professional development for people in these positions. During the presentation of the material included in this thesis, we encountered several public employees who appreciate our findings. They recognize their own situation, and state that they often are required to be internal project managers for procurement projects in addition to performing their regular organizational duties. They often complain that they lack the requisite education and support, and that their efforts are not recognized in terms of career development.

References

1. Mills, H.D., *Software Development.* IEEE Transactions on Software Engineering, 1976. **2**(4): pp. 265-273.
2. Wolverton, R.W., *The Cost of Developing Large-Scale Software.* IEEE Transactions on Software Engineering, 1974. **23**(6): pp. 615-636.
3. Walston, C.E. and C.P. Felix, *A Method for Programming Measurement and Estimation.* IBM Systems Journal, 1977. **16**(1): pp. 54-73.
4. Standish, G., *The Chaos Report*. 1994, The Standish Group.
5. Jørgensen, M. and K. Moløkken-Østvold, *How Large Are Software Cost Overruns? Critical Comments on the Standish Group's CHAOS Reports.* Submitted to Information and Software Technology (Can be downloaded at: http://www.simula.no/publication_one.php?publication_id=711), 2004.
6. Sauer, C. and C. Cuthbertson, *The State of IT Project Management in the UK 2002-2003*. 2003, Templeton College, University of Oxford.
7. Boehm, B., et al., *Software Estimation with COCOMO II*. 2000: Prentice-Hall.
8. Matson, J.E., B.E. Barrett, and J.M. Mellichamp, *Software development cost estimation using function points.* IEEE Transactions on Software Engineering, 1994. **20**(4): pp. 275-287.
9. Miyazaki, Y., et al., *Robust regression for developing software estimation models.* Journal of Systems and Software, 1994. **27**(1): pp. 3-16.
10. Heemstra, F.J. and R.J. Kusters. *Controlling Software Development Costs: A Field Study*. *International Conference on Organisation and Information Systems*. 1989. Bled, Yugoslavia. pp. 652-664.
11. McAulay, K. *Information Systems Development and the Changing Role of MIS in the Organisation*. *First New Zealand MIS Management Conference*. 1987. Wellington.
12. Wydenbach, G. and J. Paynter, *Software Project Estimation: a Survey of Practices in New Zealand.* New Zealand Journal of Computing, 1995. **6**(1B): pp. 317-327.
13. Bergeron, F. and J.-Y. St-Arnaud, *Estimation of Information Systems Development Efforts: A Pilot Study.* Information & Management, 1992. **22**: pp. 239-254.
14. Jørgensen, M., *A Review of Studies on Expert Estimation of Software Development Effort.* Journal of Systems and Software, 2004. **70**(1-2): pp. 37-60.

15. Woodward, H., *Project Management Institute practice standard for work breakdown structures*. 2001, Newton Square: Project Management Institute, Inc.
16. Tausworthe, R.C., *The Work Breakdown Structure in Software Project Management.* Journal of Systems and Software, 1980. **1**: pp. 181-186.
17. Jørgensen, M. and K. Moløkken-Østvold. *A Preliminary Checklist for Software Cost Management*. *QSIC 2003*. 2003. pp. 134-140.
18. Shepperd, M. and U. Passing. *An Experiment on Software Project Size and Effort Estimation*. *2003 ACM-IEEE International Symposium on Empirical Software Engineering (ISESE 2003)*. 2003. Frascati - Monte Porzio Catone (RM), ITALY: IEEE. pp. 120-129.
19. Engelkamp, S., S. Hartkopf, and P. Brossler. *Project experience database: a report based on first practical experience*. *International Conference on Product Focused Software Process Improvement*. 2000. Oulu, Finland. pp. 204-215.
20. Linstone, H.A. and M. Turoff, *The Delphi Method: Techniques and Applications*. 1975, London: Addison-Wesley.
21. Taff, L.M., J.W. Borcering, and W.R. Hudgins, *Estimeetings: Development estimates and a front end process for a large project.* IEEE Transactions on Software Engineering, 1991. **17**(8): pp. 839-849.
22. Gilb, T., *Principles of Software Engineering Management*. 1988: Addison-Wesley Publishing Company.
23. Graham, D.R., *Incremental Development and Delivery for Large Software Systems*, in *Software Engineering for Large Software Systems*, B.A. Kitchenham, Editor. 1990, Elsevier.
24. Cockburn, A., *In Search of Methodology*, *Object Magazine*. 1994. pp. 52-56.
25. Larman, C. and V.R. Basili, *Iterative and Incremental Development: A Brief History.* IEEE Computer, 2003(June): pp. 2-11.
26. Flyvbjerg, B., M.S. Holm, and S. Buhl, *Underestimating Costs in Public Works Projects - Error or Lie?* Journal of the American Planning Association, 2002. **68**(3): pp. 279-295.
27. DeMarco, T., *Rock and Roll and Cola War*, in *Why does Software Cost so Much?* 1995, Dorset House Publishing: New York.
28. Thomsett, R., *Double Dummy Spit and other Estimating Games.* American Programmer, 1996. **9**(6): pp. 16-22.

29. Grimstad, S., M. Jørgensen, and K. Moløkken-Østvold, *Software Effort Estimation Terminology: The Tower of Babel.* Submitted to Information and Software Technology, 2005.
30. Moløkken-Østvold, K. *Ethical Concerns when Increasing Realism in Controlled Experiments with Industrial Participants*. *Accepted for HICSS38 (Hawaii International Conference on System Sciences).* 2005. Big Island, Hawaii.
31. Moløkken-Østvold, K. and M. Jørgensen. *A Review of Surveys on Software Effort Estimation*. *2003 ACM-IEEE International Symposium on Empirical Software Engineering (ISESE 2003)*. 2003. Frascati, Monte Porzio Catone (RM), ITALY: IEEE. pp. 220-230.
32. Moløkken-Østvold, K., et al. *A Survey on Effort Estimation in Norwegian Software Industry*. *10th International Symposium on Software Metrics*. 2004. Chicago, Illinois, USA: IEEE Computer Society. pp. 208-219.
33. Moløkken-Østvold, K., et al. *Does Use of Development Model Affect Estimation Accuracy and Bias? Product Focused Software Process Improvement, 5th International Conference, PROFES 2004*. 2004. Kansai Science City, Japan: Springer (LNCS 3009). pp. 17-29.
34. Moløkken-Østvold, K., et al., *Avoiding Cost Overruns in Public Software Projects.* Submitted to Information and Software Technology, 2004.
35. Moløkken-Østvold, K. and M. Jørgensen, *The Impact of Development Model on Estimation Accuracy in Software Projects.* Submitted to IEEE Transactions on Software Engineering., 2004.
36. Cozby, P.C., *Methods in behavioral research*. 5th ed. 1993, Mountain View: Mayfield Publishing Company.
37. Moløkken-Østvold, K. and M. Jørgensen, *Expert Estimation of the Effort of Web-Development Projects: Are Software Professionals in Technical Roles More Optimistic Than Those in Non-Technical Roles?* Empirical Software Engineering, 2005. **10**(1): pp. 7-29.
38. Moløkken-Østvold, K. and M. Jørgensen, *Group Processes in Software Effort Estimation.* Empirical Software Engineering, 2004. **9**(4): pp. 315-334.
39. Edwards, J.S. and T.T. Moores, *A Conflict Between the Use of Estimating and Planning Tools in the Management of Information Systems.* European Journal of Information Systems, 1994. **3**(2): pp. 139-147.

40. Conte, S.D., H.E. Dunsmore, and V.Y. Shen, *Software Engineering Metrics and Models*. 1986, Menlo Park: Benjamin-Cummings.
41. Briand, L.C. and I. Wieczorek, *Resource modeling in software engineering*, in *Encyclopedia of Software Engineering*, J. Marciniak, Editor. 2001, Wiley.
42. Foss, T., et al., *A Simulation Study of the Model Evaluation Criterion MMRE.* IEEE Transactions on Software Engineering, 2003. **29**(11): pp. 985-995.
43. Jørgensen, M. and D. Sjøberg, *An effort prediction interval approach based on the empirical distribution of previous estimation accuracy.* Journal of Information and Software Technology, 2003. **45**(3): pp. 123-136.
44. Stensrud, E., et al. *An empirical validation of the relationship between the magnitude of relative error and project size*. *Eighth IEEE Symposium on Software Metrics.* 2002: IEEE. pp. 3-12.
45. Stensrud, E., et al., *A Further Empirical Investigation of the Relationship Between MRE and Project Size.* Empirical Software Engineering, 2003. **8**(2): pp. 139-161.
46. Kitchenham, B., et al., *What Accuracy Statistics Really Measure.* IEE Proceedings - Software Engineering, 2001. **148**(3): pp. 81-85.
47. Miyazaki, Y., et al., *Method to estimate parameter values in software prediction models.* Information and Software Technology, 1991. **33**(3): pp. 239-243.
48. Makridakis, S., *Accuracy Measures: Theoretical and Practical Concerns.* International Journal of Forecasting, 1993. **9**(4): pp. 527-529.
49. Moløkken-Østvold, K., et al., *Project Estimation in the Norwegian Software Industry - A Summary*. 2004, Simula Research Laboratory.
50. Moløkken-Østvold, K. and M. Jørgensen. *Software Effort Estimation: Unstructured Group Discussion as a Method to Reduce Individual Biases*. *The 15th Annual Workshop of the Psychology of Programming Interest Group (PPIG 2003)*. 2003. Keele, UK. pp. 285-296.

Paper I:

A Review of Surveys on Software Effort Estimation

Kjetil Moløkken-Østvold and Magne Jørgensen.
Simula Research Laboratory.

IEEE International Symposium on Empirical Software Engineering (ISESE 2003),
2003. September 30 - October 1, Rome, Italy. Page 223-230,
IEEE Computer Society. ISBN 0-7695-2002-2.

Abstract. *This paper summarizes estimation knowledge through a review of surveys on software effort estimation. Main findings were that: (1) Most projects (60-80%) encounter effort and/or schedule overruns. The overruns, however, seem to be lower than the overruns reported by some consultancy companies. For example, Standish Group's 'Chaos Report' describes an average cost overrun of 89%, which is much higher than the average overruns found in other surveys, i.e., 30-40%. (2) The estimation methods in most frequent use are expert judgment-based. A possible reason for the frequent use of expert judgment is that there is no evidence that formal estimation models lead to more accurate estimates. (3) There is a lack of surveys including extensive analyses of the reasons for effort and schedule overruns.*

Keywords: Effort estimation, surveys, software projects.

1. Introduction

The software industry's inability to provide accurate estimates of development cost, effort, and/or time is well known. This inability is described in reports from project management consultancy companies, case studies on project failures, articles in the computer press, and estimation surveys. The common belief seems to be that the cost overruns are very large, and we have experienced that few software professionals and researchers react with disbelief when being presented with the inaccuracy figures reported in Standish Group's Chaos Report [1], i.e., an average cost overrun of 89%. There may be several reasons for this attitude.

The introduction part of many research papers on software effort estimation includes references to the estimation studies with the most extreme findings. This may, in part, be due to the authors' need to legitimize their own research. Surveys, which report none, or small, overruns may not be cited as often, since they do not 'contribute' to defining effort estimation as a central problem area in software engineering.

The estimation accuracy results reported may in some cases be biased towards high inaccuracy, e.g., studies conducted by consultants selling estimation advice, journalists writing a story on project failures, or software houses who sell estimation tools.

It is difficult to get a balanced view on the software industry's estimation performance without unbiased information from a representative set of projects and organizations. The surveys presented in scientific journals and conferences may be a source for such unbiased information. This paper summarizes estimation results from surveys on software estimation. To our knowledge, there has not been conducted any structured review of the estimation surveys with the aim of summarizing our knowledge of software effort estimation performance. Besides summarizing results from estimation surveys, the main purposes of this paper are to challenge some common estimation beliefs that may be inaccurate, to discuss methodical aspects related to completion of industrial surveys, and, to support future surveys on software effort estimation.

In the remaining part of this paper, we present several research questions relevant to the field of software project estimation (Section 2). Then, we present an overview of relevant surveys (Section 3). The research questions are then discussed in light of the findings of the surveys (Section 4). Finally, we conclude and outline further research (Section 5).

2. Research questions

The rationale of this review of surveys is to summarize findings on central aspects of software estimation, to enable a more balanced estimation debate, and to suggest further research. The central estimation aspects are presented as research questions. We have only included questions addressed by more than one of the reviewed surveys. Our research questions (RQs) are:

RQ1: To what extent do software development projects deviate from the original plan, with regard to cost, schedule and functionality?

RQ2: Which methods are used to estimate software effort, and do these systematically differ in accuracy?

RQ3: How important is accurate effort estimation perceived to be, and to what extent is the level of accuracy considered a problem in the software industry?

RQ4: What are the main causes for software projects to deviate from their original plan?

Due to the limited number of surveys, the different research designs, and the methodological limitations of many of the surveys, we do not expect to discover definite answers to these questions. We do, however, hope to challenge some estimation myths and to enable a better foundation for further estimation research.

3. Estimation surveys

The surveys reviewed are listed in chronological order based on year of their first appearance, see overview in table 1.

The surveys are named after the first author of the paper in which the survey first appeared. The remaining part of this section provides a brief outline of the survey designs and research foci.

In order to avoid too much influence from one particular organization, case studies with just one subject is omitted in this review.

#	*Name*	*Year*	*Sample*	*Respondents*	*Response Rate*	*Country*	*Rnd. sample*	*Int.*
1	Jenkins	1984	N/A	72	N/A	USA	No	Yes
2	McAulay	1987	280	120	42.9%	New Zealand	No	No
3	Phan	1988	827	191	23.1%	USA	No	No
4	Heemstra	1989	2659	598	22.5%	Netherlands	No	No
5	Lederer	1991	400	112	28.0%	USA	No	No
6	Bergeron	1992	374	89	23.8%	Canada	No	No
7	Moores	1992	115	54	47.0%	UK	No	No
8	Standish	1994	N/A	365	N/A	USA	No	No
9	Wydenbach	1995	515	213	41.4%	New Zealand	No	No
10	Addison	2002	70	36	51.4%	South Africa	No	No

Table 1. Survey overview.

3.1. Survey 1: Jenkins

Jenkins, Naumann and Wetherbe [2] conducted a large empirical investigation in the beginning of the 1980s. The study focused on the early stages of system development. It included development aspects, such as user satisfaction, development time, and cost overruns. They interviewed managers from 23 large organizations and collected data on 72 projects. The average project cost was $103,000, and the average duration was 10.5 months. The study included projects that were considered small, medium and large relative to the organizations standards. A majority of the projects developed new software systems (55%), but redesign (33%) and enhancement (11%) of existing software systems were also represented. The survey measured three success factors; user satisfaction, being “on-time” and being “on-budget”.

3.2. Survey 2: McAulay

McAulay [3] conducted a survey on software metrics in New Zealand. This survey is unpublished, but some information is presented by Purvis, MacDonell et al. [4]. In order to

investigate information system development projects, questionnaires were sent to 280 organizations. Out of these, 120 were returned.

3.3. Survey 3: Phan

Phan et al. [5, 6] tried to asses to what extent, and for what reasons, software development projects encountered cost and schedule overruns. Questionnaires were sent out to 827 professionals, and they received 191 responses. The projects involved were fairly large, with an average of 102 person months.

3.4. Survey 4: Heemstra

Heemstra and Kusters [7-9] conducted a survey of cost estimation in Dutch organizations. The goal was to provide an overview of the state oft the art of estimation and controlling software development costs. They sent out 2659 questionnaires, and got responses from 598 organizations. Estimation methods, original project estimates and actual effort were analyzed.

3.5. Survey 5: Lederer

Lederer and Prasad [10-13] conducted a survey concerning software development cost estimates. Through a questionnaire, 112 software managers and other professionals (out of 400 possible) reported their views on a wide variety of cost estimation aspects. The respondents represented fairly large companies, with an average of 478 employees in the information systems department.

3.6. Survey 6: Bergeron

Bergeron and St-Arnaud [14] performed a study to identify estimation methods, and to what extent they were used. They also investigated how choice of method, and underlying factors and variables, influenced estimation accuracy. In total, 374 Questionnaires were sent to 152 organizations. The companies each received 1-4 copies of the questionnaire. The 89 responses received came from 67 different organizations. All projects included were larger than 150 person-days.

3.7. Survey 7: Moores

Moores and Edwards [15] sought to investigate why there was an apparent lack of use of software cost estimating tools. A total of 115 large UK corporations were approached, and they received 54 responses.

3.8. Survey 8: The Standish Group 'Chaos report'

Although not a strictly scientific survey, the Standish Group's 'Chaos report' seems to have made such a strong impact (perhaps more than any scientific survey) on common estimation beliefs that it deserves to be included in this review. The first and most cited version is the 'Chaos report' from 1994, but Standish Group has continued data collection during the nineties. Total sample size of the 1994-report was 365 respondents. All projects were classified as, success (delivered as planned), challenged (delivered over time, and over budget and with fewer than specified features) or impaired (cancelled). The sample selection process of organizations and projects is unknown (we have made several inquiries to the Standish Group about properties such as the sample involved. The have refused to provide these details, claiming them to be 'business secrets'.), along with other important design and measurement issues. It is possible that the estimation accuracy reported by Standish Group is misleading. For example, from our inspection of the survey questionnaire available on their web-site, it seems as projects completed ahead of plans had to be registered as projects with "less than 20% overrun". If our assumption is correct, this may have led to too high average cost estimation overrun values.

3.9. Survey 9: Wydenbach

Wydenbach and Paynter [16] investigated the estimation practices in New Zealand on basis of a previous survey [7]. They sent questionnaires to what was believed to be a representative sample of companies (515), and received 213 usable responses.

3.10. Survey 10: Addison

Addison and Vallabh [17] investigated the perceptions of project managers on software project risks and controls. A "snowball sample" (explained in Section 4.1) was used to identify 70 managers, of whom 36 returned the questionnaire. Although not a study with a focus on estimation, it reports on aspects related to budgets and plans.

4. Discussion

As seen in Section 3, the surveys are different in many aspects. This makes it challenging to compare, combine and present the results. On the other hand, the differences may also add value and insight to our research questions. This section provides a review of general properties and methodological aspects of the surveys and addresses the research questions described in Section 2.

4.1. Survey designs

The surveys addressed companies, projects, software professionals, or a combination of these. Unfortunately, none of the surveys involved a procedure to ensure random samples. There was a variety of sample selection methods, such as contacting members of a society, snowball sampling (encouraging first round survey participants to suggest second round survey participants, etc.) or deliberately targeting specific categories of companies (e.g., large). The lack of procedures to ensure random samples, and the lack of proper analyses of the population represented by the samples, may be a major problem with all of the surveys, potentially leading to difficulties when interpreting and transferring results. This, in combination with a low response rate, may have unwanted influence on the validity of the results. For those who have provided response rates, the rates ranged from 22,5% [7] to 51,4% [17].

Only one of the surveys (Bergeron) used personal interview as a data collection method [2]. The other studies mailed questionnaires to potential respondents. This may have influenced the quality of the data, since mailed questionnaires lessens the involvement and commitment of the participants [18], and make misinterpretations of answers and questions more likely. Another problem may be that many of the surveys relied on the managers being unbiased when, for example, reporting magnitude of and reasons for overruns. It is possible that managers may be biased towards under-reporting overruns and have a tendency to over-report customer issues as reasons for overruns.

Although the surveys span four continents, it is essential to observe that the surveys were conduction in only six, similar, western countries, i.e., we do not know much about the preferred estimation methods and their performance in other cultures.

There are also several possible, and relevant, research questions that we are unable to address on basis of the reviewed surveys. One such aspect is possible differences of project

properties based on the type of developing organization. Do, e.g., CMS (Content Management System)-projects differ from defence projects on choice of estimation methods? And how does in-house development versus development for customers affect estimation accuracy? Such topics have been addressed by other studies, e.g., [19]. The surveys presented in this paper also make such differentiations, to some extent.

The survey by Heemstra and Kusters [7] differentiates between in-house and other types of development. Some of the other surveys also distribute the organizations into different sectors [13-15]. However, none of the papers on the surveys report any differences between type of development projects, related to either choice of method or accuracy. They mainly report the distribution of industries in order to show that their survey included a diverse sample. For this reason, it was impossible to investigate possible differences due to project, application, development method or organization in this review.

An important aspect of project management is how to avoid project abortion and/or restarts. This may sometimes relate to effort estimation, but may also be caused by market changes, customer orders, company restructuring or similar reasons. None of the surveys presented relate the possibility of failure to estimation method, and it is therefore not treated in this review.

4.2. Survey results

The discussion in this section is structured around the research questions (RQs) presented in Section 2.

RQ1: To what extent do software development projects deviate from the original plan, with regard to cost, schedule and functionality?

The results from surveys investigating the frequency or magnitude of overruns are displayed in table 2.

Although the surveys report different results; the tendency is clear, a majority of the projects encounter overruns. Phan [6] included a report on overruns on organizational level that is not directly comparable with the other results in table 2. He found that cost overruns occurred always in 4%, usually in 37%, sometimes in 42%, rarely in 12%, and, never in 4% of the 191 organizations surveyed. Similarly, schedule overruns occurred always in 1%, usually in 31%, sometimes in 50%, rarely in 15%, and, never in 2% of the organizations.

Bergeron and St-Arnaud [14] found that 58% of the projects surveyed had cost overruns of more than 20%.

Study	*Jenkins*	*Phan*	*Heemstra*	*Lederer*	*Bergeron*	*Standish*
Cost overrun	34% (median)	33% (mean)			33% (mean)	89% (mean)
Projects completed over budget	61%		70%	63%		84%
Projects completed under budget	10%			14%		
Schedule overrun	22%					
Projects completed after schedule	65%		80%			84%
Projects completed before schedule	4%					

Table 2: Estimation accuracy results

Heemstra and Kusters [7] found that overruns increased with project size. Very large projects were defined as projects consuming more than 200 man-months. These projects had overruns of more than 10% in 55% of the cases. For all projects, overruns of 10% or more occurred in 28% of the sample. A similar tendency on larger overruns for larger projects was found by the Standish Group. In Jenkins' study, however, the occurrence of overruns was equally likely for small, medium, and large projects.

The degree of delivered functionality is difficult to measure, since it may be subject to differences in opinions. None of the surveys concerned the degree of delivered functionality opinions of the actual users, but according to the managers in the Jenkins study, 72% of the users was satisfied or very satisfied with the functionality [2]. Similarly, 70% of Phans respondents claimed that user requirements and expectations were usually met. The Standish Group [1], on the other hand, claimed that only 7.3% of the projects delivered all the functionality originally specified.

To summarize, it seems as if most projects (60-80%) are completed over budget and/or schedule. Most results also indicate that the percentage and magnitude of overruns increase as projects grow in size. The magnitude of overruns may, however, not be as dramatic as reported by Standish Groups' Chaos Report. Most surveys, e.g., [2, 6, 14], and other case studies [20, 21] suggest that a more likely average effort and cost overrun is between 30 and 40%.

RQ2: Which methods are used to estimate software effort, and do these systematically differ in accuracy?

It is difficult to compare the surveys that investigated estimation methods since none of them operated with the same categories of estimation methods. A further complicating factor is that the respondents may have interpreted pre-defined categories, e.g., analogy-based estimation, differently. We have grouped the estimation methods in three main categories: Expert judgment-based methods, model-based methods and "other". Model based estimates include formal estimation models such as COCOMO, Use-Case-based estimation, FPA-metrics or other algorithm driven methods. In the category of "other" there are methods that are not "pure" estimation methods, e.g., capacity related and price-to-win-based methods, and methods than can be used in combination with other models (top-down and bottom-up). An overview is presented in Table 3. An 'X' in the table indicates this alternative was not an option in the survey. In the McAulay column, we have joined three different software cost model method alternatives of that study. The original study found that 11% applied Function Point Analysis, 2% lines of code based models and 0% Halstead Metrics.

		McAulay (n=114)	*Heemstra (n=369)*	*Wydenbach (n=209)*	*Bergeron (n=89)*
		Percentage used			
Estimation Methods		*(more than one method possible)*			*Importance (1-4)*
Expert based	**Expert cons.**	X	26%	86%	1.8
	Intuition and exp.	85%	62%	X	3.3
	Analogy	X	61%	65%	2.5
Model based	**Cost Models**	13%	14%	26%	1.3
Other	**Price-to-win**	X	8%	16%	1.2
	Capacity related	X	21%	11%	1.6
	Top-down	X	X	13%	1.4
	Bottom-up	X	X	51%	2.4
	Other	12%	9%	0%	1.5

Table3: Estimation methods results

A problem with Table 3 is the overlapping of categories. For example, one could argue that "estimation by analogy" and "top-down" in many cases are two aspects of the same estimation method. How the respondents interpreted the estimation method categories is not possible to derive from the survey results.

The respondents were not asked about the extent of use of one method in the organization, only about whether an estimation method was used or not. This means that an estimation method used to estimate very few projects per organization gets a too high percentage in Table 3.

It is also essential to note that projects may be estimated by a combination of two or more different methods, e.g. model and expert-based. Such combination has been advocated in other studies [22]. To which extent such combination methods were used was not addressed by any of the surveys. We were therefore unable to draw conclusions on a possible beneficial effect of combining methods in this review.

'Price-to-win' is listed as an estimation method, but we believe that most managers would not report this as an estimation method even if "price-to-win" had an impact on their estimates. This may be the case because mangers are not aware of the effect of customer expectations on effort estimates [23]. They may not feel that that it is an estimation method, or they believe that they should not be impacted by the "price-to-win".

Lederer and Prasad [10] applied a different approach. Instead of asking what methods were used, they asked how estimates were influenced. Respondents rated alternatives on a five point Likert scale (max=5). The results are displayed in Table 4.

	Response Categories	*Mean*
1	Comparison to similar, past projects based on personal memory	3.77
2	Comparison to similar, past projects based on documented facts	3.41
3	Intuition	3.38
4	A simple arithmetic formula (such as summing task durations)	3.09
5	Guessing	2.76
6	Established standards (such as averages, standard deviations, etc.)	2.33
7	A software package for estimating	1.80
8	A complex statistical formula (such as multiple regression, differential equations, etc.)	1.49

Table 4: Estimation responses ranked by importance

Responses 1, 2, 3 and 5 seem to be expert judgment-based, responses 7 and 8 to be model-based, while responses 4 are 6 are more difficult to interpret, i.e., the expert judgment-based methods seem to be more important in this survey as well.

The results from all identified surveys point in one direction: Expert estimation is by far the preferred method for software estimation. This is further backed by a variety of case studies not included in this review [21, 24-26]. In fact, we have not been able to identify a single study reporting a dominant use of formal estimation methods.

Cultural similarities between the samples or organizations may, however, limit the transfer of the results to other cultures. Perhaps surveys conducted in China, India, or Germany would have yielded a different outcome.

Heemstra and Kusters [7] found that estimation accuracy did not improve when formal models were used. In fact, projects estimated with Function Point Analysis had larger overruns than the other projects. Similarly, Bergeron and St-Arnaud [14] found that price-to-win methods and algorithmic models were associated with less accurate estimates. The methods associated with the most accurate estimates were based on analogy and expert opinion. However, this may be coincidental since few respondents used price-to-win and algorithmic methods.

On a general basis, it is important to be aware of there are several aspects that may affect choice of estimation methods. It may be the case that especially challenging and/or large projects more often rely on formal estimation methods, or that the projects applying, e.g., algorithmic estimation methods may be different from the other projects. None of the surveys address this possibility.

The lack of evidence that estimation models are more accurate than expert judgment [27], may be an important reason for the widespread use of expert estimation. In addition, expert estimation may have the advantage of being easy to use and flexible.

RQ 3: *How important is accurate effort estimation perceived to be, and to what extent is the level of accuracy considered a problem in the software industry?*

The only survey that directly addressed RQ 3 was conducted by Lederer and Prasad [11]. On a five point Likert scale, the average importance rating reported by managers was 4.17. Although that survey found that managers perceived estimation as important; this does not necessarily imply that projects as a rule are estimated. We found three surveys on the proportions of software organizations that estimated costs. Heemstra [9] reports that 65%

and McAulay [3] report that 95% of the organizations as a rule estimate projects. Wydenbach and Paynter [16] report that 88% of the companies estimate at least half of their projects.

One survey studied the estimation percentage at project level [11]. They found that 87% of the companies' large projects were estimated.

Moores and Edwards [15] found that 91% of the responding managers answered 'yes' to the question 'do you see estimation as a problem?', while only 9% answered 'no'. An interesting finding is that the accepted level of estimation accuracy was typically +/- 20%. A finding supporting the belief that estimation is perceived as a problem was reported by Addison and Vallabh [17]. They found that the risk factor viewed as most problematic by software professionals was 'unrealistic schedules and budgets'.

Combined, these surveys indicate an awareness of estimation as a problem and an important activity. There are, however, many other important software development activities, e.g., contract negotiations with customers. An appropriate approach to reveal how important the companies regard effort estimation accuracy in practice is to investigate the organizations actual focus on improving estimation processes and effort spent to achieve accurate estimates. We have found no such studies.

RQ4: What are the main causes for software projects to deviate from their original plan?

A problem when analyzing reasons for project overruns is that the respondents may be biased and/or affected by selective memory. Ideally, estimation accuracy reviews should be conducted by uninvolved reviewers to, for example, avoid the "blame the others"-attitude. The results on reasons for project overruns presented here should therefore be interpreted relative to the role and perspectives of the respondents, typically project managers.

In the survey by Phan [6], the participants were asked for the reasons why projects had cost or schedule overruns. The respondents believed that cost overruns were most often caused by over-optimistic estimates (51%), closely followed by changes in design or implementation (50%). The reasons for schedule overruns were optimistic planning (44%), followed by frequent major (36%) and minor (33%) changes from the original specifications.

In Lederer and Prasad's study [11] the respondents rated 24 possible reasons for inaccurate estimates on a five point Likert scale. The top rated causes were 'Frequent

requests for changes by users' (3.89), 'Users' lack of understanding of their own requirements' (3.60) and 'Overlooked tasks' (3.60), i.e., the problems with the users were the two most important reasons for estimation inaccuracy.

In sum, over-optimistic estimates and user changes or misunderstandings were all important reasons for project overruns, from the perspective of the managers of the software provider organization.

Over-optimism does not necessarily describe properly what happens when a cost estimate is too low. For example, we have experienced that many projects initially have realistic cost estimates. Then, the management believes that the estimates are unacceptable high and put a pressure on the estimators to reduce the estimates, i.e., it may not be over-optimism but cost reduction pressure from customer or management that lead to estimates reported as "over-optimistic".

5. Conclusions and further studies

Our search for, and review of, surveys suggest that there are few that are directly related to software effort estimation. Also, the design of these surveys often make transfer of results problematic, e.g., non-random samples, low response rates, and frequent use of data collection techniques (questionnaires) potentially leading to low data quality.

The following observations, derived from the surveys, should therefore be interpreted carefully:

- Expert estimation is the method in most frequent use. There is no evidence that the use of formal estimation methods on average leads to more accurate estimates.
- Project overruns are frequent, but most projects do not suffer from major overruns. The average cost overrun reported by Standish Group's Chaos Report (89%) is not supported by other surveys. An average cost overrun of 30-40% seems to be the most common value reported.
- Managers state that accurate estimation is perceived as a problem.
- The reasons for overruns are complex, and not properly addressed in software estimation surveys. For example, software managers may have a tendency to over-report causes that lies outside their responsibility, e.g., customer-related causes.

Forthcoming surveys should seek to investigate more thoroughly on several aspects related to when, how and why estimation methods are chosen. In this way we can learn how choice of methods, or combinations of methods, may be influenced by properties such as

project type or size. Only when such information is provided may we be able to compare the level of accuracy of different methods. Perhaps we may be able to find more about topics like when and how to combine different methods.

It would also be interesting to know what level of estimation accuracy managers are satisfied with, and how this varies depending on type of development.

We are currently conducting a large in-depth survey of Norwegian companies and projects, were we, amongst others, address the research questions treated in this paper and aim to compare the current situation in Norway with previous estimation surveys. A goal of that survey is to find out how the different projects actually are estimated. For example, if a project applies an estimation model, we will find out how the model is used. Do the experts adjust the output from the model? And when a project applies expert estimation, is this done with the aid of checklist or experience databases, and is it based on a combination of different experts' estimates? The survey also tries to obtain the respondents interpretation of 'estimate'. Is it interpreted as the most likely effort, the price-to-win, or something else? In order to obtain knowledge of these, and other central aspects of software estimation, personal interviews with both top management and the project managers are employed.

Acknowledgements

This research was funded by the Research Council of Norway under the project INCO.

References

1. Standish, G., *The Chaos Report*. 1994, The Standish Group.
2. Jenkins, A.M., J.D. Naumann, and J.C. Wetherbe, *Empirical Investigation of Systems Development Practices and Results.* Information & Management, 1984. **7**: pp. 73-82.
3. McAulay, K. *Information Systems Development and the Changing Role of MIS in the Organisation*. *First New Zealand MIS Management Conference*. 1987. Wellington.
4. Purvis, M.K., S.G. MacDonell, and J. Westland, *Software Metrics in New Zealand: Recent Trends.* New Zealand Journal of Computing, 1994. **5**(2): pp. 13-21.
5. Phan, D., D. Vogel, and Nunamaker, *The Search for Perfect Project Management*, *Computerworld*. 1988. pp. 95-100.

6. Phan, D., *Information Systems Project Management: an Integrated Resource Planning Perspective Model*, *Department of Management and Information Systems*. 1990, Arizona: Tucson.
7. Heemstra, F.J. and R.J. Kusters. *Controlling Software Development Costs: A Field Study*. *International Conference on Organisation and Information Systems*. 1989. Bled, Yugoslavia. pp. 652-664.
8. Heemstra, F.J. and R.J. Kusters, *Function point analysis: Evaluation of a software cost estimation model.* European Journal of Information Systems, 1991. **1**(4): pp. 223-237.
9. Heemstra, F.J., *Software cost estimation.* Information and Software Technology, 1992. **34**(10): pp. 627-639.
10. Lederer, A.L. and J. Prasad, *Nine management guidelines for better cost estimating.* Communications of the ACM, 1992. **35**(2): pp. 51-59.
11. Lederer, A.L. and J. Prasad, *Information systems software cost estimating: a current assessment.* Journal of Information Technology, 1993(8): pp. 22-33.
12. Lederer, A.L. and J. Prasad, *Causes of Inaccurate Software Development Cost Estimates.* Journal of Systems and Software, 1995(31): pp. 125-134.
13. Lederer, A.L. and J. Prasad, *The Validation of a Political Model of Information Systems Development Cost Estimating.* Computer-Personnel, 1991. **13**(2): pp. 47-57.
14. Bergeron, F. and J.-Y. St-Arnaud, *Estimation of Information Systems Development Efforts: A Pilot Study.* Information & Management, 1992. **22**: pp. 239-254.
15. Moores, T.T. and J.S. Edwards, *Could Large UK Corporations and Computing Companies Use Software Cost Estimating Tools? - A Survey.* European Journal of Information Systems, 1992. **1**(5): pp. 311-319.
16. Wydenbach, G. and J. Paynter, *Software Project Estimation: a Survey of Practices in New Zealand.* New Zealand Journal of Computing, 1995. **6**(1B): pp. 317-327.
17. Addison, T. and S. Vallabh. *Controlling Software Project Risks - an Empirical Study of Methods used by Experienced Project Managers. SAICSIT 2002.* 2002. Port Elizabeth, South Africa. pp. 128-140.
18. Cozby, P.C., *Methods in behavioral research*. 5th ed. 1993, Mountain View: Mayfield Publishing Company.
19. Maxwell, K.D. and P. Forselius, *Benchmarking Software Development Productivity.* IEEE Software, 2000. **17**: pp. 80-88.

20. Boehm, B., *Software Engineering Economics*. 1981, Englewood Cliffs, NJ: Prentice-Hall.
21. Kitchenham, B., et al., *An Empirical Study of Maintenance and Development Estimation Accuracy.* Journal of systems and software, 2002. **64**: pp. 55-77.
22. Höst, M. and C. Wohlin. *An Experimental Study of Individual Subjective Effort Estimations and Combinations of the Estimates*. *Proceedings the 20th International Conference onSoftware Engineering*. 1998. Kyoto, Japan. pp. 332-339.
23. Jørgensen, M. and D.I.K. Sjøberg, *The impact of customer expectation on software development effort estimates.* International Journal of Project Management, 2004. **22**(4): pp. 317-325.
24. Jørgensen, M. *An empirical evaluation of the MkII FPA estimation model*. *Norwegian Informatics Conference*. 1997. Voss, Norway: Tapir, Oslo. pp. 7-18.
25. Hill, J., L.C. Thomas, and D.E. Allen, *Experts' estimates of task durations in software development projects.* International Journal of Project Management, 2000. **18**(1): pp. 13-21.
26. Hihn, J. and H. Habib-Agahi. *Cost estimation of software intensive projects: A survey of current practices*. *International Conference on Software Engineering*. 1991. pp. 276-287.
27. Jørgensen, M., *A Review of Studies on Expert Estimation of Software Development Effort.* Journal of Systems and Software, 2004. **70**(1-2): pp. 37-60.

Paper II:

A Survey on Software Estimation in the Norwegian Industry

Kjetil Moløkken-Østvold[1, 2], Magne Jørgensen[1], Sinan S. Tanilkan[2],
Hans Gallis[1, 2], Anette C. Lien[1], and Siw E. Hove[1].
[1] Simula Research Laboratory, [2] Department of Informatics, UiO.

10th International Symposium on Software Metrics. 2004. Chicago, Illinois, USA:
IEEE Computer Society. pp. 208-219

Abstract. *This paper provides an overview of the estimation methods that software companies apply to estimate their projects, why those methods are chosen, and how accurate they are. In order to improve estimation accuracy, such knowledge is essential. We conducted an in-depth survey, where information was collected through structured interviews with senior managers from 18 different companies and project managers of 52 different projects. We analyzed information about estimation approach, effort estimation accuracy and bias, schedule estimation accuracy and bias, delivered functionality and other estimation related information. Our results suggest, for example, that average effort overruns are 41%, that the estimation performance has not changed much the last 10-20 years, that expert estimation is the dominating estimation method, that estimation accuracy is not much impacted by use of formal estimation models, and that software managers tend to believe that the estimation accuracy of their company is better than it actually is.*

1. Introduction

Over the past two decades, several research surveys have focused on software project effort and schedule estimation [1-10]. This is important, since an unbiased insight is essential in order to help the industry to make more accurate estimates. Essential data to obtain is, e.g., how estimates are made, what factors motivate the choice of estimation methods and the current level of estimation accuracy.

This paper attempts to provide an assessment of the current situation related to software effort estimation in Norway. Most of the previous surveys were conducted in the eighties and early nineties. Types of hardware, programming languages, business relations, development processes, clients and software companies have changed a lot since then. Further, the surveys were conducted in larger countries than Norway, such as the United States and the United Kingdom.

Section 2 provides an account of the previous research that motivated our research questions, which are presented in Section 3. Section 4 describes the methods used to collect and analyze the data presented in Section 5. Section 6 provides a discussion, which is summarized in Section 7.

2. Previous Surveys

The surveys conducted over the past 20 years have had varying areas of interest; some of them were conducted at company level, others at project level. As presented in a recent review [11], those areas that have attracted most attention are how companies estimate projects, how important effort estimation is perceived and what their current level of accuracy is. In this paper, we include only previous research that has been subject to peer review, or in which the research method is, at least partially, described. For that reason, the frequently quoted CHAOS Report published by the Standish Group is excluded. We discuss the validity problems of that report in [12].

2.1. Frequency and magnitude of effort and schedule overruns

The topic of estimation accuracy has been surveyed by Jenkins [3], Phan [8], Bergeron and St-Arnaud [1], Heemstra and Kusters [13], Lederer and Prasad [14, 15] and Sauer and

Cuthbertson [10]. These studies have addressed either the frequency of overruns, or the average estimation accuracy, or both.

A summary of some of the results from these surveys is displayed in Table 1. A blank space in the table indicates that this information was not reported in the survey.

Study (first author)	Jenkins [3]	Phan [8]	Heemstra [2]	Lederer [14]	Bergeron [1]	Sauer [10]
Year of study's first publication	1984	1988	1989	1991	1992	2003
Cost overrun	34% (md)	33% (mean)			33% (mean)	18% (mean)
Project used more than estimated effort	61%		70%	63%		59%
Project used less than estimated effort	10%			14%		15%
Schedule overrun	22% (mean)					23% (mean)
Project completed after schedule	65%		80%			35%
Project completed before schedule	4%					3%

Table 1: Previous surveys on estimation accuracy

These surveys indicate that most projects (60-80%) are completed over estimated effort and/or schedule. The magnitude of effort overruns reported in most of the surveys [1, 3, 8] is between 30 and 40%. This has also been supported by several case studies, e.g. [16, 17].

Sauer and Cuthbert found a lesser magnitude of overruns [10], but this may have been affected by the self-selecting sample in that survey. Bergeron and St-Arnaud [1] found that 58% of the projects surveyed had cost overruns of more than 20%. Phan [8] reports on overruns at an organizational level that is not directly comparable with the other surveys. He found that cost overruns occurred always in 4%, usually in 37%, sometimes in 42%, rarely in 12%, and, never in 4% of the 191 organizations surveyed. Similarly, schedule overruns occurred always in 1%, usually in 31%, sometimes in 50%, rarely in 15%, and, never in 2% of the organizations.

2.2. Choice of estimation method

In our review [11], we found five different surveys that addressed the choice of estimation method [1, 2, 5, 18, 19]. Due to the different metrics used, and the different

categorizations, it is difficult to compare the surveys. A further complicating factor is that the respondents may have interpreted pre-defined categories differently. We have grouped the estimation methods into three main categories: expert judgment-based methods, model-based methods and "other". Model-based methods include formal estimation models such as COCOMO, Use-Case-based estimation, FPA-metrics or other algorithm driven methods. In the category of "other" there are methods that are not "pure" estimation methods, e.g., capacity-related and price-to-win-based methods, and methods than can be used in combination with other models (top-down and bottom-up). An overview is presented in Table 2. A blank space in the table indicates that this alternative was not an option in the survey.

It is essential to note that the projects surveyed may have been estimated by a combination of two or more different methods, e.g. model and expert-based. The extent to which such combination methods were used was not reported in any of the surveys. We were therefore unable to draw conclusions about the possible beneficial effect of combining methods in the review.

		McAulay	*Heemstra*	*Wydenbach*	
Study (first author)		*[5]*	*[2]*	*[19]*	*Bergeron [1]*
Year of study's first publication		*1987*	*1989*	*1995*	*1992*
Estimation Methods		*Percentage used (more than one method possible)*			*Importance (1-4, 4=most important)*
Expert based	**Expert consultation**		26%	86%	1.8
	Intuition and experience	85%	62%		3.3
	Analogy		61%	65%	2.5
Model based	**Software Cost Models**	13%	14%	26%	1.3
Other	**Price-to-win**		8%	16%	1.2
	Capacity related		21%	11%	1.6
	Top-down			13%	1.4
	Bottom-up			51%	2.4
	Other	12%	9%	0%	1.5

Table 2: Choice of estimation method

'Price-to-win' is listed as an estimation method, but we believe that most managers would not report this as an estimation method even if "price-to-win" had an impact on their

estimates. This may be the case because managers are not aware of the effect of client expectations on effort estimates [20]. They may feel that it is not an estimation method, or may believe that they should not be affected by the "price-to-win".

Lederer and Prasad [18] applied a different approach from the surveys displayed in table 2. Instead of asking what methods were used, they asked how estimates were influenced. Respondents rated alternatives on a five-point Likert scale (min=1, max=5). They had eight different categories, and the alternative "Comparison to similar, past projects based on personal memory" scored highest (average 3,77), while the alternative "A complex statistical formula (such as multiple regression, differential equations, etc.)" scored lowest (average 1.49).

2.3. How important is estimation accuracy perceived?

The only survey that directly addressed the importance of estimation accuracy was conducted by Lederer and Prasad [15]. On a five-point Likert scale (min=1, max=5), the average importance rating reported by managers was 4.17. Although that survey found that managers perceived estimation as important; this does not entail that projects are estimated as a matter of course.

Related to this finding are surveys on the proportions of software organizations that estimate costs. Heemstra [2] reports that 65% and McAulay [5] report that 95% of the organizations estimate projects as a rule. Wydenbach and Paynter [19] report that 88% of the companies estimate at least half of their projects. The survey by Lederer and Prasad studied the estimation percentage at project level [15]. They found that 87% of the companies' large projects were estimated.

Moores and Edwards [6] found that 91% of the responding managers answered 'yes' to the question 'do you see estimation as a problem?', while only 9% answered 'no'. An interesting finding is that the managers reported that the accepted level of estimation accuracy was typically +/- 20%.

A finding that supports the belief that estimation is perceived as a problem was reported by Addison and Vallabh [21]. They found that the risk factor viewed as most problematic by software professionals was 'unrealistic schedules and efforts'.

In combination, these surveys indicate awareness that estimation is both a problem and an important activity. There are, however, many other important software development activities, e.g., contract negotiations with clients. An appropriate approach to reveal how

important the companies regard effort estimation accuracy in practice is to investigate the organizations' actual focus on improving estimation processes and the effort expended on achieving accurate estimates. We have found no such studies.

3. Research Questions

In our survey, we wanted to investigate the same questions as in previous surveys, with some additions. While other surveys have focused on either company or project level, we wanted to focus on both, in order to compare responses.

Our topics of interest are divided into several research questions.

RQ1: What is the frequency and magnitude of effort estimation overruns?

RQ2: What is the frequency and magnitude of schedule estimation overruns?

RQ3: Does project size affect effort estimation accuracy or bias?

Choice of estimation method has been related to the study of effort estimation accuracy. There have been research results "in favor" of different estimation methods (e.g., expert-based vs. model-based), but according to a recent review, such evidence is inconclusive [22]. In our survey, we also wanted to investigate the potential benefits of combining different estimation approaches. Such combination has been advocated in other studies [1, 23].

RQ4: To what extent are different estimation methods (expert-based, model-based or combinations) used in the industry?

RQ5: Does choice of estimation method affect effort estimation accuracy or bias?

Many of the previous surveys, e.g. [13, 18] have been aimed at in-house development in large corporations that deal in insurance, banking, manufacturing etc. In our survey, we wanted to address the situation in consultancy companies that develop tailored solutions for clients, as well as in-house development in other software and/or telecommunications companies.

RQ6: Are there differences in estimation accuracy or bias between companies that develop projects internally and those that develop for clients?

An important aspect of effort estimation is how senior managers perceive the situation. If they are unaware of any problems, or if they believe that the problem is within an acceptable range, such as +/- 20% [6], it is unlikely that measures will be taken. We also wanted to investigate the possible reasons for selecting a particular method, as well as investigating how important effort estimation is perceived.

RQ7: How do senior managers perceive the company's level of estimation skill?

RQ8: On what basis is an estimation method selected?

RQ9: How important does the companies perceive estimation as being, in comparison with other aspects of development?

4. Method

The survey was conducted in Norway from February to November 2003.

4.1. The participating companies

In order to ensure a representative sample, stratified random sampling [24] from the population of Norwegian software development companies was used. The companies were categorized into different strata based on revenue and number of employees. This was based on different Norwegian sources, e.g. [25]. This is a reasonable approach, since we were going to investigate a limited number of companies and wanted to ensure that we had companies that represented different types of organization, such as software houses developing products for the mass market, contractors who develop for clients and the internal development departments of large companies. We also wanted companies of different sizes, both small (<10 employees), medium (between 25 and 100 employees) and large (>100 employees).

Each organization was contacted by phone and the study presented to them. If they agreed to participate, they were given time to prepare before they were visited by our researchers.

The unit of investigation was either the entire company or a specific department (the latter in the case of very large organizations with more than 1000 employees). We will, however use the term company for our unit of research in this paper. The companies that participated had between 10 and 750 employees, with an average of 141. Five of the companies developed projects to be used in-house, while two developed products for sale to the mass market. Out of the eleven companies who developed solutions for clients, nine had mainly private clients, while two had mainly public clients.

It is essential to note that the market situation in Norway at the time of the survey (2003) was very competitive, and many companies sustained losses. Senior managers, however, said that they coped with the situation by lowering their hourly rates (for those developing for clients) instead of underestimating projects.

About half of the companies' development was new projects (57.1%), while the rest was maintenance/re-engineering (38.5%) or combination projects (4.4%). The percentage of employees who owned shares or stock options in their own company ranged from zero to 100 percent, with an average of 43.5 %. For those companies that used contracts, either with clients or internally, most relied on fixed price (52.5%), while 22.5% were paid per hour and 25% were based a combination of these alternatives.

For the specific projects, the mean effort was 3124.5 man-hours, while the median was 1175 man-hours.

4.2. Data collection and analysis

In all companies we interviewed one or two senior manager(s) at company or, in the case of large corporations, department level. These managers were asked general questions about such matters as staff size, type of assignments, project resolution, estimation method and assessment of estimation accuracy.

We also interviewed the project managers (one or two for each project) of 52 different projects. These projects were selected by the companies themselves. To avoid selection bias, the companies were asked to submit their most recently completed projects. The only criteria were that they had to contain a workload of at least 100 man-hours. This was done in order to ensure that no small change requests and other one-developer projects were

submitted. This is in line with previous surveys, in which "trivial tasks routinely handled without effort estimation" were also filtered out [18].

We collected data via personal interviews, which yields data of high quality and ensures that ambiguities are resolved [24]. This was especially important in our survey, since there may be variations in the use of, and interpretations of, terms related to estimation methods. It also allows the respondents to add valuable information that it is not possible to include when completing a predefined questionnaire. Another point in favor of this approach is that our personal involvement indicates a seriousness of intent to the participants, and this may increase the likelihood of obtaining serious contributions from them. The main limitation of the approach is that it is time-consuming and hence prevented us from investigating as many companies and projects as would be possible by using mailed questionnaires. Out of the previous surveys, presented in Section 2 and the recent review [11], only one used personal interview as an approach [3].

Each interview lasted between 30 and 70 minutes. All researchers signed a confidentiality agreement at each company. The respondents were informed that their responses were anonymous, and that no feedback about the respondents' answers was to be reported to outsiders or to company managers.

All interviews were taped. Following data collection, results from the questionnaires and tapes were entered into databases and processed by independent personnel who had no stake in the research. This is especially important, because it ensures that there are no biases regarding how possibly ambiguous data, such as the estimation approaches, are classified.

In cases where the participants reported that they followed a company-defined estimation approach, we asked them to provide thorough descriptions. After all of the interviews had been conducted, the independent analyst listened to the tapes and categorized the customized methods according to the predefined categories presented in previous sections (model/expert/combination).

In order to assess estimation accuracy, both related to effort and schedule, the Balanced Relative Error (BRE) [26, 27] was used. It is calculated as:

$$BRE = \frac{|x - y|}{\min(x, y)}, \quad \text{x = actual and y = estimated.} \qquad (1)$$

The BRE is different from the more common MRE (Magnitude of Relative Error) measure [28]. MRE is calculated as:

$$MRE = \frac{|x - y|}{x}, \quad \text{x = actual and y = estimated value.} \qquad (2)$$

Even though MRE is the most widely used measure of estimation accuracy [29], one must be aware that it has unfortunate properties [26, 30]. The main concern for our case is the fact that underestimated and overestimated projects are weighted unevenly. The BRE, as its name indicates, is a more balanced measure.

However, it is important to note that the difference between using MRE and BRE does not have a large impact on statistical tests based on medians, which are mainly used in this paper.

Estimation bias is calculated using a similar equation. The only difference is that the absolute value is not used in order to reveal an eventual direction of inaccuracy.

$$BREbias = \frac{(x - y)}{\min(x, y)}, \quad \text{x = actual and y = estimated value.} \qquad (3)$$

In previous surveys in which estimation accuracy has been calculated, other measures have been used. Bergeron and St-Arnaud [1], used a formula recommended by Conte et al.[28], which is calculated as:

$$Accuracy = \frac{|x - y|}{y}, \quad \text{x = actual and y = estimated value.} \qquad (4)$$

They argue that this measure is more meaningful, since profit or loss should be calculated on the basis of expected cost by most project managers.

The wording in the paper reporting Phan's survey, e.g., "Software projects, on average, overrun planned costs by 33%" [31], implies a use of that accuracy measure. It is also possible that it is used in the survey conducted by Jenkins, which describe vague terms, such as "under-run and over-run", with no indications of how these were calculated [3].

4.3. What is an estimate?

Even though all previous surveys on software estimation have differences in method, and study different aspects of the process, they tend, with some exceptions [1], to treat a

software estimate as a single fixed value. During the course of our research, however, we have noticed that software projects often have several estimates [32]. This aspect, and how this poses challenges to estimation models, has also been addressed by Edwards and Moores [33].

First of all, the estimate often changes over the course of a project, depending on the *stage* at which the estimate is made. For example, a project can have an early estimate, based on vague requirements, a planning estimate based on a detailed requirement specification, and one (or more) re-estimates during the course of development. These estimates, all for the total effort of the project, may or may not be entirely different in magnitude.

Second, a very important factor is who the estimate is for, or communicated to. A single project may have two estimates at the planning stage: one that is used internally in the project team, e.g., by project managers, and another that is used for clients, whether they are internal or external.

The problem becomes even clearer when we find papers that describe bidding strategies, such as "price to win" as an estimation method! This was been done in several textbooks and research papers.

For this reason, it is obvious that one project can operate with, two, three or even more different estimates during development. This may often be unproblematic for developers and project managers who differentiate between these measures, but pose a significant challenge to scientific researchers.

When conducting research on the accuracy of estimates of software projects, it is necessary to differentiate between different types of estimates. What estimate(s) to use depends on the focus of research. If the goal is to investigate the estimation accuracy of professionals in a company, as it is in this paper, it is meaningful to use the *most likely* estimates at the *planning stage*, instead of, for example, early estimates communicated to clients. The latter may be affected by factors that have nothing to do with estimation skill, such as market competition.

In our survey, we collected data on all available estimates for each project. Some had only one, while others had as many as six different estimates. Nonetheless, this is not problematic when taken into account at the time of data analysis.

5. Results

The results are structured according to the research questions presented in Section 3. We include both responses from senior managers at company level and the responses from the project managers, based on specific projects. The responses from the senior managers are often at a general level, and reflect attitudes, beliefs and prioritizations. The data from specific projects, on the other hand, are based on recorded data submitted by the project managers. An overview of the project data is displayed in Appendix I.

The required data was available in 44 out of the 52 projects we analyzed. The other projects were discarded because they involved fewer than 100 man-hours (1), or most estimation and development work had been done by outside consultants (2), or the project managers had not kept accurate track of estimated and/or actual effort (5).

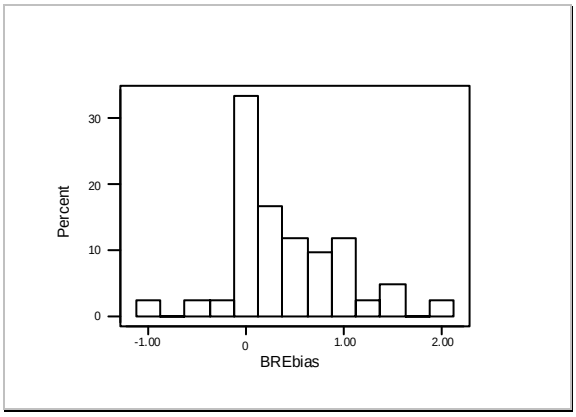
Of the 44 projects whose inclusion in the analysis was meaningful, two (5%) was aborted without being completed, and five (11%) were completed on time, on estimated effort and with required functionality. The rest (84%) were challenged with respect to effort overruns, schedule overruns, functionality, or a combination of these (see Appendix I).

5.1. What is the frequency and magnitude of effort estimation overruns? (RQ1)

Out of the 42 projects that were completed, 32 (76%) had effort overruns. Two projects (5%) ended up on target, while eight (19%) used less effort than indicated.

The estimation bias (BREbias) ranged from -0.88 (estimated effort 88% higher than actual effort) to 1.91 (actual effort almost three times the estimated effort). The distribution of project effort estimation bias is shown in Figure 1.

Figure 1: Distribution of effort estimation BREbias



The average effort BREbias was 0.41, while the median effort BREbias was 0.21. This corresponds to a mean cost overrun of 41%, while the median cost overrun was 21%.

5.2. What is the frequency and magnitude of schedule estimation overruns? (RQ2)

Regarding schedule, 26 projects (62%) had overruns, while 15 (36%) were completed on schedule, and one (2%) before schedule. Mean schedule overrun was 25% (BREbias 0.25), while the median overrun was 9% (BREbias 0.09).

5.3. Does project size affect effort estimation accuracy or bias? (RQ3)

To see whether project size could affect effort estimation accuracy, we divided the 42 projects into halves, based on actual effort. Mean BREbias for the 21 largest and the 21 smallest projects were 0.52 and 0.30, respectively. An Anderson-Darling test reveals that the samples are not normally distributed, but since the samples sizes are equal (both 21) and the variance is similar [34], a statistical t-test was used to examine the difference in mean accuracy values between large and small projects. The results for accuracy (BRE) and bias (BREbias) are displayed in Table 3.

	Large	*Small*	*p-value*
BRE	0.52	0.46	0.34
BREbias	0.52	0.30	0.11

Table 3: Effort estimation accuracy and bias by project size.

As indicated in the table, the large projects may be more prone to under-estimation (higher BREbias).

5.4. To what extent are different estimation methods used in the industry? (RQ4)

When asked about the kinds of estimation approaches that were used in the projects, 13 of the companies answered that they relied 100% on expert estimation. Three of the companies used a combination of expert judgment and estimation model 100% of the time, while two companies sometimes used expert estimation and sometimes a combination of expert and model. Out of the five companies that used a combination of model and expert, four of the companies used use-case based estimation models [35-37].

Of the 44 projects we analyzed (including two that were aborted), 37 (84%) of the managers reported that they relied entirely on expert estimation, while seven of the managers reported that they used a combination (16%) of expert and model estimation. Out of these seven, six stated that they relied on a use case-based estimation method tailored to their company's types of projects and historical data. The last project used a combination of expert judgment and a company-defined estimation model.

5.5. Does choice of estimation method affect estimation accuracy or bias? (RQ5)

An Anderson-Darling test of the samples when divided by estimation method excludes normality. Since the sample sizes are unequal and the variances are unequal a non-parametric Kruskal-Wallis test on effort estimation medians is used [34]. The analysis on differences in median effort estimation accuracy (BRE) and bias (BREbias) grouped by choice of estimation method is displayed in Table 4.

	Expert	*Combination*	*p-value*
BRE	0.30	0.35	0.84
BREbias	0.22	0.17	0.45

Table 4: Effort estimation accuracy and bias by estimation method.

Based on our data, we find no indication that the use of estimation models in combination with experts leads to more accurate or unbiased estimates compared with expert estimates alone.

5.6. Are there differences in estimation accuracy or bias between companies that develop projects internally and those that develop for clients? (RQ6)

Out of the 42 projects that were completed, 11 were developed in-house (internal), while 31 were developed on contract for a client (external).

The samples do not follow a normal distribution, and there are differences in sample size and variance, as in the last subsection. The results of a Kruskal-Wallis test on differences in median accuracy values of internal and external development are displayed in Table 5.

	Internal	*External*	*p-value*
BRE	0.07	0.42	0.10
BREbias	0.05	0.35	0.07

Table 5: Effort estimation accuracy and bias by client.

This indicates that projects that are developed internally may be more accurate and less biased related to effort estimation than externally developed projects.

5.7. How do senior managers perceive the company's level of estimation skill? (RQ7)

In order to get an understanding of the managers' view of their companies' estimation accuracy, we asked them to provide us with their own assessment of the company's mean effort estimation accuracy. Responses ranged from 10% overestimation to 50% underestimation, with an average for all companies of 15.9% underestimation.

The managers were also asked about the outcome of their projects. They were asked to categorized projects conducted during the past year into three different categories:

(i) "success" – delivered on schedule and on effort, with the required functionality,

(ii) "challenged" – failure to meet either schedule, effort or functionality requirements

(iii) "aborted" – projects that were either stopped or underwent major revision.

The average response was 45% for "success", 49% for "challenged" and 6 % for "aborted". Note that these calculations are not adjusted for company size or number of projects.

There was consequently a difference between the estimation accuracy observed in our study and the beliefs of the managers. This difference is analyzed and discussed in Section 6.

5.8. On what basis is an estimation method selected? (RQ8)

In order to identify a possible rationale for the use of a particular estimation approach, the managers were asked to rate several possible reasons for choosing an estimation method. Each reason had to be given a rating from one to five, where five was the maximum. A summary of the average ratings is given in Table 6.

Reason for choosing method	*Rating (1-5)*
Estimator has had success with method	4.2
Consultant advice	2.7
Thorough testing	2.7
Structured analysis	2.1
Lectures at universities/colleges/courses	1.8
Review of other companies experiences	1.7
Market popularity	1.2

Table 6: Reasons for choice of estimation methods

Out of all possible reasons, only one scored above average (3) on the ratings. This was that the estimator had had previous success with the method. A total of thirteen companies gave this reason the highest possible rating (5). The managers were also given the opportunity to describe important, non-predefined reasons for selecting an estimation method. Among the reasons provided were "using function point methods has been shown

to be inefficient" and "the method used is good for persuading senior management to approve a project"!

5.9. How important does the organization perceive estimation as being, in comparison with other aspects of development? (RQ9)

When asked about how important effort estimation was viewed in comparison with other development aspects, the managers provided free text responses. Of the eighteen companies, fourteen answered that estimation was very important, extremely important or most important.

6. Discussion

The surveys available for comparison with our data are mainly between ten and twenty years old. Despite this, and other differences, our results are similar to those of the previous surveys. This suggests that the estimation methods and performance of software companies has not changed much since the 1980s. A summary of our results is displayed in Table 7.

Cost overrun	41% (mean), 21% (md)
Project used more than estimated effort	76%
Project used less than estimated effort	19%
Schedule overrun	25% (mean), 9% (md)
Project completed after schedule	62%
Project completed before schedule	2%

Table 7: Summary of Results

The frequency and magnitude of effort and schedule overruns in our survey (RQ1 and RQ2) were similar to those reported in previous surveys (see Section 2). We also observed that there may be a tendency that larger projects are more prone to underestimation than smaller projects (RQ3).

Our observations are similar to the results of the other surveys with respect to choice of estimation method (RQ4). In our 44 projects, 16% relied on an estimation model in combination with expert estimation, while 84% relied entirely on expert estimation. If we take into consideration that cost models are often complicated to use, and have not proved to be superior to expert-based methods [22], this lack of use of formal estimation models is not

surprising. Those of our companies who used models relied on use-case based models in combination with expert judgment, i.e., none of the projects relied on estimation models as the only estimation method. We were unable to provide conclusive results as to whether use of a combination of expert and model estimation was beneficial (RQ5), but this may have been due to the small sample of combination projects.

It seems as if there are differences between those who develop in-house, and those who develop for clients, regarding effort estimation accuracy and bias (RQ6). A possible reason for this observation is that in-house developers have closer proximity to the customer and more stable system properties, such as requirements, platform and implementation language.

The managers believed that there were almost as many "successful" (45%) as "challenged" (49%) projects, and only 6% aborted projects (RQ7). What we observed, on the other hand, was that only 11% were completed on schedule, on effort and with full functionality, while as much as 84% were "challenged". A potential reason for this is that a project that overruns its effort or schedule estimate by a few hours or days will not be interpreted as challenged by managers. If we define projects with a 10% overrun as "successful", the percentage of successful projects rises to 25%, while the percentage of challenged projects falls to 70%. If the success range is increased to include 25% overruns, the percentage of successful projects is 43%, while the percentage of challenged projects is 52%. Moores and Edwards [6] have previously described that most managers are comfortable with a level of accuracy around +/- 20%. The managers also believed that the magnitude of effort overruns was less (15.9%) than what we observed (mean 41% and median 21%).

Regarding choice of estimation method, the senior managers were quite aware of how the projects were estimated. However, it did not seem that they were concerned with analyzing the performance of their methods (RQ8). When asked about reasons for choosing a method, the only alternative with an above average rating was "the estimator has had success with the method". On the other hand, the managers were very clear when responding on the importance of effort estimation in comparison to other topics (RQ9), since 14 out of 18 companies rated the topic to be very important, extremely important or most important.

A possible connection between the three last observations may be that even if the managers perceive the topic of estimation to be important (RQ9), a lack of structured analysis of the situation (RQ8) may mislead them into believing that their estimates are more accurate than they actually are (RQ7).

6.1. Threats to validity

The most obvious threat to internal validity is the sample size, which is small when applying statistical inference-based analysis methods. The small sample size is a results of our labor-intensive data collection method based on interviews and the belief that it was more important to get high quality, in-depth information about a few projects instead of lower quality information about more projects, applying a questionnaire-based study method. Personal interviews helped to clarify numerous ambiguities, and to discard projects that did not retain accurate information on important aspects, such as estimated effort.

Most projects claimed to have met requirements related to functionality, but it is important to note that this was based on the managers own responses, since we did not have access to the end-users.

When evaluating the generalizability of the results one must consider that this is a survey of Norwegian companies. Norwegian companies are on average smaller (both in number of employees and revenue) and complete smaller projects than companies in countries such as the United States, the United Kingdom and Canada. There may also be business cultural issues that reduce the generalizability of the results.

Our results are, however, similar to those reported by the previous surveys presented in Section 2. This is an indication that the sample is probably not biased in any particular direction.

An important factor in general when measuring estimation skill in companies is that we can only analyze projects that were approved by managers or that won contracts. Such projects may be selected because they have optimistic estimates. At the same time, realistically estimated projects may be turned down because they seemed too expensive. This may be a “winners curse” that affects estimation accuracy in completed projects [38] which is reflected in surveys such as this.

7. Summary

The observations in our survey were similar to those reported by other researchers. It seems that choice of estimation method and level of estimation accuracy and bias are stable, being independent of year, technology and location. A possible reason for this observation is that alternative estimation approaches have failed to provide evidence that their use increase estimation accuracy.

It also appears that the focus on analysis of estimation accuracy is low in the Norwegian software industry. This may lead to an assessment of estimation skill that is misleading, which in turn may hamper improvement and education.

Further research will be to monitor the development in Norway over time, to see how different process improvement efforts or technological changes may affect estimation performance. We would also like to expand the survey, with replications in other countries.

Acknowledgements

This research was funded by the Research Council of Norway under the project INCO. Thanks to Chris Wright for valuable comments.

References

1. Bergeron, F. and J.-Y. St-Arnaud, *Estimation of Information Systems Development Efforts: A Pilot Study.* Information & Management, 1992. **22**: pp. 239-254.
2. Heemstra, F.J., *Software cost estimation.* Information and Software Technology, 1992. **34**(10): pp. 627-639.
3. Jenkins, A.M., J.D. Naumann, and J.C. Wetherbe, *Empirical Investigation of Systems Development Practices and Results.* Information & Management, 1984. **7**: pp. 73-82.
4. Lederer, A.L., et al., *Information System Cost Estimating: A Management Perspective.* MIS Quarterly, 1990. **14**(2): pp. 159-178.
5. McAulay, K. *Information Systems Development and the Changing Role of MIS in the Organisation*. *First New Zealand MIS Management Conference*. 1987. Wellington.
6. Moores, T.T. and J.S. Edwards, *Could Large UK Corporations and Computing Companies Use Software Cost Estimating Tools? - A Survey.* European Journal of Information Systems, 1992. **1**(5): pp. 311-319.
7. Paynter, J. *Project estimation using screenflow engineering*. *International Conference on Software Engineering: Education and Practice*. 1996. Dunedin, New Zealand. pp. 150-159.

8. Phan, D., *Information Systems Project Management: an Integrated Resource Planning Perspective Model*, *Department of Management and Information Systems*. 1990, Arizona: Tucson.
9. Purvis, M.K., S.G. MacDonell, and J. Westland, *Software Metrics in New Zealand: Recent Trends.* New Zealand Journal of Computing, 1994. **5**(2): pp. 13-21.
10. Sauer, C. and C. Cuthbertson, *The State of IT Project Management in the UK 2002-2003*. 2003, Templeton College, University of Oxford.
11. Moløkken-Østvold, K. and M. Jørgensen. *A Review of Surveys on Software Effort Estimation*. *2003 ACM-IEEE International Symposium on Empirical Software Engineering (ISESE 2003)*. 2003. Frascati, Monte Porzio Catone (RM), ITALY: IEEE. pp. 220-230.
12. Jørgensen, M. and K. Moløkken-Østvold, *How Large Are Software Cost Overruns? Critical Comments on the Standish Group's CHAOS Reports.* Submitted to Information and Software Technology (Can be dowloaded at: http://www.simula.no/publication_one.php?publication_id=711), 2004.
13. Heemstra, F.J. and R.J. Kusters. *Controlling Software Development Costs: A Field Study*. *International Conference on Organisation and Information Systems*. 1989. Bled, Yugoslavia. pp. 652-664.
14. Lederer, A.L. and J. Prasad, *Causes of Inaccurate Software Development Cost Estimates.* Journal of Systems and Software, 1995(31): pp. 125-134.
15. Lederer, A.L. and J. Prasad, *Information systems software cost estimating: a current assessment.* Journal of Information Technology, 1993(8): pp. 22-33.
16. Boehm, B., *Software Engineering Economics*. 1981, Englewood Cliffs, NJ: Prentice-Hall.
17. Kitchenham, B., et al., *An empirical study of maintenance and development estimation accuracy.* Journal of Systems and Software, 2002. **64**(1): pp. 57-77.
18. Lederer, A.L. and J. Prasad, *Nine management guidelines for better cost estimating.* Communications of the ACM, 1992. **35**(2): pp. 51-59.
19. Wydenbach, G. and J. Paynter, *Software Project Estimation: a Survey of Practices in New Zealand.* New Zealand Journal of Computing, 1995. **6**(1B): pp. 317-327.
20. Jørgensen, M. and D.I.K. Sjøberg, *The impact of customer expectation on software development effort estimates.* International Journal of Project Management, 2004. **22**(4): pp. 317-325.

21. Addison, T. and S. Vallabh. *Controlling Software Project Risks - an Empirical Study of Methods used by Experienced Project Managers. SAICSIT 2002.* 2002. Port Elizabeth, South Africa. pp. 128-140.
22. Jørgensen, M., *A Review of Studies on Expert Estimation of Software Development Effort.* Journal of Systems and Software, 2004. **70**(1-2): pp. 37-60.
23. Höst, M. and C. Wohlin. *An Experimental Study of Individual Subjective Effort Estimations and Combinations of the Estimates*. *Proceedings the 20th International Conference onSoftware Engineering*. 1998. Kyoto, Japan. pp. 332-339.
24. Cozby, P.C., *Methods in behavioral research*. 5th ed. 1993, Mountain View: Mayfield Publishing Company.
25. HegnarOnline, *Kapital DATAs 1500 største*. 2000.
26. Jørgensen, M. and D. Sjøberg, *An effort prediction interval approach based on the empirical distribution of previous estimation accuracy.* Journal of Information and Software Technology, 2003. **45**(3): pp. 123-136.
27. Miyazaki, Y., et al., *Method to estimate parameter values in software prediction models.* Information and Software Technology, 1991. **33**(3): pp. 239-243.
28. Conte, S.D., H.E. Dunsmore, and V.Y. Shen, *Software Engineering Metrics and Models*. 1986, Menlo Park: Benjamin-Cummings.
29. Briand, L.C. and I. Wieczorek, *Resource modeling in software engineering*, in *Encyclopedia of Software Engineering*, J. Marciniak, Editor. 2001, Wiley.
30. Stensrud, E., et al. *An empirical validation of the relationship between the magnitude of relative error and project size*. *Eighth IEEE Symposium on Software Metrics.* 2002: IEEE. pp. 3-12.
31. Phan, D., D. Vogel, and Nunamaker, *The Search for Perfect Project Management*, *Computerworld*. 1988. pp. 95-100.
32. Jørgensen, M., *How much does a vacation cost? or What is a software cost estimate?* ACM Software Engineering Notes, 2003. **28**(6): pp. 30.
33. Edwards, J.S. and T.T. Moores, *A Conflict Between the Use of Estimating and Planning Tools in the Management of Information Systems.* European Journal of Information Systems, 1994. **3**(2): pp. 139-147.
34. Cohen, J., *Statistical power analysis for the behavioral sciences*. 1969, New York: Academic Press, Inc.
35. Anda, B., E. Angelvik, and K. Ribu. *Improving Estimation Practices by Applying Use Case Models*. *4th International Conference on Product Focused Software*

Process Improvement, December 9 - 11. 2002. Rovaniemi, Finland.: Springer-Verlag,. pp. 383-397.

36. Anda, B., et al. *Estimating Software Development Effort Based on Use Cases - Experiences from Industry*. *4th International Conference on the Unified Modeling Language (UML2001), October 1-5, 2001*. 2001. Toronto, Canada.: Springer-Verlag. pp. pp. 487-502.
37. Anda, B. *Comparing Effort Estimates Based on Use Case Points with Expert Estimates*. *Empirical Assessment in Software Engineering (EASE 2002). April 8-10, 2002*. 2002. Keele, UK.
38. Gilley, O.W., G.V. Karels, and e. al., *Uncertainty, experience and the 'winners curse' in OCS lease bidding.* Management science, 1986. **32**(6): pp. 673-682.

Appendix I: Survey data

Nr	Client	Method	Estimate (hrs)	Actual	BREbias	Estimate (days)	Actual	BREbias	Funct.
1	External	Expert	330.0	319.0	-0.03	235	235	0.00	100%
2	Internal	Expert	560.0	1000.0	0.79	84	140	0.67	110%
3	Internal	Expert	300.0	600.0	1.00	156	156	0.00	100%
4	Internal	Expert	700.0	1400.0	1.00	182	224	0.23	100%
5	External	Expert	4227.0	5170.0	0.22	98	98	0.00	110%
6	External	Expert	1077.0	1150.0	0.07	42	49	0.17	110%
7	External	Expert	4000.0	9000.0	1.25	293	335	0.14	110%
8	Internal	Expert	1500.0	1512.0	0.01	70	70	0.00	100%
9	Internal	Expert	1125.0	1000.0	-0.13	106	106	0.00	100%
10	Internal	Expert	1249.0	1242.0	-0.01	153	214	0.40	98%
11	Internal	Expert	1410.0	955.0	-0.48	74	64	-0.16	98%
12	External	Comb.	12000.0	14000.0	0.17	619	640	0.03	95%
13	External	Expert	1249.0	1242.0	-0.01	92	106	0.15	100%
14	External	Expert	487.5	562.5	0.15	103	106	0.03	110%
15	External	Expert	640.0	1085.0	0.70	54	84	0.56	90%
16	Internal	Expert	3937.5	4012.5	0.02	138	152	0.10	95%
17	Internal	Expert	750.0	1200.0	0.60	117	457	2.91	100%
18	External	Expert	533.5	466.5	-0.14	97	104	0.07	100%
19	External	Expert	570.0	907.0	0.59	109	116	0.06	112%
20	External	Expert	292.0	342.0	0.17	113	113	0.00	105%
21	External	Expert	914.0	1903.0	1.08	196	217	0.11	120%
22	External	Expert	400.0	432.0	0.08	42	56	0.33	110%
23	External	Expert	705.0	1000.0	0.42	60	70	0.17	105%
24	External	Expert	2265.0	2732.0	0.21	210	245	0.17	120%
25	External	Expert	1932.0	5631.0	1.91	220	281	0.28	120%
26	External	Expert	2340.0	3454.0	0.48	140	245	0.75	150%
27	Internal	Expert	650.0	696.0	0.07	49	49	0.00	100%
28	Internal	Expert	27241.0	28645.0	0.05	296	336	0.14	90%
29	Internal	Expert	7520.0	8063.0	0.07	395	395	0.00	99%
30	Internal	Expert	6728.0	n/a	n/a	151	n/a	n/a	n/a
31	Internal	Expert	5450.0	8910.0	0.63	212	304	0.43	110%
32	Internal	Expert	90.0	180.0	1.00	21	56	1.67	100%
33	Internal	Expert	2720.0	n/a	n/a	152	n/a	n/a	n/a
34	External	Comb	145.0	195.5	0.35	79	84	0.06	100%.

35	External	Comb	190.0	101.0	-0.88	60	85	0.42	100%
36	External	Expert	593.5	593.5	0.00	152	152	0.00	100%
37	External	Comb	506.0	506.0	0.00	70	70	0.00	100%
38	External	Comb	3784.0	3746.0	-0.01	266	266	0.00	100%
39	External	Expert	1030.0	1335.0	0.30	122	122	0.00	115%
40	External	Expert	2170.0	3831.0	0.77	54	54	0.00	100%
41	External	Comb	3086.0	7844.0	1.54	183	183	0.00	100%
42	External	Comb	1982.0	3140.0	0.58	153	153	0.00	145%
43	External	Expert	133.5	261.0	0.96	273	334	0.22	100%
44	External	Expert	340.0	866.5	1.55	72	91	0.26	125%

Paper III:

Project Management of Public Software Projects: Avoiding Effort Overruns

Kjetil Moløkken-Østvold[1], Magne Jørgensen[1], Pål Sørgaard[2] and Stein Grimstad[1].
[1] Simula Research Laboratory, [2] Telenor R&D.

Submitted to Information and Software Technology.

---

1. Introduction

Effort overruns, abandonment, lawsuits, system breakdowns, and other "scandals" appear to be the rule, rather than the exception, where public software projects are concerned. This has been reported in the United States [1], The United Kingdom [2], Norway [3] and several other OECD countries [4]. It is important to note that this is not only the opinion of scandal-seeking tabloids, but also that of the more serious technical press. In addition, in the past five years, public officials in several countries have hosted conferences on the topic, and several reports addressing this problem have been written.

However, is it just the transparency of public projects that make them easily accessible by the media and the general public? Do public projects really face larger effort overruns than private projects? Or is this just a myth? In order to address this problem, we conducted a survey that compared effort overruns, and other factors relevant for software engineering project managers, of public and private software projects in Norway. We found that there are, indeed, causes for concern for those involved in public projects. These projects had effort overruns of a significantly greater magnitude than private projects.

Depending on the type of contract, effort overruns are either paid for by the client, written off as losses by the contractors, or there is a shared responsibility. In addition, project managers who face problems may be tempted to cut back on testing or functionality in order to reduce potential overruns, thus delivering lesser value to the clients. Therefore, we present an overview of the problem, and offer advice that is relevant for both software providers and public clients that seek to reduce effort overruns.

2. State of Practice

Effort overruns appear to be frequent in software development projects, whether public or private. In fact, a recent review of all surveys on software estimation found that 60-70% of all projects face effort overruns. The average magnitude of effort overruns is reported to be 30-40% [5]. Independent of when or where the survey was conducted, studies on software estimation found the frequency and magnitude of effort overruns to be the same.

Similar findings are reported from other research areas, such as projects for transport infrastructure. A recent comprehensive study on that topic, by Bent Flyvbjerg and his colleagues, found that 86% of the projects faced effort overruns, and that the average

magnitude of these overruns was 28% [6]. They also observed that the choice of estimation method did not affect estimation accuracy. In addition, the estimates were systematically biased towards underestimation, and the situation did not improve over time. Their data did not fit common explanations for effort overruns, such as lack of experience, poor estimation techniques and human optimism. Therefore, they renounced such explanations. Instead, they claim that stakeholders deliberately *lie* in order to get a project approved. Such lies are claimed to have economic and political motives. The projects they investigated in the transport sector were all public projects; it is therefore possible that their explanations of overruns apply to public software projects as well.

Terminology

Effort estimate: The most likely number of work-hours believed to be necessary to complete a project, as assessed by the managers and developers responsible for delivery.

Public software projects can be anything from large government defence initiatives to stand-alone web-portal projects for a small institution. Many studies have been conducted on why public software projects, independent of type, encounter problems so often. However, many of these studies have been initiated as a response to a particular high-profile failure. Often, they do not compare public "failures" systematically with projects in the private sector, or with successful public projects. In Texas, USA, the State Auditor's Office reported that "among the 21 largest IT projects it was monitoring as of October 2002, the average project was $388,000 over budget and 21 months behind schedule" [1]. However, these projects were not systematically compared to similar projects in the private sector in Texas, or public projects in other parts of the USA. The situation is similar in Norway. Only in particular cases does the Office of the Auditor General analyze projects, and then only particular problematic projects.

Research surveys over the past twenty years that have addressed software estimation [5] have paid little attention as to whether the projects were conducted in the public or private sector when analyzing patterns of overruns. Only one survey investigated possible differences between public and private projects. It found that private and public projects were equally likely to "fail" (experiencing effort or schedule overruns, or not delivering a

complete set of specified functionality) [7]. However, this survey does not report on any possible difference in the *magnitude* of effort overruns, i.e., a project with a 1% effort overrun is counted just as much a failure as is a project with a 150% effort overrun. In addition, there was a self-selecting, non-random, sample in that survey and this may have affected the results.

3. A Survey on Effort Overruns in Public and Private Software Projects

Between February and November 2003, we conducted a survey on estimation practices in the Norwegian software industry [8]. A total of 18 software companies participated. We used stratified random sampling, in order to ensure that small (less than 25 employees), medium (between 25 and 100 employees) and large companies (more than 100 employees) were represented. The companies submitted from one to four of their projects (based on available resources) for scrutiny. The criteria were that the projects should be over 100 hours (to exclude trivial tasks), be ended (either completed or abandoned), be the most recent cases (in order to ensure a non-biased sample), and that we had access to the managers of the projects. This resulted in a repository of 52 projects. We conducted in-depth interviews with senior managers of all companies, and the project managers. All interviews were performed semi-structured, face-to-face, and lasted between 30 and 70 minutes. Due to different interpretations of estimation-related concepts in the professional community, it was essential to have personal interviews in order to resolve ambiguities.

We excluded eight projects because they lacked information, or because most of the estimation and implementation work had been conducted by external sub-contractors. Two of the projects were abandoned before completion. This left 42 projects for the analysis.

In this analysis, we focused on the *most likely* effort estimate, as deemed required to complete a project based on the requirements of the client. This is the estimate used internally (i.e. not price-to-win) by those responsible for delivery at the stage when the decision to start the project was made. This estimate was then compared to the actual effort. This is in accordance with international practices on effort estimation studies [6]. Both estimates and actual efforts were measured in man-hours. There are several ways to analyze estimation performance in use in the software engineering research community. Our *accuracy* measure is calculated as:

$$accuracy = \frac{(x - y)}{\min(x, y)}, \quad \text{x = actual and y = estimated value.} \qquad (1)$$

This measure puts an even emphasis on over- and underestimation. An accuracy of 0% indicates that the project was on target. An accuracy of -100% indicates that the project used half the estimated effort; while an accuracy of 100% means that it used twice the planned effort. All positive values indicate overruns, and vice versa.

The main findings were that 76 % of the projects had effort overruns, and that the mean effort overrun was 41%. These findings are in line with those reported by other surveys and case studies on software estimation over the past twenty years [5], and studies on effort overruns in other areas [6]. This makes us confident that we did not have a particularly biased sample. Out of the 42 completed projects, 24 were undertaken for private clients, while 18 had a public client. Since most companies had both private and public clients, it is meaningful to compare the projects by analyzing estimation accuracy and related aspects based on the type of client. An overview of accuracy based on the different types of client is displayed in Figures 1 and 2.

Figure 1: Estimation Accuracy of Private Projects

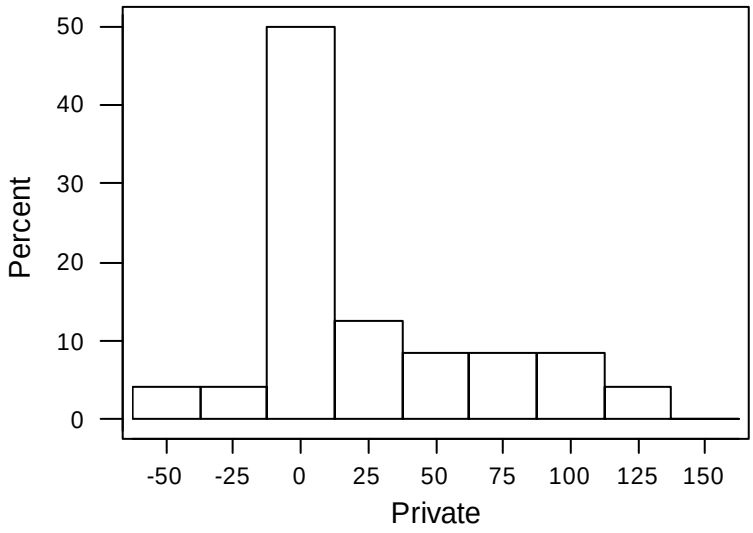
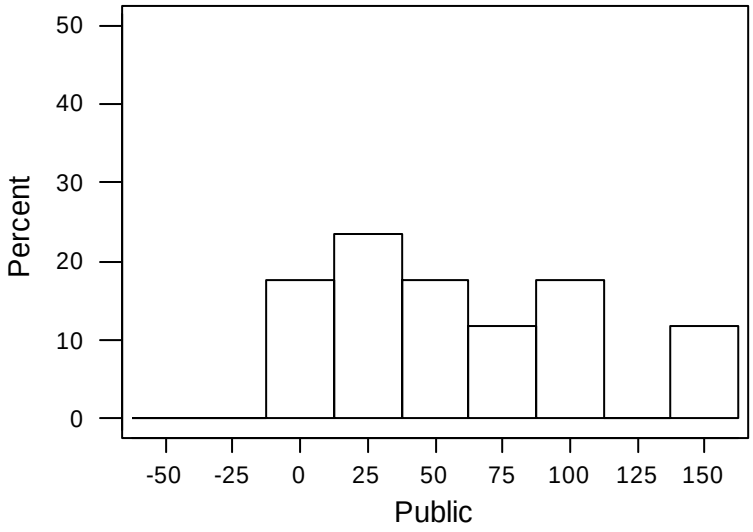


Figure 2: Estimation Accuracy of Public Projects



It seems as though projects undertaken for public clients encountered more problems than those performed for private clients with respect to being completed according to the estimated effort. Eighty-three percent of the public projects encountered effort overruns, compared to 71% of the private projects. More interesting is that the average effort overrun was 67% for projects that had a public client, while it was 21% for projects with a private client. When performing a statistical analysis (one sided t-test) on the magnitude of effort overruns, the resulting p-value is 0.005, so it is a very unlikely that the observation is coincidental. The corresponding median effort overrun is 53% for public projects, and only 7% for private projects.

We were not able to identify any differences between the samples related to important project properties that could explain the observed difference. Public projects had an average workload of 3068 man-hours, while private projects had an average workload of 3167 man-hours. Nor was there any difference in factors related to estimation approach, such as choice of estimation method, use of Work Breakdown Structures, checklists, combination of estimates or experience databases. There was a slight difference in delivered functionality. The mean amount of delivered functionality, as stated by the project managers, was 110% for public projects, and 103% for private projects (median values were 100% for both).

However, this small difference does probably not account for the large observed differences in effort overruns. Also important is the fact that the delivered functionality is a subjective measure, based on the managers' responses. Hence, reports of delivered functionality could be influenced by a desire to rationalize the effort overruns, e.g., managers who had encountered large effort overruns might try to justify this by claiming to have delivered more functionality than that which was originally specified.

In all interviews, the project managers were also asked to provide free-text responses to explain the project outcomes, related to estimates, actual effort and delivered functionality. We found that the descriptions that were most often associated with public projects, but not with private projects, were the following:

1. Lack of, or poor, requirement specification.
2. Complicated procurement procedures.
3. The client had allocated inadequate resources to decision making.
4. Immature clients.

Again, it is essential to notice that these responses are from the project managers of the contractors. The opinions of the internal project managers and other parties at the client may be the opposite.

4. Reasons for Effort Overruns in Public Software Projects

We found that not only were public software projects prone to effort overruns, but also that they had significantly larger effort overruns than private software projects. There was no evidence of what Flyvbjerg [6] characterizes as technical differences, i.e. differences in estimation approach or choice of development model, that could explain our findings. We therefore suggest there are other systematic differences between the types of client that may explain our results. In order to explore such differences, and offer advice to software engineering project managers and clients, we use information from research reports on public projects, feedback from professionals, and our previous experiences, in addition to results from our survey.

Most of the relevant experiences exist in the form of reports commissioned by government bodies. They often focus on public software failures in a wide sense, and effort overrun is only one, albeit important, measure of failure. One of the most thorough

investigations of public software projects was conducted by the British Parliamentary Office of Science and Technology [2]. They compared the investigations of several groups, including one from the industrial perspective [9] and one from the public client perspective [10]. On an international level, a report was compiled by OECD [4] after an expert meeting on public projects. Among the eleven contributions to that report, there is a section from Norway prepared by the Directorate of Public Management [3]. It is also interesting to compare different countries directly, such as has been done with Norway and Finland [11].

We also include our experiences from industry and from work at the Directorate of Public Management. Another important source for explaining the observed differences is the feedback we have received over the past six months while presenting our work in different contexts. This includes meetings with the Office of the Auditor General in Norway, the EFFIN–seminar (The EFFIN project goal is to develop user-centred methods and tools for use in the context of the public sector when introducing new technology), a presentation for the international management of one of Europe's premier consultancy companies, and senior managers and project managers participating in a seminar on software project management hosted by Simula Research Laboratory.

When we compare the information from all our sources, the main differences between public and private projects appear to be on three levels: political, organizational and individual. For some of the reasons provided, we only have access to opinions or reports that perceive this to be a problem in public projects. It is possible that such factors may be a problem in private companies as well, but this has not, to our knowledge, been systematically investigated.

4.1. Political level

The political nature of the decision processes in public administration may prove dysfunctional in a software engineering context.

- *Political deadlines of projects and efforts.* Politicians and their loyal government officials have a clear focus on measures that can be presented to the electorate as achievements of the current government. This focus may lead to unhealthy time estimates and to reduced emphasis on long-term measures that cannot be attributed to a specific government. It makes it less likely that endeavours such as process improvement programmes will be initiated. This political agenda may explain, at

least partially, the lack of effort and interest in learning from previous failures and previous advice as to what should be done to improve the situation [11].

- *Idiosyncrasies of the budget process.* In the Norwegian government, the annual budget process generally involves a flat reduction in running funds and a so-called "profile allocation" used to strengthen purposes of high political priority. As a result, it may be difficult to obtain funding for a modest software project, while there may be possibilities within the profile allocation provided that the goals of the project are sufficiently politically attractive. Such mechanisms are also found in other countries, and may act as an incentive to increase the size and level of ambition of the projects, which is counterproductive, since large projects are more likely to encounter problems [2, 4]. Moreover, the high focus on the budget process leads to a comparatively weaker emphasis on implementation and follow-up activities. As a result, there is less focus on the projects once started, and the incentives to learn from their achievements are weak. Therefore, issues regarding implementation and realism of expectations and estimates may receive little attention when projects are approved [12].
- *Political games.* We have experienced that the willingness to see and admit that there are major problems with public software projects is limited. To admit the existence of a problem may amount to an acknowledgement of responsibility. W learned that several middle-level officials that attended our briefings had been encouraged *not* to even meet with us, as senior officials believed that to meet would be tantamount to an acknowledgment of blame. Several officials were also surprised by our findings, and some even met them with denial.

4.2. Organizational level

The public sector has a set of structural and organizational characteristics that differ from private business that may have negative effects on the probability of success of software projects.

- *Regulations on procurement.* In countries subject to European Union law, public projects over a certain value are subject to extensive procurement procedures. The goals of these regulations are to stimulate competition and avoid "under the table" deals. However, the result is that many public clients feel pressured to choose the lowest offer whether constrained by legislation or not [2]. When overruns appear, or

other goals are not met, they feel they can justify their choice by stating that they "did everything by the book" and based their choice on price. Choosing the cheapest solution is especially dangerous when one does not have the competence to assess the capabilities of the competing contractors. In a recent study on software project bidding, we observed that the contractors with the least experience delivered the lowest bids [13]. Probably, this is caused by a combination of naïve optimism, and quoting a "price to win" in order to gain entry to the market. The United Kingdom is increasingly trying to use value for money as a criterion for evaluation, instead of the lowest price. The Office of Government Commerce points out that "value for money" is *rarely* synonymous with the "lowest price" [2]. This shows that the use of common sense, within the possibilities of current legislation, is not discouraged. Quite commonly, there will be opportunities for skilled clients to choose the best *overall* solution for their purpose, and not blindly choose a supplier based on one or two criteria alone.

- *Regulations on development processes.* Until recently, many government bodies in the United States were required to follow sequential (waterfall) based development models, even though flexible (e.g. incremental) development models were recommended by many independent sources [14]. A somewhat similar situation has been the case in Norway, were most standard IT contracts provided by the Directorate of Public Management have been based on the waterfall model [3]. Many public clients have believed that it was too difficult to use incremental development and to split a project into several smaller deliveries, which is highly recommended by independent reports [2, 4]. In our survey however, there was no difference in the use of development models with respect to the type of client, i.e., the type of development model cannot explain the observed difference in estimation accuracy by appeal to the type of client. However, further analysis, as presented in Table 1, suggests that public projects do not receive full benefits from incremental development. Reasons for this lack of benefit, as reported by the project managers in our survey, may be that although flexible models may be used in public projects, there are problems related to lack of user feedback and problems with delayed decision-making that reduce the benefits.

	Flexible	*Sequential*	*Total*
Private	-2% (n=11)	40% (n=13)	21% (n=24)
Public	58% (n=8)	74% (n=10)	67% (n=18)
Total	24% (n=19)	55% (n=23)	*41% (n=42))*

Table 1: Mean effort overruns based on client and development model type

- *Difference in business culture of the private contractor and the public client.* Most development work in public projects is done by outside contractors. Often, there will be a clash of culture between a private contractor and a public client [3, 9, 15]. In extreme cases, a public client will view the private contractor as an adversary [2]. On the other hand, the private contractors may also have little knowledge about the political factors involved in public projects, and are often too optimistic in their efforts to win contracts. Such optimism may materialize in project bids that are based on a best-case scenario, even though problems are likely to arise. It is the responsibility of the contractors to provide *realistic* estimates.
- *One-of-a-kind systems and preference for new technology.* We have observed a tendency to opt for new and advanced, instead of old and proven, technologies in public projects. This may be stimulated by political conditions, which favour high levels of ambition. A way to project resolve and competence is often to choose the newest and often experimental technologies, instead of selecting more conservative and well-tested options [3, 9, 15]. It may also be a result of the trivial fact that government, almost by definition, has a series of one-of-a-kind systems, and therefore may tend to use less off-the-shelf software than private companies. This opens up the potential for more (too much?) creativity with respect to functionality and technical solutions, which leads to projects with higher risk.

4.3. Individual level

The public sector has a reputation for not being able to attract professionals who are skilled in technology and project management. We do not claim that professionals in the public sector are incompetent or negligent; it is, rather, a fact that there seems to be a poor mix of competences that is not so apparent in private enterprises. There are simply more

computing specialists, engineers and IS-professionals in a private company than in comparable public agencies.

- *Lack of project managers.* A good project outcome also depends on a good internal project manager. A common limitation in the public sector is that there is a lack of good, business-oriented, project managers [2, 9] since there is no career path for internal project managers in public institutions. In addition, the lack of a management model could be responsible for the lack of requisite skills that is most prominent at the management level [3]. Such shortcomings may result in overly ambitious projects, where the underlying risks are not properly identified and managed [3, 15].
- *Lack of IT skills in organizations.* The OECD report stated that a recurrent problem is the lack of IT skills in the public sector [15]. Using government statistics, Guri Verne from the Norwegian Directorate of Public Management found that only 0.07% of over 100,000 government employees hold a Masters degree in computing[1]. We do not have similar figures from other countries, but reports point to this factor being a common problem in the public sector. In addition, in Norway, the Directorate of Public Management has documented that top-level management does not posses a adequate level of IT competence [3]. In Britain, the number of IT practitioners in government fell from 12,000 to below 3,000 in just five years [2]. The lack of IT skills contributes to the *problem of poor requirement specifications* in public projects [2, 3], which makes estimation work difficult.

5. Conclusions

Our empirical findings indicate that the average effort overrun in public sector software projects is significantly higher than in similar projects in the private sector. We recognize that some claim that this is due to underreporting of failures in private sector, but we have no indications that this is the case in our material, which was randomly provided by the contractors. Under any circumstances, the rate and size of the overruns is an indicator of severe problems in public sector software projects.

Based on our material, previous experience and on the feedback we have received on our quantitative material, we offer a set of possible explanations for our findings. We believe

[1] There are probably quite a few computing specialists that are registered as unspecified MSc. Real figures are therefore not that dramatic, but failure to recognize computing as an important specialization is also an important part of the problem.

these explanations provide a convincing interpretation of the nature of the project estimation (and project management) problems we have documented.

The challenge for public sector, and their providers, is to get to the grips with these problems, and to do so in a convincing way. To do so, we would propose the following series of actions:

- A change of attitude, involving a realization that the public sector is, increasingly, a software-based sector, and hence needs to address software issues more seriously.
- Increased awareness of the need for software competence when specifying, choosing and developing solutions. If such competence is not available, outside help should be hired.
- Implementation of mechanisms to learn from experiences from previous projects (post-mortem evaluations).
- Reduction of project size and risk, e.g. through use of incremental development models.
- Increased use of evaluation of bids according to value-for-money, as opposed to price being the only criterion.
- Initiation of software process improvement activities.

Since we have observed that similar advice has been previously ignored, and that the willingness to realize the seriousness of these problems is limited, we believe that it will be very demanding to take these actions. We believe, however, that this is needed in order to meet the ever-increasing ambitions of electronic government.

References

1. Strayhorn, C.K., *Limited Government, Unlimited Opportunity - Recommendations of the Texas Comptroller*. 2003, Texas Comptroller of Public Accounts.
2. Pearce, S., *Government IT Projects*. 2003, The Parliamentary Office of Science and Technology.
3. Sørgaard, P. and M. Vestre, *Country Report from Norway*, *OECD-PUMA expert meeting on management of large IT projects.* 2003, Statskonsult: Oslo.
4. Kristensen, J., *Management of Large Public IT Projects: Case Studies*. 2001, OECD.
5. Moløkken-Østvold, K. and M. Jørgensen. *A Review of Surveys on Software Effort Estimation. 2003 ACM-IEEE International Symposium on Empirical Software*

Engineering (ISESE 2003). 2003. Frascati, Monte Porzio Catone (RM), ITALY: IEEE. pp. 220-230.

6. Flyvbjerg, B., M.S. Holm, and S. Buhl, *Underestimating Costs in Public Works Projects - Error or Lie?* Journal of the American Planning Association, 2002. **68**(3): pp. 279-295.
7. Sauer, C. and C. Cuthbertson, *The State of IT Project Management in the UK 2002-2003*. 2003, Templeton College, University of Oxford.
8. Moløkken-Østvold, K., et al. *A Survey on Effort Estimation in Norwegian Software Industry*. *10th International Symposium on Software Metrics*. 2004. Chicago, Illinois, USA: IEEE Computer Society. pp. 208-219.
9. West-Knights, L., *Getting IT Right for Government - A Review of Public Sector IT Projects*. 2000, Intellect.
10. Steward, A. and I. McCartney, *Successful IT - Modernising Government in Action*. 2000, Central IT Unit: London.
11. Sørgaard, P., *IT Co-Ordination and Public Management Reform - A Comparison Between Finland and Norway*. 2000, Ministry of Finance, Public Management Department, Helsinki, Finland.
12. Sørgaard, P. *Co-ordination of E-government. IFIP WG 8.6*. 2003. Boston, USA: Kluwer. pp. 53-77.
13. Jørgensen, M. and G. Carelius, *An Empirical Study of Software Project Bidding.* Accepted for IEEE Transactions on Software Engineering (preliminary version can be downloaded from: http://www.simula.no/publication_one.php?publication_id=709). 2004.
14. Larman, C. and V.R. Basili, *Iterative and Incremental Development: A Brief History.* IEEE Computer, 2003(June): pp. 2-11.
15. Kristensen, J. and B. Bühler, *The Hidden Threat to E-Government - Avoiding Large Government IT Failures*. 2001, OECD.

Kjetil Moløkken-Østvold is a PhD student at the University of Oslo and the software engineering research group at Simula Research Laboratory in Norway. His main research interests are software effort estimation, group processes and software process improvement. He received his MSc degree in computer science from the University of Oslo, Norway, in 2002. He joined Simula Research Laboratory in October 2002. Contact him at kjetilmo@simula.no.

Magne Jørgensen is a professor in software engineering at University of Oslo and member of the software engineering research group of Simula Research Laboratory in Oslo, Norway. His research interests include software effort estimation, uncertainty assessments in software projects, expert judgment processes, and, learning from experience. He received the Diplom Ingeneur degree in Wirtschaftswissenschaften from the University of Karlsruhe, Germany, in 1988 and the Dr. Scient degree in informatics from the University of Oslo, Norway in 1994. He has 10 years industry experience as consultant and manager. Contact him at magnej@simula.no.

Pål Sørgaard is a research manager within Telenor R&D in Norway. His research interests include telecommunications policy, public sector IT and systems development. He received his MSc in computing science in 1985 and his PhD in 1989, both from Aarhus University in Denmark. He has previously worked as a researcher at the Norwegian Computing Center and as an associate professor at the University of Oslo. From 1997 to 2001 he served as Assistant Director General in the Directorate of Public Management, where he was responsible for the IT department. He can be contacted at pal.sorgaard@telenor.com.

Stein Grimstad is a PhD student at the University of Oslo and the software engineering research group at Simula Research Laboratory in Norway.

Paper IV:

Expert Estimation of Web-Development Projects: Are Software Professionals in Technical Roles More Optimistic Than Those in Non-Technical Roles?

Kjetil Moløkken-Østvold and Magne Jørgensen.
Simula Research Laboratory.

Empirical Software Engineering, 2005, Volume 10, Issue 1, pp 7-29.
Editor: Michael S. Deutsch

Abstract: *Estimating the effort required to complete web-development projects involves input from people in both technical (e.g., programming) and non-technical (e.g., user interaction design) roles. This paper examines how the employees' role and type of competence may affect their estimation strategy and performance. An analysis of actual web-development project data and results from an experiment suggest that people with technical competence provided less realistic project effort estimates than those with less technical competence. This means that more knowledge about how to implement a requirement specification does not always lead to better estimation performance. We discuss, amongst others, two possible reasons for this observation: (1) Technical competence induces a bottom-up, construction-based estimation strategy, while lack of this competence induces a more "outside" view of the project, using a top-down estimation strategy. An "outside" view may encourage greater use of the history of previous projects and reduce the bias towards over-optimism. (2) Software professionals in technical roles perceive that they are evaluated as more skilled when providing low effort estimates. A consequence of our findings is that the choice of estimation strategy, estimation evaluation criteria and feedback are important aspects to consider when seeking to improve estimation accuracy.*

Keywords: Effort estimation, web development, bidding process, individual differences.

1. Introduction

This paper examines the relationship between the accuracy of expert effort estimates of web-development projects and the estimators' type of competence. In spite of a high number of published estimation models, see [1] for an overview, software development effort estimates frequently rely on expert judgement [2-4], i.e., an estimation process where one or more competent people estimate project effort with little or no help from formal estimation models. One reason for the frequent use of expert estimation is that we lack substantial empirical evidence in favour of formal estimation models. We found [3] fourteen studies comparing the estimation accuracy of expert estimates with that of model estimates for software development and maintenance work. Of those studies, five were in favour of expert estimation [5-9], five found no difference [10-14], and four were in favour of model-based estimation [15-18]. Another argument in favour of expert estimates is that experts are more flexible regarding the type and format of the estimation information than formal estimation models.

Expert estimates may be based on a variety of estimation strategies, and conducted by people with different types of competence. The present knowledge on how an expert's strategy and competence affect estimation accuracy is, however, limited. The only software study related to this topic, as far as we know, suggests that neither maintenance skills, measured as frequency of unexpected problems, nor length of experience are good indicators of accurate estimates of one's own maintenance work [18]. Similar lack of correspondence between expertise in completing the task and estimation accuracy is reported in other studies on human judgment [19].

This paper focuses on effort estimates used as input to a bidding process. Such estimates may easily be affected by a price-to-win strategy [20] and experts may, therefore, provide effort estimates that are much too low, due to the so-called "anchoring-effect" [21, 22].

If a company is selected as a contractor based on a bid that is too low, they may later try to skip functionality or expand the project by means of change requests to make the project profitable. This is, however, a dangerous strategy. The customer expects the specified functional solution for a price at least in the neighbourhood of what was initially agreed upon, and may select another company for the next project. Clearly, realism in the estimates used as a basis for bidding is important for a company's long-term financial success.

Most of the previous research on software effort estimation has focused on traditional software projects, and there has not been much focus on web-development projects [23]. Although the challenges in the web-development industry are, to a large extent, similar to those in the traditional software development industry, there may be additional important estimation issues related to the increased focus on graphic and user interaction design in web-development projects. Another important difference from traditional software development companies, which often have few, but large projects, is that many web-development companies provide a very high number of project bids on relatively small projects [24]. For example, the web-development company described in Section 2 of this paper had less than 100 employees, but nevertheless provided about 500 project bids per year. A high amount of project bids on small projects means that it may be difficult to conduct high quality estimation work on each bid. There may, for example, be insufficient estimation resources available to use many employees for several days on each estimate. The selection of competent personnel to conduct fast and sufficiently accurate expert estimates is, therefore, essential.

Web-development company employees have varied backgrounds and roles. There is, however, little literature describing the different roles in web development [23], and we have been unsuccessful in discovering any earlier research about how employees in roles typical for web-development projects differ in their estimation process and performance, which is the topic of this paper.

The remainder of this paper contains a description of the web-development company and software professionals we used as our case organization and experiment participants (Section 2), the hypothesis and motivation (Section 3), the design of the experiment (Section 4), the results from the experiment (Section 5), a discussion of the results (Section 6), and a conclusion and description of further work (Section 7).

2. The Web-development Company and its Employees

The company that participated in our study is the Norwegian branch of a large international web-development company. At the time of the study, the branch had about 70 employees. For their last completed budget year (2000), the branch studied had a turnover of 119 million NOK (16.5 million USD). The role of the company is that of a contractor [23], producing web-solutions for its customers. The projects were mainly web portals, e-commerce solutions or content management systems.

The organization can be viewed as quite immature, since it only had existed in its current form for about two years at the time of the study. It had been incorporated in an international organization after a gradual merging of five different national companies. They were not concerned with either CMM or ISO certification, and were therefore not rated according to these standards.

They had switched between several practices for web engineering during the past few years. The one in use at the time of the study was a company defined work process. It was based on waterfall development, and contained six phases: strategy and concept, specification, development, test, implementation and evaluation. Through a corporate initiative, they were beginning to implement a tailored version of Rational Unified Process (RUP) internationally and at the local branch. This was only in the initial phase at the time of the study.

The employee responsible for a project was usually also responsible for the estimation of that particular project. All estimation was expert based, and no algorithmic models, databases or checklists were in use. However, some of these experts used a predefined estimation process with a project work breakdown structure [25]. Depending on the type and size of the project, the expert responsible could ask other experts (mainly technicians or designers) for input about their areas of expertise. Table shows estimation accuracy information based on a survey of all projects (n=275) conducted by the company in the period January to October 2001.

Overestimated Projects	*Projects on Target*	*Underestimated Projects*
7 %	36 %	57%

Table 1: Company Project Estimation Performance

Most projects (57%) were underestimated, and there was a total of 15% write-off (hours underestimated that were non-billable) for all projects. These 275 projects had a total of 94 748 billable and 111 430 actual work hours. This is not uncommon for software development, as observed in several other surveys [4].

From the interviews conducted at the company, we found that a typical reason for a project being "on target" was that it was paid per work-hour, overestimated, and the "remaining effort" was used to improve the solution. For small projects, with a customer-focus on quality of delivery, this may be a reasonable project approach. Other projects that

were originally underestimated may also have been completed "on target" due to simplifications of the original estimated delivery. For this reason, one should not interpret the project review presented here purely as a measure of estimation skill. A previous paper [22] contains a more comprehensive discussion on this topic.

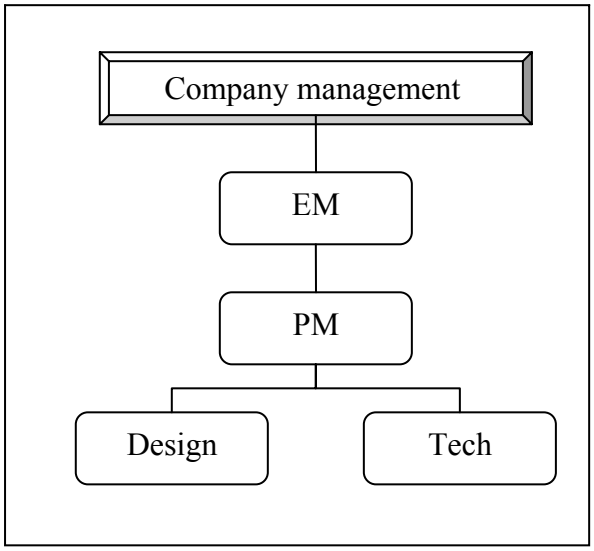
Most members of the company are assigned to one of the following four company roles:

- Engagement Manager (EM) – A person in this role is responsible for the customer contact, and usually handles contract negotiations. Typically, EMs have a business, or other non-technical, background and education.
- Project Manager (PM) – The leader of the project. Handles planning, resource allocation and day-to-day supervision of the project. A PM typically has a technical background.
- User Interaction Designer (Design) – A person in this role is responsible for Human Computer Interaction (HCI), graphical design, etc. User Interaction Designers have a large variety of backgrounds, most of them non-technical.
- Technology Developer (Tech) – This role is similar to that of the traditional software developer. Typically, Technology Developers have a technical education and background.

These roles are similar to those described by McDonald and Welland [23]. The only exception is that the company we describe does not have designated domain experts. From what we observed, that role in each project was typically covered by the customers.

The organization divides the project work into four tracks, depending on role: EM-track, PM-track, Design-track and Tech-track. The distribution of workload on the different tracks is similar to that described by McDonald and Welland [23], with an average proportion of the total project effort of 10% on the EM-track, 20% on the PM-track, 35% on the Design-track, and 35% on the Tech-track. When analyzing the company, we found a clear control hierarchy based on the different roles (see Figure 1).

Figure 1: Role Hierarchy



EMs and PMs have the most contact with customers.

3. Hypothesis

When reviewing data from completed projects more thoroughly, we observed a tendency for projects with heavy Tech-track involvement to be more often underestimated than those with medium or low Tech-track involvement.

A common scenario, when the estimator responsible for a project required assistance, was that people with a technical background estimated the Tech-track work and people with a non-technical background estimated the Design-track work. Consequently, if people with a non-technical background were less prone to underestimate their work, we should find a lower amount of non-billable work (write-off) on projects with a high proportion of non-technical work. To test this relationship, we randomly selected seven projects with a high degree (>40%) of Design-track work. These projects included much less Tech-track work than most of the other web-development projects in the company, with an average distribution of 21%, 54%, 23% and 2% for the different tracks (Tech, Design, PM and EM). The average track distributions of all projects were, as described earlier, 35%, 35%, 20% and 10%. The average write-off of the seven 'high Design-track involvement' projects was only 6%, i.e., a write-off much lower than the average 15%. This finding motivates the hypothesis that software professionals with a technical background are more optimistic than

those in non-technical roles. Observational studies of this type, however, can only indicate a relation, since there are other possible explanations for the finding, e.g.:

- There is a greater probability that unforeseen problems will arise in Tech-track work, and hence it is more difficult to estimate. Such problems can be difficult to anticipate in an early stage of the project [20], i.e., the stage when the bidding estimates are completed.
- When a project is behind schedule, it is easier to cut down on non-technical than technical activities, i.e., the actual use of effort for Tech-track work is less controllable by the developers. Earlier interviews with the employees of the company [22] indicated a very high "flexibility" in user interaction and graphic design activities.

There is, for these reasons, a need for more controlled studies to supplement the observational findings. In particular, there is a need for controlled studies on the level of optimism where comparable tasks are estimated. Motivated by the observations described above, we hypothesized the following relationship between role and estimation performance:

H1: Experts in technical roles provide more optimistic effort estimates than experts in non-technical roles.

An experiment that was designed to test this hypothesis is reported in the following sections.

4. Experiment Design

To test our hypothesis, we randomly selected 20 (five from each company role) software professionals employed by the company described in Section 2. Through payment we were able to select experienced software professionals in all roles. Based on their background and current roles, we categorized the participants in the EM and Design roles as "Non-technical" (NT) and the participants in the PM and Tech roles as "Technical" (T). All T-group members had a technical background, while the NT-group members had various other non-technical backgrounds (design, economics etc.). All personnel in the technical roles had education as engineers, master of engineering or equivalent. Their previous and current roles, both in the company studied and in earlier places of employment, had either been as

technicians, software developers and/or project manager. Personnel in non-technical roles were educated as graphic designers, or held a Master of Arts, an MBA or another, equivalent, qualification. All participants had at least three years of college or university education, and an average of over six years' work experience in the IT-industry.

All participants were instructed to estimate the effort required to complete a web-development project based on the same requirement specification, employing their preferred estimation strategy. The project to be estimated was a project conducted by the company, selected by their Chief Project Manager. It was described as representative, and meaningful to estimate in about one hour. The actual project had received little publicity, and none of the participants in the study knew about it. The requirement specification given to the participants did not include the actual customer or project name. The complete requirement specification, as presented to the participants, is enclosed as Appendix I.

The project was in its start-up phase when it was selected. The actual effort of the project later turned out to be 2 365 work-hours. The project lasted approximately six months, and was completed several months after the study was conducted. It did not include development of unusually complex software, although it had a rather high proportion of technical activities (56% on Tech-track work vs. an average of 35%), see Table 2.

Track	*Proportion of Effort*
EM	4 %
PM	17 %
Design	23 %
Tech	56 %

Table 2: Actual Project Distribution of Effort per Track

As estimation input, each participant was given the requirement specification as delivered by the actual customer. The requirement specification was two pages long, which was not atypical for specifications received by the company as a basis for project bids. The participants estimated the most likely effort (ML) measured in work-hours. The participants used 50-60 minutes on the estimation task.

After this individual estimation, there was a group discussion, the purpose of which was to come to an agreement on a common estimate for the project effort. These discussions were videotaped. In addition to the testing of the hypothesis H1, we wanted to investigate whether discussion in unstructured groups could be a reasonable method for combining the

judgment of several experts. The results indicate that it may be beneficial to combine estimates from experts with different backgrounds in order to reduce individual biases. This part is described in a separate paper [26].

Two questionnaires were completed by each participant, including information about the individuals' backgrounds and their estimation processes. We apply the information from the questionnaires and videotape in our discussion of the experiment results in Section 6. An overview of the steps in the experiment is presented in Table 3

Step	*Activity*	*Description*
1	Selection of company	The described company was selected on the basis that it was evaluated as being representative of the web-development community.
2	Selection of task	The task was selected in cooperation with the chief project manager to ensure that the project was representative and meaningful to estimate in about one hour.
3	Selection of subjects	In cooperation with the chief project manager, subjects that had no knowledge of the project, and that could be clearly defined as being either technical or non-technical employees were selected.
4	Individual estimation	The subjects estimated the assignment individually
5	Questionnaire #1	The subjects answered a background questionnaire
6	Group estimation	The subjects formed groups and re-estimated the assignment.
7	Questionnaire #2	The subjects answered a questionnaire related to the assignment and their personal view on the effort required.
8	Debriefing	The subjects were debriefed on the background and purpose of the experiment.

Table 3: Experiment Steps

Although we did our best to achieve a high level of realism in the experiment, the study had limitations that should be considered when interpreting the results:

- The experimental setting did not enable the participants to contact the customer. However, this situation is similar to the first bidding round of a normal estimation process. If the customer is satisfied with the first bid, further contact is initiated, and more thorough estimation work is conducted.
- The estimators could not discuss estimation issues with colleagues in the study. This restriction on realism was a part of the design of our study on individual estimates, but means that the participants may have made better individual estimations in a more realistic setting.

- The experiment did not allow the participants to use as much time to complete the estimate as they might have wanted. On the other hand, as described earlier, the general situation with many bids and small projects implied that participants in all roles were used to provide effort estimates within a short time in real estimation tasks. None of the participants complained about problems with delivering their estimates within the time available.
- The participants knew that this was an experiment and may, therefore, have had a lower motivation for providing accurate estimates. The experiment addressed this threat to validity in three ways. (1) Participants were informed that only serious participation would enable participation in further studies. (2) As a way of motivating serious participation, participants were paid according to the time spent, which was reported as billable hours. (3) Following their individual estimates, the participants met in groups to discuss their estimates. This means that the participants knew that their estimates would be evaluated by other members of their organization.
- It was not common for people in the Design and Tech roles to estimate the total project effort. However, the participants in these roles had participated in several projects and knew approximately how large a part of the total effort their work on different types of projects would be, and could use this knowledge to derive the total effort estimate.

However, since we were studying an eventual difference between technical and non-technical groups, it was most essential to keep the experimental conditions equal for both groups. This was done in the experiment, as all participants received the same treatment. We must, however, be aware of the possibility that the groups might have been affected differently by the experimental conditions. It could affect the results if, e.g., the limited time available were to cause more stress in, and hence lower the performance of, one of the groups. However, we do not believe that this was the case, because participants from both groups regularly worked together on the same assignments, with the same clients and deadlines.

The main argument supporting the internal validity of the experiment is the similarity between the participants' estimates of most likely effort and the actual project's estimate. The project plan describes a planned effort of 1 240 work-hours. The mean estimate of the participants in the experiment was 1 087 work-hours, i.e., only a little less than the

originally planned effort. The experimental situation does not seem to bias the results in any significant way.

The external validity of the experiment is more difficult to assess. Only one company and only one project were used. However, our experiences with other companies lead us to believe that the company we studied is similar to many other web-development companies, both in size and (lack of formal) estimation process. Our observations of the company also indicate that they are similar to other small companies who employ an ad-hoc estimation approach, as described by Moses and Clifford [27].

The time available limited us to use one project in the experiment, but the chief project manager assured us prior to the experiment that it was representative, and we conducted interviews with the employees who participated in the actual project after the project was completed. These interviews revealed no extreme properties of the project, only that their initial estimates had been far too optimistic.

Even if the limitations of the experiment were to have had an unwanted impact on the estimates, this would only be problematic if the impact was much larger on some groups of the participants, e.g., the people from Tech-track. We have no reason to believe that this was the case.

5. Experiment Results

The individual estimates, divided into the NT and T groups are shown in Table 4. All estimates are in work-hours.

The actual effort of the project was 2 365 work-hours, and most participants gave an estimate that was much lower than the actual effort. However, the actual effort of a new project based on the same specification may have required less (or more!) effort than the completed project. The actual effort of the completed project cannot, for this reason, be taken as more than an indication of estimation accuracy in this experiment. Nevertheless, based on the actual effort and the original estimate, we believe that ML-estimates of less than 1 000 work-hours point to strong over-optimism. As can be seen in Table 4, nine out of ten participants in the T group had ML-estimates less than 1 000 work-hours, compared with only two out of ten in the NT group.

Non-technical (NT) roles		*Technical (T) roles*	
Role	**ML**	**Role**	**ML**
EM	1 200	PM	960
EM	1 550	PM	1 820
EM	1 850	PM	300
EM	547	PM	914
EM	2 286	PM	984
Design	1 500	Tech	960
Design	1 140	Tech	585
Design	1 260	Tech	220
Design	620	Tech	660
Design	1 500	Tech	900
Mean	**1 345**	**Mean**	**830**
Median	**1 380**	**Median**	**907**

Table 4: The Most Likely (ML) Effort Estimates

5.1. Test of Hypothesis 1

The results displayed in Table 4 indicate that the ML-estimates were systematically higher for the NT group than for the T group. We tested the strength of this difference by applying a statistical t-test. To use the t-test properly, the underlying distributions should be approximately normal. An Anderson-Darlington normality test [28] did not exclude normality ($p > 0.1$ for all relevant distributions), and a visual inspection of the distributions suggested that they were close to normal. There were moderate departures from the normal distribution, but this has generally negligible effects on the validity of both Type I and Type II error calculations [29]. The only exceptions are when there are substantially unequal variances and substantially unequal sample sizes, which was not the case here.

The results from the t-test (one-sided, assuming different variance) are shown in Table 5. We provide the actual p-values, as suggested by Wonnacott and Wonnacott [30], instead of pre-defining a significance level for rejection. To measure the size of the difference in mean values, we included Cohen's size of effect measure (d) [29]. The size of effect (d) was calculated as: d = (mean value NT group– mean value T group) / pooled standard deviation amongst groups NT and T.

	NT Group (mean work-hours)	*T Group (mean work-hours)*	*Difference in Mean*	*Pooled Standard Deviation*	*t-test (p-value)*	*Size of Effect (d)*
ML	1 345	830	515	486	0.02	1.1

Table 5: Differences between mean Most Likely (ML) estimates for the NT and T Groups.

The results in Table 5 show that the p-value was low (0.02). As indicated by the size of effect (d), the difference was considered "large", i.e., d was larger than 0.8 [29]. The robustness of the results is illustrated by a non-parametric statistical Kruskal-Wallis [31] test on the median estimates results, which gave a p-value less than 0.03. The analysis, therefore, supports Hypothesis 1.

6. Discussion

The observation that people in technical roles showed a higher degree of optimism in effort estimates does not necessarily imply that the roles or backgrounds were the direct *causes* of the difference. The effects may be due to a third variable. Perhaps difference in, for example, estimation strategy, has a direct effect on the estimates. This section discusses potential reasons for our observations. We investigate both personal differences due to length of experience and gender, and estimation differences such as perceived estimation skill, formal estimation training and estimation strategy. Interesting aspects related to the organizational structure, namely estimation goals and (lack of) feedback are also treated.

6.1. Different Length of Experience

A potential explanation of the observed difference is that the NT-group participants were more experienced. However, as shown in Table 6 there were no large differences between the groups in the measured types of experience.

Group	*Total Experience IT-Industry*	*Experience in Current Role*	*Experience in Providing Effort Estimates*
NT	5.1	2.0	1.2
T	7.5	3.8	1.7

Table 6: Subjects' Average Experience

Even if there were differences in length of experience, they could probably not explain the difference in estimation optimism, as shown by the following analysis. We divided the participants into two equally sized experience-groups, where each member of the "Long Experience"-group had greater experience than had all members of the "Short Experience"-group. A Kruskal-Wallis analysis of the difference in median estimates (ML) of the two groups yielded p=1.0 when the groups were divided according to total IT-industry experience, p=0.41 when divided according to experience in current role, and p=0.55 when divided according to estimation experience. This lack of relation between length of experience and estimation performance is similar to the results reported by Jørgensen and Sjøberg [18]. Interestingly, all three analyses exhibited the same trend with respect to the effect of experience: Shorter experience led to higher, i.e., more realistic, estimates. More studies are needed to investigate this relationship.

6.2. Gender Differences

Results reported by Henry [32] suggest that the level of optimism may depend on gender, i.e., that female estimators provide higher estimates. There were only three female participants in our study (all of them worked in the Design-track), and it is, therefore, unlikely that gender explains all of the variance. Nevertheless, it is interesting to observe that the three female participants had the highest estimates (ML) of those in the Design-track; their median estimate (1 500 work-hours) was much higher than that of the male participants (960 work-hours). The impact of gender on optimism of effort estimates is an interesting topic for further studies.

6.3. Differences in Perceived Estimation Skill

We asked the participants about their opinions on the company's and their own estimation skills. The possible answers ranged from 1 (very poor) to 5 (very good) in questions 1 and 2, and from 1 (low importance) to 5 (high importance) in question 3. Table 7 shows that there was almost no difference in the NT and T-group participants' opinions regarding effort estimates. Interestingly, both the T and NT-group participants viewed their own estimation skill as inferior (!) to that of the company (2.6 vs. 3.0 for the T group and 2.6 vs. 3.1 for the NT group).

Subjective Opinion on...	*NT (mean values)*	*T (mean values)*
1. The company's estimation skill (1-5):	3.1	3.0
2. Own estimation skill (1-5):	2.6	2.6
3. The importance of accurate effort estimates to the company (1-5):	4.4	4.6

Table 7: Subjective Opinions on Estimation Skill and Importance

The questionnaires were completed after the project was estimated. The participants may therefore have been influenced by the somewhat difficult estimation task of the experiment, and rated their skill lower than they would normally do. Still, this would affect both NT and T-groups to a similar extent. In addition, when we are measuring estimation precision, it is preferable that the estimation task influences the questionnaire answers, and not the other way around.

6.4. Differences in Formal Estimation Training

There were no differences regarding the training in estimation received at universities, or earlier or current employer. In the T group five out of ten, and in the NT group four out of ten, reported that they had received some estimation training.

6.5. Differences in Choice of Estimation Strategy

Based on an informal analysis of the estimation material completed by the participants, and of the videotape of the group discussion, it was possible to derive a classification of the estimation strategies applied. One should be cautious when interpreting these results, because we only had access to the participants' mental processes through the discussion, not, for example, through a think-aloud protocol. Dividing the strategies into the broad categories, top-down and bottom-up [33], we categorized the distribution of estimation strategies as shown in Table 8.

	NT Group (# participants)	*T Group (# participants)*
Bottom-up	2	8
Top-down	6	1
Uncertain	1	1
Mixed	1	0

Table 8: Distribution of Estimation Strategy

In top-down estimation, a project is reviewed as a whole, and the project's effort estimates are derived, for example, through a recall of the effort used on similar projects, i.e., the effort estimate is based on an "outside view" of the project. The project estimate can then be broken down into phases, activities, or tracks. As described in [20], major advantages of this method are its efficiency, and the fact that it can be combined easily with more formal analogy based estimation strategies [2].

In bottom-up estimation the project is divided into components or activities, where each component or activity is estimated individually. The total effort is then calculated as the sum of all the component and activity effort values that are identified. The advantage of this method is that each activity is estimated and is available for future project plans [33]. The risks are that activities can be easily forgotten, and that the risk budget covering unexpected tasks is not sufficiently large [20, 33]. One reason that technicians prefer bottom-up estimation may be that they are familiar with the method, since they will often be required to estimate each activity at later project stages.

The estimation material and the video-analysis of the subjects in the group-discussions indicate that a major reason for the optimistic estimates was forgotten activities. It was explicitly reported in the questionnaires by four T participants and one NT participant that missing activities were among the reasons for their estimates being lower than those of the other participants. Informal interviews with the Chief Project Manager of the company and other company staff also indicated that the main source of over-optimism was forgotten activities.

We conducted a Kruskal-Wallis test on the median values of those participants who followed a clear bottom-up or top-down strategy, to search for an eventual impact of the choice of strategy on the estimates. Table 9 shows that the p-value does not approach a significant level, but that the median value of the top-down based estimates are higher than those based on a bottom-up strategy.

Strategy	*Estimate*
Bottom-up (N= 10)	937
Top-down (N=7)	1 260
p-value	0.44

Table 9: Median Estimates Grouped by Strategy

We should not exclude the possibility that the effect of the estimation strategy is important, based on the statistical test alone, because:

- The power of the analysis described in Table 9 is low, due to large variance and few observations.
- The group discussions, the questionnaire answers, results from similar studies [34], and the direction of the measured difference in estimates, all point in the same direction, i.e., all available evidence suggests that the choice of estimation strategy may have an impact on the level of optimism in estimates.

The difference between the NT- and T-group participants' estimates was, however, much larger than the difference between the bottom-up and top-down strategy participants estimates, i.e., it is unlikely that choice of strategy can account for all of the difference between the NT and T groups.

6.6. Difference in Estimation Goals

The way in which a company measures its employees' competence can affect estimation performance. Estimation accuracy is only one out of many possible estimation goals. Other goals for the estimators may be, for example, to signalize personal competence, or to have sufficient time available to deliver a good project result.

As reported in earlier interviews [22], software professionals in the Design-track were not evaluated for "speed of delivery". More important evaluation factors were design quality and usability, which are more difficult to measure [35]. For designers, their estimation goals may be to receive as much time as possible within the projects limits, to achieve the best possible result. To achieve this goal, they prefer to provide the project managers with less optimistic estimates at an early stage.

The software professionals from the Tech-track, on the other hand, are expected to deliver functional programs, and "speed of delivery" is an important evaluation measure of their competence. One way to signal competence is, therefore, to provide low estimates.

This is the basic concept of Tesser's *Self-evaluation maintenance theory* [36]. If a person feels threatened, e.g., by other technicians potentially delivering low estimates on similar tasks, they will adjust their behavior, e.g., deliver very optimistic estimates, in order to preserve their self-esteem.

For software professionals in the PM and EM roles, the estimation goals may be more complex. A possible explanation for the optimism of PMs is that most of them were programmers, who had been promoted to PMs, i.e., they may still tend to think very much as programmers. In addition, a PM's estimates may be strongly influenced by the expectations from the management, e.g., a PM with low estimates may achieve an immediate positive feedback from the management. On the other hand, PMs may also desire high estimates on their project, as this increases the probability that the project is completed on time. The degree to which these two evaluation considerations affect estimates may depend on how much weight the PMs put on short-term (immediate positive evaluation from low estimates) versus long-term competence evaluation benefit (positive evaluation when the project is completed successfully due to realistic effort estimates).

The videotaped group discussion showed that almost all the EMs were concerned about how much the customer was willing to pay for the project. This should have led to a bias towards very low effort estimates, i.e., the opposite of what we observed. On the other hand, EMs are also responsible for the pay-off of the projects. Similarly to the PMs, the EMs may estimate with a trade-off in mind between short-term positive evaluation (get the contract) and long-term positive evaluation (make the company profitable).

6.7. Differences in Feedback

Lack of feedback prevents learning from experience when estimating effort. As described by Jørgensen and Sjøberg [22] the estimation feedback to the employees of the company is very limited. The feedback on total project effort may be better than that on individual tasks, i.e., it may be in favour of EMs and PMs. This may have caused the EMs to perform better than the designers and the PMs to perform better than the technicians in the experiment. However, these systematic role-dependent differences cannot explain the difference in optimism between the participants in the T and NT roles.

In the questionnaire on the participants' background, four of the ten T-group participants reported that they usually underestimated effort; four believed that they usually were on target, while two reported that they typically overestimated the effort. The corresponding

numbers for the NT-group participants were seven (underestimation), three (on target) and zero (overestimation). This indicates that the NT-group participants were more aware of their biases.

An important consequence of lack of feedback may be that goals other than estimation accuracy become more important. For example, when software developers know that they will not get any feedback on, or evaluation of, an effort estimate, greater weight may be placed on the short-term goal of "appearing skilled".

7. Conclusions and Further Work

The experimental results suggest that software professionals in technical roles and with a technical background were more optimistic in their estimates than those in non-technical roles and with a non-technical background. The reasons for this difference are more difficult to pinpoint, but there are probably several contributing factors. As discussed earlier, explanations such as length of experience, perceived estimation skill and formal estimation training seem to have no significant impact. Differences between gender and estimation strategy were difficult to investigate because of the small number of participants, but these variables may have an impact. Although it is impossible to measure and test statistically, evidence from other research and study of the company might indicate that estimation goal and (lack) of feedback could cause individual differences between the groups. In our opinion, the three most likely explanations for the observed difference are:

- *Estimation strategy.* Software professionals with technical competence typically use a bottom-up estimation strategy, while people in non-technical roles must base their estimates on "outside" properties of the project, and employ a top-down strategy. Our study indicates that this bottom-up strategy leads to estimation optimism, mainly because of forgotten activities. The indication is, however, not strong and we intend to conduct further studies where the difference in estimation strategy is the main topic. Observations that the recall of very similar, previously completed, projects is a pre-requisite for accurate top-down based effort estimation has already been made [37].
- *Estimation goals.* The evaluation of competence depends on the software professionals' role in the organization. It seems that programmers are perceived as more competent by others (and themselves) when they provide low estimates. The evaluation criteria of the non-technical and PM roles may be more context-

dependent and complex and we need more studies to examine whether, and how, different estimation contexts stimulate optimism, realism and pessimism in expert estimates.

- *Lack of Feedback*. Lack of estimation feedback seems to be typical for software development companies, and implies that other, conflicting goals become more important than estimation accuracy. In particular, we believe that the desire to appear skilled gains stronger weight when there is no feedback. Probably, a combination of these factors influences the estimates. It would, for this reason, be interesting to study the estimation behaviour of software professionals in situations where there is more feedback on estimates.

To summarize, there are reasons to believe that the company roles of software professionals affects estimation strategy and goals, which in turn affects software estimation optimism. We believe that company role, in combination with poor project feedback, explains most of the observed difference between software professionals with technical and non-technical backgrounds in our study.

It seems as if the choice of estimation strategy, estimation evaluation criteria, and feedback all seem to affect estimation accuracy. We encourage managers and experts to consider these elements in their estimation process to avoid underestimation. For example, if the estimators are not evaluated on estimation accuracy and apply a bottom-up estimation strategy, there may be a high risk of the estimates being too low.

Acknowledgements

Thanks to Simula Research Laboratory for funding the experiment, Dag Sjøberg for valuable comments, and to all participants and organizers at the company that participated in the study. Kjetil Moløkken-Østvold was funded by the Research Council of Norway under the project INCO.

References

1. Walkerden, F. and R. Jeffery, *Software cost estimation: A review of models, process, and practice.* Advances in computers, 1997. **44**: pp. 59-125.
2. Hughes, R.T., *Expert judgment as an estimating method.* Information and Software Technology, 1996(38): pp. 67-75.

3. Jørgensen, M., *A Review of Studies on Expert Estimation of Software Development Effort.* Journal of Systems and Software, 2004. **70**(1-2): pp. 37-60.
4. Moløkken-Østvold, K. and M. Jørgensen. *A Review of Surveys on Software Effort Estimation. 2003 ACM-IEEE International Symposium on Empirical Software Engineering (ISESE 2003)*. 2003. Frascati, Monte Porzio Catone (RM), ITALY: IEEE. pp. 220-230.
5. Kusters, R.J., M.J.I.M. Genuchten, and F.J. Heemstra, *Are software cost-estimation models accurate?* Information and Software Technology, 1990. **32**(3): pp. 187-190.
6. Vicinanza, S.S., T. Mukhopadhyay, and M.J. Prietula, *Software effort estimation: An exploratory study of expert performance.* Information systems research, 1991. **2**(4): pp. 243-262.
7. Mukhopadhyay, T., S.S. Vicinanza, and M.J. Prietula, *Examining the feasibility of a case-based reasoning model for software effort estimation.* MIS Quarterly, 1992(June): pp. 155-171.
8. Pengelly, A., *Performance of effort estimating techniques in current development environments.* Software Engineering Journal, 1995(September): pp. 162-170.
9. Kitchenham, B., et al., *An Empirical Study of Maintenance and Development Estimation Accuracy.* Journal of systems and software, 2002. **64**: pp. 57-77.
10. Heemstra, F.J. and R.J. Kusters, *Function point analysis: Evaluation of a software cost estimation model.* European Journal of Information Systems, 1991. **1**(4): pp. 223-237.
11. Lederer, A.L. and J. Prasad, *Software management and cost estimation error.* Journal of Systems and Software, 2000. **50**: pp. 33-42.
12. Ohlsson, N., C. Wohlin, and B. Regnell, *A project effort estimation study.* Information and Software Technology, 1998. **40**: pp. 831-839.
13. Walkerden, F. and R. Jeffery, *An empirical study of analogy-based software effort estimation.* Journal of Empirical Software Engineering, 1999. **4**: pp. 135-158.
14. Bowden, P., M. Hargreaves, and C.S. Langensiepen, *Estimation support by lexical analysis of requirement documents.* Journal of Systems and Software, 2000. **51**: pp. 87-98.
15. Atkinson, K. and M. Shepperd. *Using function points to find cost analogies*. *European Software Cost Modelling Meeting*. 1994. Ivrea, Italy.
16. Jørgensen, M. *An empirical evaluation of the MkII FPA estimation model*. *Norwegian Informatics Conference*. 1997. Voss, Norway: Tapir, Oslo. pp. 7-18.

17. Myrtveit, I. and E. Stensrud, *A controlled experiment to assess the benefits of estimating with analogy and regression models.* IEEE Transactions on Software Engineering, 1999. **25**: pp. 510-525.
18. Jørgensen, M. and D.I.K. Sjøberg, *Impact of experience on maintenance skills.* Journal of Software Maintenance and Evolution: Research and practice, 2002. **14**: pp. 1-24.
19. Lichtenstein, S. and B. Fischhoff, *Do those who know more also know more about how much they know?* Organizational Behaviour and Human Performance, 1977. **20**: pp. 159-183.
20. Boehm, B., *Software engineering economics.* IEEE Transactions on Software Engineering, 1984. **10**(1): pp. 4-21.
21. Whyte, G. and J.K. Sebenius, *The effect of multiple anchors on anchoring in individual and group judgment.* Organizational behaviour and human decision making, 1997. **69**(1): pp. 75-85.
22. Jørgensen, M. and D.I.K. Sjøberg, *Impact of software effort estimation on software work.* Journal of Information and Software Technology, 2001. **43**: pp. 939-948.
23. McDonald, A. and R. Welland. *Web Engineering in Practice*. *Proceedings of the Fourth WWW10 Workshop on Web Engineering*. 2001. pp. 21-30.
24. Wiegers, K.E., *Software process improvement in web time.* IEEE Software, 1999. **16**(4): pp. 78-86.
25. Reifer, D.J., *Web development: estimating quick-to-market software.* IEEE Software, 2000. **17**(6): pp. 57-64.
26. Moløkken, K. and M. Jørgensen. *Software Effort Estimation: Unstructured Group Discussion as a Method to Reduce Individual Biases*. *The 15th Annual Workshop of the Psychology of Programming Interest Group (PPIG 2003)*. 2003b. Keele, UK. pp. 285-296.
27. Moses, J. and J. Clifford. *Learning how to improve effort estimation in small software development companies*. *24th Annual International Computer Software and Applications Conference*. 2000. Taipei, Taiwan: IEEE Comput. Soc, Los Alamitos, CA, USA. pp. 522 - 527.
28. Christensen, R., *Analysis of variance, design and regression. Applied statistical methods*. 1998: Chapman & Hall/Crc.
29. Cohen, J., *Statistical power analysis for the behavioral sciences*. 1969, New York: Academic Press, Inc.

30. Wonnacott, T.H. and R.J. Wonnacott, *Introductory statistics*. 5th ed. 1990: John Wiley & Sons.
31. Siegel, S. and N.J. Castellan, *Non-parametric Statistics for the Behavioral Sciences*. 2nd Edition ed. 1988: McGraw Hill College Div.
32. Henry, R., *The effects of choice and incentives on the overestimation of future performance.* Organizational Behavior and Human Decision Processes, 1994. **57**: pp. 210-225.
33. Kitchenham, B., *Software Metrics: Measurement for Software Process Improvement*. 1996, Oxford: NCC Blackwell.
34. Buehler, R., D. Griffin, and M. Ross, *Exploring the "Planning fallacy": Why people underestimate their task completion times.* Journal of Personality and Social Psychology, 1994. **67**(3): pp. 366-381.
35. Rosenfeld, L. and P. Morville, *Information architecture for the world wide web*. 1st. ed. 1998, Sabastopol: O'Reilly and Associates, Inc.
36. Aronson, E., T.D. Wilson, and R.M. Akert, *Social Psychology*. 3rd. ed. 1999: Addison-Wesley Educational Publishers Inc.
37. Jørgensen, M., *Top-Down and Bottom-Up Expert Estimation of Software Development Effort.* Journal of Information and Software Technology, 2004. **46**(1): pp. 3-16.

Appendix I:

User Requirements

This is the requirement specification for the project, as delivered by the customer.

The customer

The customer is the producer of an established technical encyclopaedia that numbers 800 editions with 10 000 illustrations and 1 600 tables. They have 20 000 subscribers. The publication frequency is low, with two shipments each year, each containing several magazines.

The magazines exist both on paper and CD-ROM. The CD-ROM contains some extra features, and there are plans to add other sources of information.

There is also a simple intranet version of the CD-ROM that is used internally and by a handful of existing customers.

All documents are created in MS-word, and approved and converted to HTML by a central unit. They have no need for a very complicated CMS-system.

Status

The starting point for a web version is the existing CD-ROM and intranet based system. The primary goal is to build on the functionality of these systems, but also to offer the encyclopaedia commercially over the Internet, both to companies and individuals. It is natural to use this opportunity to look at the possibilities for a new medium, as well as revising existing production and administration routines.

All documents that will be used exist on the CD-ROM in HTML versions, and can be copied directly to the website. No changes are needed. Design is of less importance.

All the users of the site are expected to be technical competent people, with experience and knowledge of the CD-ROM and/or paper versions.

Desired functionality

This is the required functionality.

1. Basis functionality (searching for and displaying information)

Display documents in HTML-format (document including local table of contents (TOC) in magazine)

Navigation through an expanding TOC

Navigation through an index

Free search (Expansion of the existing version)

2. Downloading of documents and pictures

The system must allow the downloading of documents in other formats.

PDF versions of documents for better prints where such exist

Figures in high-resolution bitmap (TIFF)

Figures in vector format (DWG)

Displaying of video-clips

The system must handle the fact that not all documents and figures exist in all formats.

3. Extranet functionality

Only paying subscribers shall have access to the service. For companies this can be implemented by access-limitation on a net-level (IP). Company customers can then skip logon procedures with usernames and passwords. An alternative method is to use personal subscriptions. The system must be able to handle different types of subscriptions that grant different degrees of access to the system.

4. Trade solution

Possibility of subscribing via the web

Possibility of buying a single magazine for downloading or delivery by mail

Possibility of paying online by credit card or other forms of payment

5. Demo-/sales-version

The system shall have an open part, granting access to some functionality, such as navigation and search, and limited content. This functionality should be combined with a function that allows the purchase of single magazines for downloading.

6. Reply service

This contains an overview of the FAQ. The answers should have links to magazine editions with extensive information. The user is checked, and receives the option to log in, buy a single magazine, or buy a subscription.

7. User adjustment and information

Possibility of having personal settings for each user

Possibility for users to register own comments to the magazine

Possibility for the editors to publish comments to the magazines

Possibility for users to give feedback directly from a magazine, reporting errors etc.

Discussion forum tied to magazines

8. Administration of users

The administration system stores information on all customers, both subscribers and buyers of single issues. The administration system must monitor the use of the system. One must be able to extract different types of statistics, as well as blocking users who abuse the system or fail to pay their subscription.

9. Integration with existing administrative systems

Subscriber- and logistics system, Agresso.

Customer and support, Superoffice

Degree of integration must be based on cost/value aspects.

10. Adjustment of the production system

The production system is based on a personally developed database, containing a parser that translates documents from MS Word format to HTML. The system must be adapted to a new medium and a new system. Other possible changes to consider:

Parsing of word documents to XML instead of HTML, which will add to the system's flexibility.

Expansion of the database to support the administration of manuscripts.

Adapt the base for the production of additional documents used in the printed issue (TOC, index list, overview of new, changed and expired magazines etc.)

11. Interface with other systems

By implementing the magazine in a web setting, integration with other systems should be considered. An interface (API) for communication with other systems (Using SOAP, XML etc.) should be defined. It must be possible to link to documents by URL.

12. Choice of technology

The goal is to use already known technology. This is to handle development and changes with internal resources.

OS: Windows 2000

Web server: Internet Information Server w/ASP

Database: Microsoft SQL Server

Languages: Visual Basic, NET technology

13. Hosting

The system is to be installed on the client's servers.

Figure 1: Front page Example

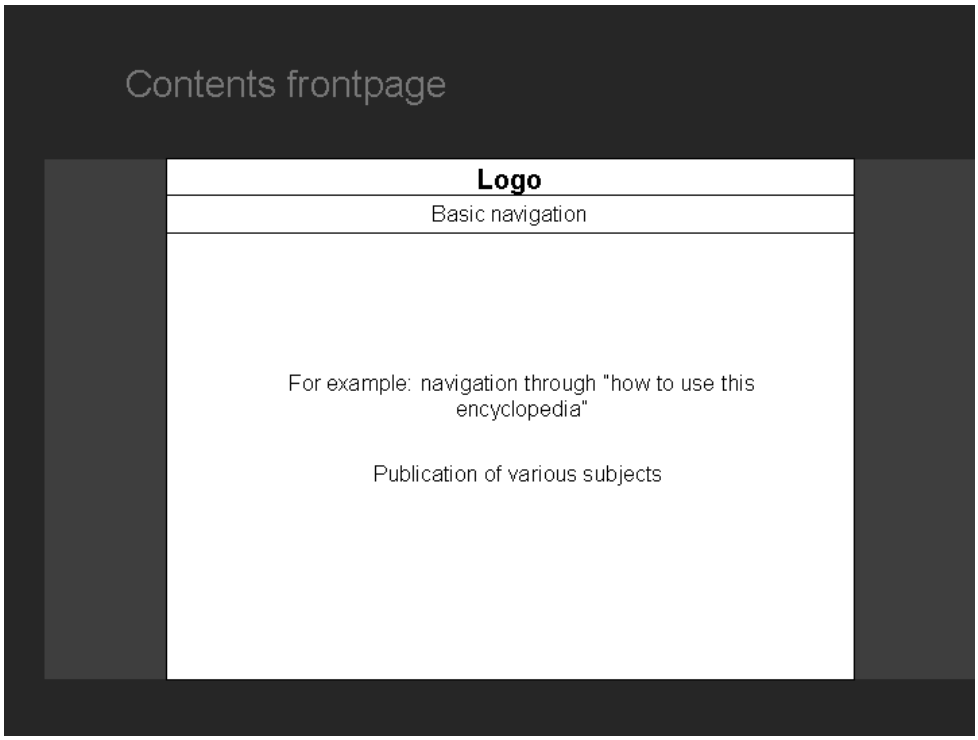
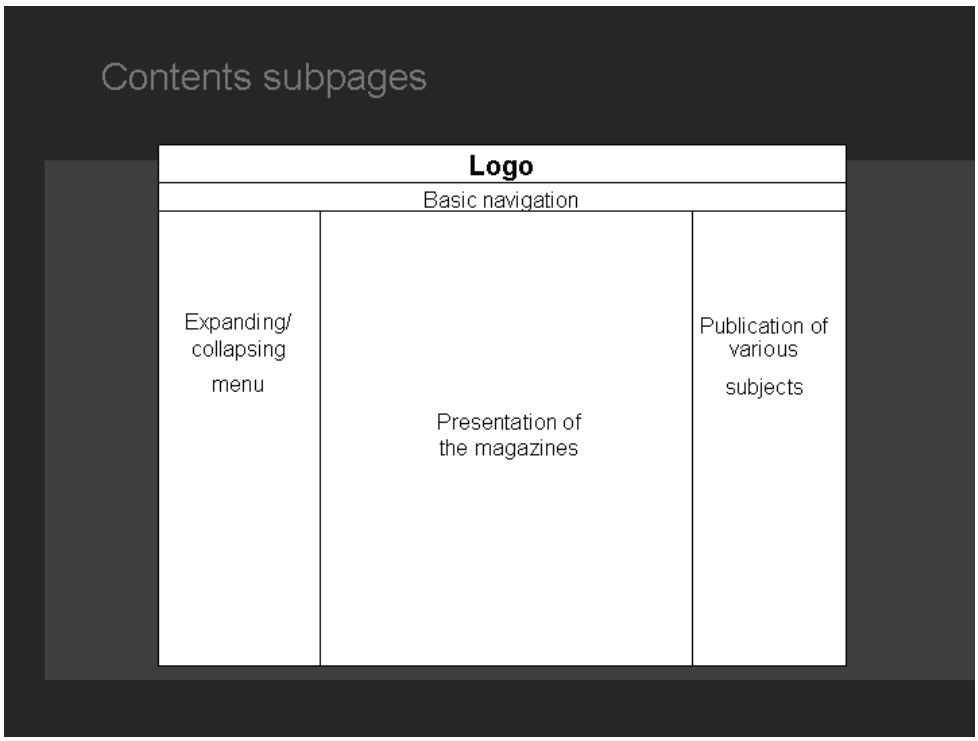


Figure 2: Sub page Example



Paper V:

Group Processes in Software Effort Estimation

Kjetil Moløkken-Østvold and Magne Jørgensen.
Simula Research Laboratory.

Journal of Empirical Software Engineering, 2004, Volume 9, Issue 4, pp 315-334.
Guest Editors: Dr. Marian Petre, Prof. David Budgen and Dr. Jean Scholtz

---

Abstract: *The effort required to complete software projects is often estimated, completely or partially, using the judgement of experts, whose assessment may be biased. In general, such bias as there is seems to be towards estimates that are overly optimistic. The degree of bias varies from expert to expert, and seems to depend on both conscious and unconscious processes. One possible approach to reduce this bias towards over-optimism is to combine the judgments of several experts. This paper describes an experiment in which experts with different backgrounds combined their estimates in group discussion. First, twenty software professionals were asked to provide individual estimates of the effort required for a software development project. Subsequently, they formed five estimation groups, each consisting of four experts. Each of these groups agreed on a project effort estimate via the pooling of knowledge in discussion. We found that the groups submitted less optimistic estimates than the individuals. Interestingly, the group discussion-based estimates were closer to the effort expended on the actual project than the average of the individual expert estimates were, i.e., the group discussions led to better estimates than a mechanical averaging of the individual estimates. The groups' ability to identify a greater number of the activities required by the project is among the possible explanations for this reduction of bias.*

Keywords: Software development, effort estimation, expert judgment, group processes, expert bias.

1. Introduction

"When you lie about the future, that's called optimism, and it is considered a virtue. Technically speaking you can't "lie" about the future because no one knows what will happen. When you apply this unique brand of optimism (not lying!) at work, that's called forecasting."

- Scott Adams [1]

Improving the quality of effort estimation is one of the great challenges for software project management. Across a wide range of software projects, from web applications to time-critical financial or medical systems, poor effort estimation is observed, and the problem seems to increase with project size [2]. To account for this, several formal methods and processes to support estimators, such as COCOMO II [3], ANGEL [4] and WEBMO [5] have been proposed. This paper, however, focuses on the estimation of effort by *experts*. This seems to be the most common estimating method [6], and is employed over a wide range of software projects. One of the main problems with expert estimation, however, is that it is no better than its participants, and hence is subject to their individual biases [7, 8] and political pressure brought to bear by the company [8]. A possible method of reducing the risk of unwanted influence on the estimates is to use group discussion as an estimation method. This is not a new idea, but it seems to have been neglected in the empirical research on software engineering. In other fields, however, there is continuous ongoing research on how to best combine the opinions of several experts [9].

We believe that new companies in particular, which have limited historical data upon which to draw for experience, may benefit from using groups to improve their effort estimation process. This paper describes a controlled experiment on the performance of estimation groups in a web-development company that uses no formal estimation process or project experience database. A typical web-development project has several characteristics that distinguish it from traditional software development projects [5]. These characteristics include project size, development approach, processes employed and people involved. Some of the differences are due solely to the nature of the technology involved, but others may be related to company size and maturity. Most web-development companies are small compared with traditional software development companies, and smaller companies may

face additional estimation challenges, such as access to domain experts and process management [10].

In our experiment, twenty individual experts submitted project estimates that ranged from 220 to 2286 hours, with an average of 1088 hours. Estimates from five groups of four experts each ranged from 1100 to 2251 hours. The actual effort expended on the project was 2365 hours, indicating that some, but not all, of the bias towards optimism was eliminated by group discussion. In essence, the results reported in this paper will apply to similar companies and estimation contexts, but the basic idea of using unstructured group discussion to reduce individual bias may also be transferable to more traditional development projects and other estimation contexts. The company studied, our experiences from this and other companies, and the extent to which the study has external validity, will be discussed in later sections.

The remainder of this paper is structured as follows: (2) Background for Expert and Group Estimation, (3) Hypotheses, (4) Research Method, (5) Results, (6) Discussion and (7) Conclusion and Further Research.

2. Background for Expert and Group Estimation

Expert judgment is a commonly employed method for estimating the effort required to complete software projects. As reported in a review of studies on expert estimation by Jørgensen [6], several independent surveys [11-16] rate it to be the preferred method among professional software developers. Although expert estimation is commonly used, it is probably not because of its precision that the method is favoured. In fact, expert estimation seems to be just as imprecise as the use of formal estimation models [6]. However, expert estimates can be especially useful (and often the only option) for companies that lack documented experience from earlier projects [17] or have limited estimation resources [10]. This is often the case for young and/or unstable organizations. Both these characteristics apply to most web-development companies, including the one we studied and report on in this paper.

A search on the terms “software effort” or “size estimation” in the leading software engineering journals, IEEE Transactions on Software Engineering, Journal of Systems and Software, Journal of Information and Software Technology and Journal of Empirical Software Engineering, yielded slightly more than 100 relevant papers. Of these, only 16 had expert estimation as a topic, and only one of them had group estimation as a topic. In that

paper, Taff and his colleagues present a structured model for expert estimating in groups, called Estimeetings [18], but do not compare these estimates with individual or other group based estimates, only with actual effort expended. In other words, papers in the leading journals for software engineering did not contain a single study on individual versus group-based software effort estimation.

Group expert estimates may be categorized according to two different characteristics, though other categorizations are possible.

The first characteristic concerns the involvement of the estimators, and offers two possible approaches: (i) There are designated estimation groups [19], whose only objective is to estimate the project, and do not participate in the development process. (ii) Employees who are likely to develop the project are responsible for the estimates. Both approaches have advantages and disadvantages. A separate estimation group may be much less prone to personal or political biases, and more likely to improve their estimation skill over time [19]. On the other hand, to have such a designated team would require a large organization, and good internal communication. Expert estimators, who are likely to implement the solution themselves, will probably get to know the project better than anyone else, and may have a higher motivation for a thorough project analysis [8]. For smaller organizations, this approach may be the only possibility due to financial and resource allocation restrictions. A recent study by Jørgensen [20] found that estimation accuracy improved when the estimator participated in project development. This is supported in previous research [21]. However, one of the main problems associated with not using independent estimators is that personnel with high stakes in the project, e.g. project managers, may provide unrealistic estimates in order to get their project approved.

The second characteristic is a structured versus an unstructured approach. An elaborate method of reducing problems in groups related to company politics is to employ the Delphi technique [22], which is often recommended in management papers [23]. The Delphi technique does not involve face-to-face discussion, but anonymous expert interaction through several iterations, supervised by a moderator until an agreed-on majority position is attained. A modification of this technique, which includes more estimation group interaction, was developed by Barry Boehm and his colleagues, and labelled the Wideband Delphi technique [24]. This technique is a hybrid of unstructured groups and the traditional Delphi method. As in the Delphi technique, there is a moderator (labelled coordinator), that supervises the process and collects estimates. In this approach, however, the experts meet for group discussions both prior to, and during the estimation iterations. This approach has

been suggested as an effort estimation method in books and articles on software metrics [25], software process improvement [26], project management [27] and effort estimation [8]. The Estimeetings process [18] involves a series of group meetings, where at some stage, parts of the requirement specification is handed down to several individual estimators with in-depth knowledge of the problems. It is a complicated process that require several weeks. It was specifically designed for one large project, and we have not found any evidence that it has been used anywhere else.

To the best of our knowledge none of the Estimeetings, the Delphi or the Wideband Delphi techniques has been subject to extensive empirical research in a software engineering context during the last 25 years. We are aware of only one study that describes a company which employed the Delphi approach [28]. By contrast, we know from experience that many software organizations apply unstructured group discussions in their work leading to software development effort estimates.

3. Hypotheses

Some of the scepticism towards group-based effort estimation may be attributed to misinterpretation of the research from other research areas, e.g., as presented in some introductory textbooks in psychology. Many of these are quite extensive in their coverage of the possible dangers of group processes [29-31]. Such books may be misleading, if the described results are not understood properly.

The terms group polarization and choice shift refer to two similar, and often confused, concepts in group psychology. These related concepts concern, among other things, how an initially optimistic or risky decision can be rendered even more optimistic or risky by group discussion. Zuber, Crott et al [32] define choice shift as the difference between the arithmetic average of the individual decisions and the group decision. Group polarization is defined as the difference between individual pre- and post- group discussion responses. Studies conducted on group decision making have found choice shift and group polarization effects in decisions made in a range of different areas, from burglary to management [32-36]. Some of this knowledge may have been oversimplified during the transfer from psychology to other professions, such as project management. The literature in these other professions often states the dangers of group interaction, but not *why* or in *which* situations these dangers apply. This may have resulted in an incomplete picture, in which valuable group work aspects such as motivation [37] and information sharing [23] have been omitted.

We find evidence of this omission in software management textbooks, where both separate estimating groups [19] and methods such as the Wideband Delphi technique [24, 25] are described as countering choice shift and group polarization, with no further explanations of the phenomena. For example, Boehm [24] states that "…group members may be overly influenced by figures of authority or political considerations."

Fortunately, there are research communities that have investigated the different properties of group decision making. According to Kernaghan and Cooke [38], the engineering management community has gradually accepted that the output of groups is likely to be superior to its average member. The forecasting community also constantly addresses best practices for combining expert judgements [9]. Textbooks specialized in group psychology [39, 40] also present a balanced view on when and how group processes can be beneficial, depending on the task and individuals involved.

A review of the literature on the Delphi technique in forecasting [37] suggests that it, on average, outperforms unstructured groups in which group members discuss and interact freely. However, the review has also shown that there are tasks for which unstructured groups are better suited. In some situations, it is possible that extra information and group motivation exist in an unstructured group, and this can facilitate the process and enable it to surpass a Delphi group in performance [37]. Perhaps typical software estimation processes represent such situations. In an estimation process, there may be several experts who contribute different project experiences and knowledge. Such experiences can more easily be shared in a face-to-face group than through a moderator, as in the Delphi technique.

Earlier reviews of the literature and experiments have concluded that it seems to be less important which combination method, from a set of "meaningful" methods, is used [41]. It may, for example, not matter much whether simple averaging, unstructured groups, the Delphi technique or other combination methods are used. Other factors, such as cost and political issues, should determine which combination method to employ [41].

Several software engineering textbooks [24, 42] and papers [7, 8, 23] point out that forgotten tasks are among the major obstacles to successful estimation by experts, especially when employing a bottom-up estimation approach, i.e., the decomposition of a project into activities and the estimation of each activity individually. A group approach to estimation will help to remove this obstacle, because several estimators will identify at least as many activities as the best single estimator alone. This will be especially true if estimates from experts with different company roles and experiences are combined in group discussion.

In sum, previous findings lead us to believe that unstructured groups can be used to reduce individual estimation optimism. The latter belief is evaluated through experimental testing of the following hypotheses:

H1: Group effort estimates are, on average, less optimistic than the average of the experts' individual effort estimates.

H2: Individual effort estimates are, on average, less optimistic after group discussion with other experts than before group discussion.

4. Research Method

The research presented in this paper attempts to follow the guidelines suggested by Kitchenham, Pfleeger et al. [43], which includes specifying as much information about the organization, the participants and the experiment as possible, in addition to the complete experimental results. The rationale for including so much information is to ensure that the study is easy to replicate, and that the results are appropriately interpreted and transferred.

4.1. The Company Studied

The company studied is a web-development company, and operates as an independent contractor that develops a wide range of complete solutions for its customers. At the time of the study the company had about 70 employees. The employees (excluding administration and support staff) were allocated to the following four business roles: Engagement Manager/Sales and Client Responsibility (EM), Project Manager (PM), User Analyst/Designer (User) and Technical Programmer (Tech). These roles are similar to those described by McDonald and Welland [44]. Average participant experience in the IT-business was 6.3 years, in their current role 2.7 years and with effort estimation 1.5 years. Half of the participants were educated to at least Master's level, while the rest had Bachelor's or comparable degrees.

The organization had no formal estimation procedure, their projects had short development cycles and the development processes involved were ad-hoc. These are all typical characteristics of web-development companies [5, 44]. The company based its estimates on expert judgment. A project estimate was most often provided by the person(s)

responsible for the project. This is, to the best of our knowledge, a common situation in web-development companies, and small companies in general [10].

4.2. The Estimation Task

Twenty participants were selected at random from the company; five from each of the four company roles (EM, PM, User and Tech). Each participant was required to estimate the most likely effort needed to complete a project, based on a requirement specification including some screenshots from a CD-ROM. The specification, i.e., the document describing the software to be developed, was taken from a project currently under development by the company. The complete requirement specification, as presented to the participants, is enclosed as Appendix I.

The project and the customers were anonymous, and none of the participants had any knowledge about them. The project was the development of a "publication solution" for a technical magazine. It was medium to large, compared to the average size of projects developed by the company. The estimation instructions stated that the participants in the experiment should behave as realistically as possible. The participants were also informed that a project crew had not been selected, and that they should base their estimates on average company productivity. These measures were undertaken in order to reduce any political biases [19, 45], e.g., that the estimates would be influenced by a belief about what would be the right price to win the contract. What we wanted to investigate was the group discussion effect on the most likely estimate. During the sessions, the experimenter was present to answer questions and take notes.

The experiment had the following steps:

- The participants developed their individual estimates of the effort needed to produce the software during a 45-60 minutes estimation session.
- After the individual session, the participants formed groups, with one Engagement Manager (EM), Project Manager (PM), User Analyst (User) and Technical Programmer (Tech) in each group.
- Each group was given about 60 minutes to agree on an estimate for the same project as in the individual estimation session. These sessions were also videotaped to ensure thorough analysis of the group discussions.

- After the group decision, each participant was asked: "After an individual assignment and following teamwork, what is now your personal opinion about the estimation assignment".

During the experiment each participant completed two questionnaires, in which he provided background information about his experience, education, estimation training and skill level, and gave his comments on the experiment.

5. Results

The original individual estimates, the group estimates, and the individual opinions after group discussion are displayed in Table 1. There were no participant dropouts or incomplete responses. All statistical calculations were done with the package MINITAB[2] 13.3 for Windows.

	Group A		*Group B*		*Group C*		*Group D*		*Group E*	
Business role	*Before*	*After*	*Before*	*After*	*Before*	*After*	*Before*	*After*	*Before*	*After*
EM	1200	1000	1550	1500	1850	1500	547	1000	2286	2800
PM	960	1200	1820	1500	300	1550	914	1400	984	2200
User	1500	1200	1140	1500	1260	1500	620	1000	1500	2000
Tech	960	1000	585	1400	220	220	660	1000	900	2000
Average	1155	1100	1273.8	1475	907.5	1192.5	685.3	1100	1417.5	2250
Group	1100		1500		1550		1339		2251	
Actual effort	2365									

Table 1: Individual Pre-group (Before), Group and Individual Post-group (After) Estimates.

The group estimate was less optimistic than the average expert opinion in four out of five groups. An analysis was performed with a paired t-test [46], as suggested in similar research on choice shift [47]. Since the hypothesis suggests a direction of effect (groups are less optimistic than average individual experts), the paired t-test was one-sided. The result is displayed in Table 2.

[2] http://www.minitab.com

	Group average	*Individual average*	*Difference*	*Pooled StDev*	*p-value*	*Size of effect (d)*
ML	1548	1088	460	368	0.024	1.25

Table 2: Average values for shift between individual and group estimates.

We provide the actual p-values, as suggested by Wonnacott and Wonnacott [46], instead of pre-defining a significance level for rejection. To measure the size of the difference in average values, we included Cohen's size of effect measure (d) [48], where d = (average value group – average value individuals) / pooled standard deviation amongst groups and individuals. The analysis of possible choice shift on the estimates gave a discernible result (p=0.024), and a large effect (d=1.25).

Group polarization is, as described earlier, the difference between individual responses made *before* the group session and the responses made *after* the group session [32]. Both median and average individual estimates were less optimistic in the post-group answers than in the initial responses. The median values increased by 478 hours, from 972 to 1450. The average individual estimates increased by 336 hours, from 1088 to 1424. A one sided, paired t-test on the before and after values yielded a discernible result (p=0.003), and a medium (d=0.62) effect size (Table 3).

	Average individual estimates after group discussion	*Average individual estimates before group discussion*	*Difference*	*Pooled StDev*	*p-value*	*Size of effect (d)*
ML	1424	1088	336	544	0.003	0,62

Table 3: Average Values, p-values and Effect Size (d) for a Change in Individual Optimism.

6. Discussion

The actual effort expended on the project was 2365 work-hours, that is, most participants gave an estimate that was much too low compared with the actual effort expended. However, the actual effort of a new project based on the same specification may have used less (or more!) effort than the completed project, i.e., there are many possible effort usage outcomes of a project based on the same specification. The actual effort expended to complete the project cannot, for this reason, be taken as more than an indication of estimation inaccuracy in this experiment. Nevertheless, based on the actual effort and the

original company estimate (1240 hours), we believe that estimates of less than 1000 work-hours point to strong over-optimism. As can be seen in Table, 11 out of 20 of the original individual effort estimates indicated a workload of less than 1000 hours. The variation among the individual estimates shows that in reality, opinions regarding the same project may differ by a magnitude of up to ten!

There seems to be a tendency for both the group decisions and the individual post group discussion decisions to be less optimistic than the original estimates. None of the group decisions, and only 1 out of 20 individual post-group estimates, was less than 1000 hours.

The experiment conducted has several properties that need to be viewed in light of previous research on group processes, applications for researchers and practitioners, and experimental validity.

6.1. Group Processes

When discussing the use of groups in the context of software effort estimation, it is essential to understand what kind of task effort estimation is. Effort estimation is a complex task in which certain properties, such as "quality", are difficult to measure. In his book on group processes, Rupert Brown [39] differentiates between "group productivity" and "group decision making". In his view, group productivity concerns tasks where there exists a measurable performance, while group decision-making concerns tasks where it is impossible to measure performance.

Regarding group decision making, we have already described some of the previous research on choice shift and group polarization. The literature on group processes [29-31, 39, 40] and software management [19, 24, 25], frequently warns about the possibility that group pressure (e.g. an unspoken "competition" to appear as the most risky or efficient programmer) and political preferences (e.g. a management that demands optimistic estimates), could influence group decisions unreasonably. Observation of post-group opinions that significantly differed from the group responses might indicate that the participants merely complied, for example, with authority. In our experiment, however, the individual post group estimates were much closer to the group decisions than they were to the original individual estimates. This, together with responses in the submitted questionnaires, indicates that the participants may have been mostly influenced by the arguments and extra information presented in the group discussion.

The nature of the task involved in our experiment differs significantly from those in most previous studies on group decision making, which typically ask groups of randomly selected people to decide on choice dilemmas, or professionals in an occupation to determine risk associated with different tasks. Those studies suggest that initially risky or optimistic decisions become more extreme after group discussion. Our study, however, yields the opposite result. Possible explanations from a group decision making perspective, is the diverse background of our participants, their shared commitment, and how the group politics were handled.

Our groups were able to identify more activities than did our individuals alone. It was explicitly reported in the questionnaire, by five of the participants in our experiment, that having forgotten activities was the main reason for their estimates being lower than the group estimates. The groups sometimes even identified necessary activities that none of the individuals had reported. In the experiment, none of the participants in each group had the same company role. This may improve the quality of the group [23], as long as they share terminology and an understanding of the problem [22]. This applies to the participants in our experiment, since they frequently work on the same projects in the same company. Experts with different backgrounds consider the same project from different angles, and are probably able to identify more activities than experts with similar backgrounds. It is possible that groups of experts with similar backgrounds would show less reduction of optimism in their estimates [49].

During the video-analysis, we observed how all the groups behaved in a way which allowed the opinions of all participants to be heard and that differing views were discussed openly. They regarded the assignment from a professional point of view, and there was no apparent peer pressure to be either “risky” or “conservative”. All groups resolved the task by a consensus estimate, and there were no incidents of a majority decision, e.g. through a vote. As seen in table 1, most participants retained the group estimates when they ware asked to re-estimate the tasks individually after group discussion. This can indicate that their personal opinion had had been acknowledged by the group.

In accordance with Brown’s productivity aspects, might it not also be possible to measure some kind of productivity in the estimation process? Thus far, we have, in the main, considered the process of estimating effort as a whole. Given the complexity of estimating the effort required to complete software development projects, an analysis of the different parts of the estimation process might yield a better understanding of how far productivity can be measured. Such analysis is, of course, a challenging endeavour and one

upon which it would be foolhardy to embark in the present paper, but from a cursory examination it should be clear that some parts of the estimation process are measurable. For example, it would seem to be possible to measure the successful identification of the tasks that need to be performed to complete a software project.

The task of identifying project activities is related to the classic brainstorming process. A brainstorming process often involve several individuals seated together in order to generate input to a specific domain, e.g. a name for a new washing detergent. Brown [39] reports studies that have found brainstorming procedures to be ineffective, due to both social and coordination problems. On the other hand, if the participants first prepared themselves, productivity increased significantly. Therefore, it is essential to consider not only what projects or part of projects to estimate in groups, but also how this group process is implemented. In our experiment, the individuals prepared themselves during the individual estimation. They first identified tasks they believed were necessary before the group discussion, during the individual estimation session. Since each group consisted of personnel with different roles, they also emphasized different activates. Later, in the group session, they were able to combine their activities, as well as identifying entirely new ones, in order to construct a complete project break-down. This break-down was then used as a basis for the group estimate.

In a more general sense, group productivity can exceed the sum of the individual performances if some kind of group motivation exists [39]. In our experiment, and in actual development projects, the participants should be motivated to perform well together with their colleagues.

6.2. Implications for Researchers and Practitioners

It is difficult to discuss our findings in the light of actual estimation practices employed by software professionals. To our knowledge, no surveys have been conducted that analyze the extent to which groups of experts are used in effort estimation. A recent review of all known surveys on software effort estimation [50] found that expert judgement is by far the most common method used to estimate software projects. By comparison, formal models, such as COCOMO or FPA-based, are not used to any great extent. The surveys analyzed in the review, however, failed to elicit how expert estimates were made. None of them asked, for example, whether groups of experts were used at any stage of the process.

From all the surveys, experiments and case studies on software estimation that we reviewed, we found only one that identifies a formal group processes actually used in the industry [28]. In a case study of 145 projects at an outsourcing company, it was found that 3 out of 145 projects were estimated by using the Delphi procedure. By comparison, 104 projects were estimated by the project manager alone, and were defined as expert opinion. We found no independent reports of actual use of the Wideband Delphi or Estimeetings procedures discussed earlier.

Even though there are few descriptions in the literature of group-based estimates (formal or informal) this does not necessarily mean that no companies conduct estimates in groups. From our experience as field researchers, and through review of other studies, we have found that the terms "expert judgment", "expert decision" or "expert estimate" include a wide range of estimation procedures. These expert-based procedures may include varying degrees of group interaction, from none (e.g. a project manager estimates the whole project) to full group interaction (e.g. as in our study). In between these alternatives, there exist other methods to include several experts, e.g. a manager can ask subordinates to each estimate parts of projects, and the manager then adds the estimates for a project total.

In our study, the group-based estimates were less optimistic than the average estimate of individual experts. This may, however, not be the case for other companies or projects. Therefore, it is necessary that the individual companies themselves analyze their projects and decide which projects, and which stages in the estimation process, are suited for group collaboration. As discussed earlier, a typical stage at which the views of several experts may be beneficial is when the requirements are mapped onto different project activities.

The group process is probably more valuable than mechanical combination of estimates because then the experts discuss not only their estimates, but also the assumptions they made when calculating them [51]. Still, it is not unlikely that similar results would have been found had the unstructured group discussions been replaced with more structured group processes, e.g., the Wideband Delphi technique. The combination of several experts' opinions seems to be beneficial in most cases. The way in which expert opinions should be combined should be decided relative to estimation context factors, such as type of process and project [41]. Projects with high political stakes may not be suited for unstructured group estimation at all, and more formal procedures may be appropriate. Structured approaches, such as the Wideband Delphi technique, have qualities different from unstructured approaches [37], and may, for example, be better suited when the level of personal disagreement is high, or political prestige is involved.

Companies may also develop their own Work Breakdown Structures (WBS) and/or estimation checklists to achieve some of the benefits that we found accrue from group discussion, e.g., individual estimators have a lower risk of forgetting activities when using an extensive WBS or high quality checklists. The use of a group collaboration estimation method can then be reserved for especially challenging projects, for example, projects involving new technology and business areas.

6.3. Experimental Validity

The main threat to validity of the study may be that the experimental setting distorts the realism of the estimation process. The following analysis, however, suggests that at least the outcome of the estimation process was realistic. The average project estimate provided by the experiment participants (1088 hours) did not deviate substantially from the actual effort estimate made by the company (1240 hours). We must, however, be aware of the large individual deviation, with estimates from 220 to 2286 hours.

There are also validity threats related to the lack of customer contact and limited time available present in our study. For effort estimates applied as input to a bidding process, however, this was a common situation in the company, which developed more than 500 estimates in the year 2001.

The main problem regarding transfer of results to other organizations may be that the participants in the study were from the same company. The participants were so chosen for practical reasons, but our experiences with other companies lead us to believe that the company we studied is similar to many other web-development companies, both in size and (lack of formal) estimation process. Our observations of the company also indicate that they are similar to other small companies who employ an ad-hoc estimation approach, as described by Moses and Clifford [10].

It may be a threat to validity that only one project was used. The project chosen may, for example, have been especially difficult to estimate, and the optimism reduction may not have been of the same magnitude in other "easier" projects.

The impact from the threats to validity means that our findings may be applicable mainly to small companies with lack of formal estimating processes, when estimating more than average "difficult" projects with limited information and strong time restrictions, i.e., estimation situations typical for web-development companies when providing a project bid.

7. Conclusions and Further Research

The findings in our study are that group estimates and individual estimates after group discussion were less optimistic and more realistic than the individual estimates before group discussion. The main sources of this increase in realism seem to be identification of additional activities and an awareness that activities previously identified may be more complicated than was initially thought.

The unstructured group effort estimation approach presented here may be a simple and inexpensive approach for companies to improve the precision of their estimates.

Our contribution to the software industry is not to invent a new procedure, since group-based estimation procedures are probably employed by a wide range of companies all over the world. We seek, instead, to counterbalance the view on group processes presented in many papers and textbooks on software engineering. As mentioned earlier, group-based estimates are often described as dangerous, and often, these descriptions are not accompanied by any deeper explanations as to when and how such dangers applies.

Two topics for further research are comparison of the unstructured approach studied in this paper with more structured group processes, such as the Delphi [22] or the Wideband Delphi [24] techniques, and comparison of unstructured estimating approaches with WBS and checklist-supported estimates.

Acknowledgements

We wish to thank Simula Research Laboratory for funding the experiment, Dag Sjøberg for valuable comments, and all the participants and organizers at the company participating in the study. Kjetil Moløkken-Østvold was funded by the Research Council of Norway under the project INCO.

References

1. Adams, S., *Dilbert and the way of the weasel*. First ed. 2002: HarperCollins Publishers Inc.
2. Gray, A., S. MacDonnell, and M. Shepperd. *Factors systematically associated with errors in subjective estimates of software development effort: the stability of expert*

judgment. *Sixth International Software Metrics Symposium*. 1999: IEEE Comput. Soc, Los Alamitos, CA, USA. pp. 216-227.

3. Boehm, B., et al., *Software cost estimation with Cocomo II*. 2000, New Jersey: Prentice-Hall.
4. Shepperd, M., C. Shofield, and B. Kitchenham. *Effort estimation using analogy*. *International Conference on Software Engineering*. 1996. Berlin, Germany: IEEE Comput. Soc. Press, Los Alamitos, CA, USA. pp. 170-178.
5. Reifer, D.J., *Web development: estimating quick-to-market software.* IEEE Software, 2000. **17**(6): pp. 57-64.
6. Jørgensen, M., *A Review of Studies on Expert Estimation of Software Development Effort.* Journal of Systems and Software, 2004. **70**(1-2): pp. 37-60.
7. Boehm, B.W., *Software engineering economics.* IEEE Transactions on Software Engineering, 1984. **10**(1): pp. 4-21.
8. Hughes, R.T., *Expert judgement as an estimating method.* Information and Software Technology, 1996. **38**(2): pp. 67-75.
9. Armstrong, J.S., *Principles of forecasting*. 2001, Boston: Kluwer Academic Publishers.
10. Moses, J. and J. Clifford. *Learning how to improve effort estimation in small software development companies*. *24th Annual International Computer Software and Applications Conference*. 2000. Taipei, Taiwan: IEEE Comput. Soc, Los Alamitos, CA, USA. pp. 522 - 527.
11. Heemstra, F.J. and R.J. Kusters, *Function point analysis: Evaluation of a software cost estimation model.* European Journal of Information Systems, 1991. **1**(4): pp. 223-237.
12. Paynter, J. *Project estimation using screenflow engineering*. *International Conference on Software Engineering: Education and Practice*. 1996. Dunedin, New Zealand: IEEE Comput. Soc. Press, Los Alamitos, CA, USA. pp. 150-159.
13. Hill, J., L.C. Thomas, and D.E. Allen, *Experts' estimates of task durations in software development projects.* International Journal of Project Management, 2000. **18**(1): pp. 13-21.
14. Jørgensen, M. *An empirical evaluation of the MkII FPA estimation model*. *Norwegian Informatics Conference*. 1997. Voss, Norway: Tapir, Oslo. pp. 7-18.

15. Hihn, J. and H. Habib-Agahi. *Cost estimation of software intensive projects: A survey of current practices*. *International Conference on Software Engineering*. 1991: IEEE Comput. Soc. Press, Los Alamitos, CA, USA. pp. 276-287.
16. Kitchenham, B., et al., *An Empirical Study of Maintenance and Development Estimation Accuracy.* Journal of systems and software, 2002. **64**: pp. 55-77.
17. Höst, M. and C. Wohlin. *An experimental study of individual subjective effort estimations and combinations of the estimates*. *International Conference on Software Engineering*. 1998. Kyoto, Japan: IEEE Comput. Soc, Los Alamitos, CA, USA. pp. 332-339.
18. Taff, L.M., J.W. Borcering, and W.R. Hudgins, *Estimeetings: Development estimates and a front end process for a large project.* IEEE Transactions on Software Engineering, 1991. **17**(8): pp. 839-849.
19. DeMarco, T., *Controlling software projects*. 1982, New York: Yourdon Press.
20. Jørgensen, M. *An Attempt to Model Software Development Effort Estimation Accuracy and Bias*. *Proceedings of Conference on Empirical Assessment in Software Engineering - 2003 (EASE 2003).* 2003. Keele, UK. pp. 117-128.
21. Lederer, A.L. and J. Prasad, *Information systems software cost estimating: a current assessment.* Journal of Information Technology, 1993. **8**(1): pp. 22-33.
22. Helmer, O., *Social Technology*. 1966, New York: Basic Books.
23. Fairley, D., *Making Accurate Estimates.* IEEE Software, 2002. **19**(6): pp. 61-63.
24. Boehm, B.W., *Software engineering economics*. 1981, New Jersey: Prentice-Hall.
25. Fenton, N.E., *Software Metrics*. 1995, London: Thompson Computer Press.
26. Humphrey, W.S., *Managing the Software Process*. 1990: Addison-Wesley Publishing Company, Inc.
27. Wiegers, K.E., *Stop Promising Miracles.* Software Development Magazine, 2000(February).
28. Kitchenham, B., et al., *An Empirical Study of Maintenance and Development Estimation Accuracy.* Journal of systems and software, 2002a. **64**: pp. 55-77.
29. Aronson, E., T.D. Wilson, and R.M. Akert, *Social Psychology*. 3rd. ed. 1999: Addison-Wesley Educational Publishers Inc.
30. Atkinson, R.L., et al., *Hilgard's Introduction to Psychology*. 12th ed. 1996, Orlando: Harcourt Brace College Publishers.
31. Hewstone, M., W. Stroebe, and G.M. Stephenson, *Introduction to social psychology*. 2nd ed. 1996, Oxford: Blackwell Publishers Ltd.

32. Zuber, J.A., H.W. Crott, and J. Werner, *Choice shift and group polarization: An analysis of the status of arguments and social decision schemes.* Journal of Personality and Social Psychology, 1992. **62**(1): pp. 50-61.
33. Bem, D.J., M.A. Wallach, and N. Kogan, *Group decision making under risk of aversive consequences.* Journal of Personality and Social Psychology, 1965. **1**(5): pp. 453-460.
34. Stoner, J.A.F., *A comparison of individual and group decisions involving risks*. 1961.
35. Wallach, M.A., N. Kogan, and D.J. Bem, *Diffusion of responsibility and level of risk taking in groups.* Journal of abnormal and social psychology, 1964. **68**(3): pp. 263-274.
36. Cromwell, P.F., et al., *Group effects on decision making by burglars.* Psychological reports, 1991. **69**: pp. 579-588.
37. Rowe, G. and G. Wright, *Expert opinions in forecasting: The role of the Delphi process*, in *Principles of forecasting: A handbook for researchers and practitioners*, J.S. Armstrong, Editor. 2001, Kluwer Academic Publishers: Boston. pp. 125-144.
38. Kernaghan, J.A. and R.A. Cooke, *Teamwork in planning innovative projects: Improving group performance by rational and interpersonal interventions in group process.* IEEE Transactions on Engineering Management, 1990. **37**(2): pp. 109-116.
39. Brown, R., *Group Processes*. 1988, Cambridge: Blackwell Publishers.
40. Forsyth, D.R., *Group Dynamics*. 1999: Wadsworth Publishing Company.
41. Fischer, G.W., *When oracles fail--a comparison of four procedures for aggregating subjective probability forecasts.* Organizational Behaviour and Human Performance, 1981. **28**(1): pp. 96-110.
42. Kitchenham, B., *Software Metrics: Measurement for Software Process Improvement*. 1996, Oxford: NCC Blackwell.
43. Kitchenham, B., et al., *Preliminary Guidelines for Empirical Research in Software Engineering.* IEEE Transactions on Software Engineering, 2002. **28**(8): pp. 721-734.
44. McDonald, A. and R. Welland. *Web Engineering in Practice*. *Proceedings of the Fourth WWW10 Workshop on Web Engineering*. 2001. pp. 21-30.
45. Thomsett, R., *Double Dummy Spit and other estimating games.* American Programmer, 1996. **9**(6): pp. 16-22.
46. Wonnacott, T.H. and R.J. Wonnacott, *Introductory statistics*. 5th ed. ed. 1990: John Wiley & Sons, Inc.

47. Liden, R.C., et al., *Management of poor performance: A comparison of manager, group member, and group disciplinary decisions.* Journal of Applied Psychology, 1999. **84**(6): pp. 835-850.
48. Cohen, J., *Statistical power analysis for the behavioral sciences*. 1969, New York: Academic Press, Inc.
49. Jørgensen, M. and K. Moløkken-Østvold. *Combination of Software Development Effort Prediction Intervals: Why, When and How? Fourteenth IEEE Conference on Software Engineering and Knowledge Engineering (SEKE'02).* 2002. Ischia, Italy. pp. 425-428.
50. Moløkken-Østvold, K. and M. Jørgensen. *A Review of Surveys on Software Effort Estimation*. *2003 ACM-IEEE International Symposium on Empirical Software Engineering (ISESE 2003)*. 2003. Frascati, Monte Porzio Catone (RM), ITALY: IEEE. pp. 220-230.
51. Winkler, R.L., *Combining forecasts: A philosophical basis and some current issues.* International Journal of Forecasting, 1989. **5**(4): pp. 605-609.

Appendix I:

User Requirements

This is the requirement specification for the project, as delivered by the customer.

The customer

The customer is the producer of an established technical encyclopaedia that numbers 800 editions with 10000 illustrations and 1600 tables. They have 20000 subscribers. The publication frequency is low, with two shipments each year, each containing several magazines.

The magazines exist both on paper and CD-ROM. The CD-ROM contains some extra features, and there are plans to add other sources of information.

There is also a simple intranet version of the CD-ROM that is used internally and by a handful of existing customers.

All documents are created in MS-word, and approved and converted to HTML by a central unit. They have no need for a very complicated CMS-system.

Status

The starting point for a web version is the existing CD-ROM and intranet based system. The primary goal is to build on the functionality of these systems, but also to offer the encyclopaedia commercially over the Internet, both to companies and individuals. It is natural to use this opportunity to look at the possibilities for a new medium, as well as revising existing production and administration routines.

All documents that will be used exist on the CD-ROM in HTML versions, and can be copied directly to the website. No changes are needed. Design is of less importance.

All the users of the site are expected to be technical competent people, with experience and knowledge of the CD-ROM and/or paper versions.

Desired functionality
This is the required functionality.

1. Basis functionality (searching for and displaying information)

Display documents in HTML-format (document including local table of contents (TOC) in magazine)

Navigation through an expanding TOC

Navigation through an index

Free search (Expansion of the existing version)

2. Downloading of documents and pictures

The system must allow the downloading of documents in other formats.

PDF versions of documents for better prints where such exist

Figures in high-resolution bitmap (TIFF)

Figures in vector format (DWG)

Displaying of video-clips

The system must handle the fact that not all documents and figures exist in all formats.

3. Extranet functionality

Only paying subscribers shall have access to the service. For companies this can be implemented by access-limitation on a net-level (IP). Company customers can then skip logon procedures with usernames and passwords. An alternative method is to use personal subscriptions. The system must be able to handle different types of subscriptions that grant different degrees of access to the system.

4. Trade solution

Possibility of subscribing via the web

Possibility of buying a single magazine for downloading or delivery by mail

Possibility of paying online by credit card or other forms of payment

5. Demo-/sales-version

The system shall have an open part, granting access to some functionality, such as navigation and search, and limited content. This functionality should be combined with a function that allows the purchase of single magazines for downloading.

6. Reply service

This contains an overview of the FAQ. The answers should have links to magazine editions with extensive information. The user is checked, and receives the option to log in, buy a single magazine, or buy a subscription.

7. User adjustment and information

Possibility of having personal settings for each user

Possibility for users to register own comments to the magazine

Possibility for the editors to publish comments to the magazines

Possibility for users to give feedback directly from a magazine, reporting errors etc.

Discussion forum tied to magazines

8. Administration of users

The administration system stores information on all customers, both subscribers and buyers of single issues. The administration system must monitor the use of the system. One must be able to extract different types of statistics, as well as blocking users who abuse the system or fail to pay their subscription.

9. Integration with existing administrative systems

Subscriber- and logistics system, Agresso

Customer and support, Superoffice

Degree of integration must be based on cost/value aspects.

10. Adjustment of the production system

The production system is based on a personally developed database, containing a parser that translates documents from MS Word format to HTML. The system must be adapted to a new medium and a new system. Other possible changes to consider:

Parsing of word documents to XML instead of HTML, which will add to the system's flexibility.

Expansion of the database to support the administration of manuscripts.

Adapt the base for the production of additional documents used in the printed issue (TOC, index list, overview of new, changed and expired magazines etc.)

11. Interface with other systems

By implementing the magazine in a web setting, integration with other systems should be considered. An interface (API) for communication with other systems (Using SOAP, XML etc.) should be defined. It must be possible to link to documents by URL.

12. Choice of technology

The goal is to use already known technology. This is to handle development and changes with internal resources.

OS: Windows 2000

Web server: Internet Information Server w/ASP

Database: Microsoft SQL Server

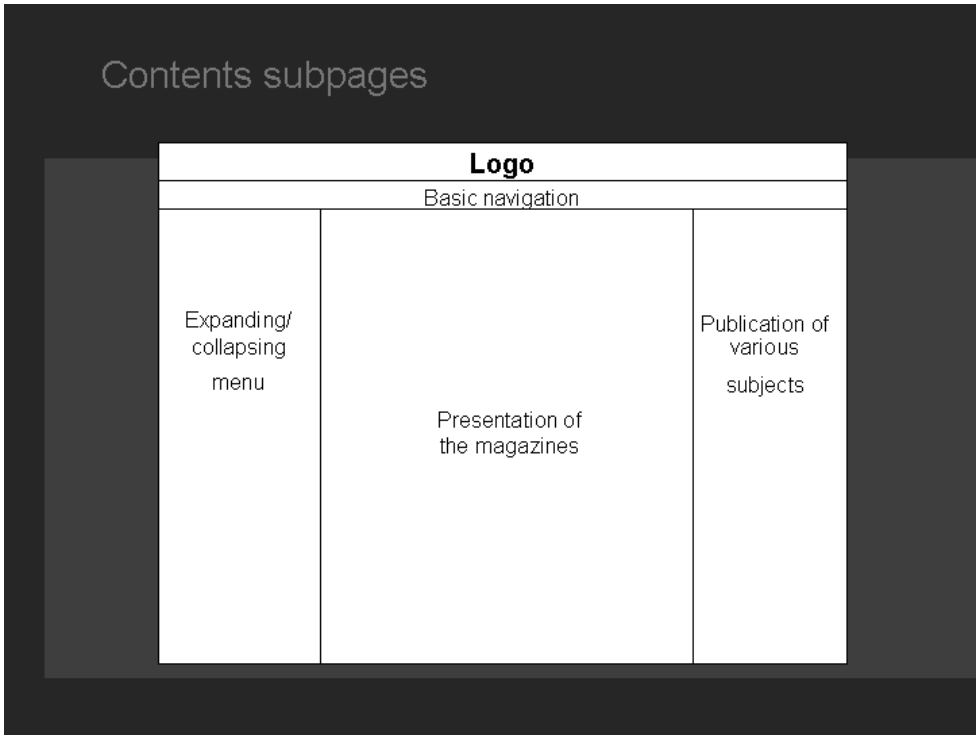
Languages: Visual Basic, NET technology

13. Hosting

The system is to be installed on the client's servers.

Figure 1: Front page Example

Figure 2: Sub page Example



Paper VI:

The Impact of Development Model on Estimation Accuracy in Software Projects

Kjetil Moløkken-Østvold and Magne Jørgensen.
Simula Research Laboratory.

Submitted to IEEE Transactions on Software Engineering.

Abstract. *Flexible software development models, e.g., evolutionary and incremental, have become increasingly popular. Advocates of these models claim that among the benefits is improved estimation accuracy, which is one of the main challenges of software project management. This paper describes an in-depth survey of software development projects. The results support the claim that estimation accuracy improves when a flexible development model is applied. The reason for the improvement is not obvious. We found, for example, no difference in project size, estimation process, or delivered proportion of planned functionality between projects applying different types of development model. However, we did find that the type of client had a strong impact on the estimation accuracy when applying flexible development models. This suggests that a better client relationship, which is facilitated by flexible development models, is an important reason for the observed improvement in estimation accuracy.*

Keywords: Cost estimation, management, project control and modelling, software development models.

1. Introduction

Software projects are infamous for exceeding their originally estimates [1]. A recent review of surveys on estimation performance reports that 60-80% of all software projects encounter effort overruns [2]. The average effort overrun appears to be 30-40%. Similar findings are reported for schedule overruns.

A large amount of research initiatives directed at improving estimation accuracy have focused on the development of formal estimation models. Formal estimation models have been based on a variety of measures of development size, such as lines of code [3] and function-points [4], and a variety of model development approaches, such as linear regression [5]. There is, however, no conclusive evidence to show that the employment of formal estimation models results in improved estimation accuracy [2, 6]. Consequently, expert estimation remains the preferred approach for most software professionals [2, 6].

Another line of research has focused on improving the expert estimation process by introducing a variety of supporting tools and processes, including work breakdown structures (WBS) [7, 8], checklists [9, 10], experience databases [11] and group-based estimates [10, 12-16]. These supporting tools may lead to improvements in some cases, but there is no evidence to suggest that they solve the problem of inaccurate estimates.

These observations are similar to results obtained in other areas of research. For example, in the transportation infrastructure sector, it has been reported that 86% of projects face cost overruns, and that the average magnitude of the overruns is 28%. These overruns appear to be of the same magnitude, independent of location and the era of the projects. Further, which estimation approach is used does not appear to affect estimation accuracy [17]. A consequence of these findings is that researchers have begun to explore factors other than the estimation process when trying to explain overruns.

Interestingly, in software engineering research, much less attention has been paid to ways in which improved project management may, through more appropriate development models, affect estimation accuracy. The use of incremental and evolutionary development models, for example, is said to facilitate more accurate estimates, when compared to sequential development [18-21].

The use of terms to describe development models may be confusing. It is, for example, frequently not easy to separate iterative, evolutionary, agile, and incremental development models. In this paper we apply the term ‘sequential’ to denote development models similar

to the waterfall model and the term 'flexible' to denote all types of non-sequential development models, e.g., iterative and incremental development models. The term 'flexible' to describe the broad class of non-sequential development models has previously been used by, for example, Iansiti and MacCormack [22] to reduce problems of classification and definition.

Advocates of flexible development models offer different explanations as to why such models should improve estimation accuracy. The estimation process associated with the incremental development model is, for example, claimed to lead to better managed projects [3, 19, 23, 24]. This is explained as an effect that results from the experiences of the first increment(s) [19]. If a software project in which an incremental development model is applied suffers from overruns, later increments can be eliminated or reduced, and the most useful parts of the system will still be produced within the original budget. There may, however, be management and control problems in connection with incremental development [19], and hence the overall impact of the use of the incremental development model is not obvious.

Improved project estimation has also been attributed to evolutionary development. Tom Gilb, for example, claims that "*Evolutionary delivery, and the feedback it produces, often leads to management acceptance of revised estimates for time and money at early stages of the project*" [25].

As stated in a comprehensive review of the history of iterative and incremental development, flexible development models have been in existence for some time [26]. In fact, they were first formulated as early as the 1930s. Still, such models were not adopted by practitioners, textbook authors and government bodies until recently [26]. The claimed benefits related to estimation accuracy have not been empirically investigated.

In order to assess the claims stated by proponents of flexible development models, we therefore investigated the use of development models in a survey on effort estimation in Norwegian software projects. The results are presented in this paper. Previous surveys on software estimation [2] have not addressed this topic. For some of the surveys, this is because they were conducted before the use of flexible development models became widespread [26].

The remainder of this paper is organized as follows. Section 2 examines earlier empirical studies on the relation between development model and estimation accuracy. Section 3 states our research questions. Section 4 provides a brief description of differences between the most common development models. Section 5 describes our research method. Section 6

reports the results of our survey. A discussion of the results is provided in Section 7. Section 8 concludes.

A preliminary version of this paper was presented at PROFES 2004 [27]. This version is substantially expanded, in terms of both the number of analyzed projects and depth of analysis.

2. Previous Studies

We were unable to find surveys or experiments that examined the impact of particular development models on effort estimation accuracy. The only studies examining this impact were case studies with results that are difficult to generalize. A selection of the case studies is briefly described in this section.

IBM Federal Systems report experiences with 45 deliveries during a four-year period applying evolutionary development models [28]. It is described that all 45 deliveries were on time and did not exceed the allocated budget, i.e., exceptionally good estimation accuracy results.

Using the evolutionary spiral development model for a PC-based ground control system developed at the Marshall Space Flight Centre, NASA also achieved good estimation accuracy results [29]. The researchers found that the evolutionary spiral development model forced risks to be identified, planned for, and resolved, while still providing the flexibility to meet volatile constraints. Before entering the first iteration of the spiral, the number of iterations that would be needed for completion of the project was not known. Hence, the team planned only the first iteration and did not engage in detailed, long-term spiral planning. Once the first iteration was complete and the second begun, an effort estimate was made for the additional iterations needed to address the current release requirements. The team gained knowledge and experience as the project moved through the spiral iterations, thereby increasing the quality and accuracy of planning and risk assessment.

Royce [30] describes experiences when using an incremental development model for large software systems for North American Aerospace Defence Command/Air Force Space Command. He reports that use of the incremental model increases the probability that a project will be completed according to schedule and budget.

Less positive estimation experiences related to the use of an evolutionary development model have also been reported [31]. Here, a high number of user-initiated changes caused one of the projects to experience large effort overruns, i.e., the use of twice as much effort

as estimated. The release was delayed by three months. However, the end users and the developers strongly believe that the delay, which was due to a large number of change requests from the user, was necessary for the company to deliver a product that met the real needs of the client. The project managers, however, regarded the delay as a disaster, and were very confused. They believed that they lost control of the project, because the requirements changed continuously while the formal documentation did not undergo the appropriate revisions. Therefore, the evolutionary model was abandoned and replaced by a formal waterfall model.

3. Research Questions

The study reported in this paper tries to contribute to more evidence-based discussion about the expected benefits of selecting a particular development model. This is done through a study of differences in effort and schedule estimation accuracy and bias (defined in Section 5.3). The research questions of the study were as follows:

RQ 1.1: *Are there differences in effort estimation accuracy that depend on the use of flexible or sequential development models*?

RQ 1.2: *Are there differences in effort estimation bias that depend on the use of flexible or sequential development models*?

RQ 2.1: *Are there differences in schedule estimation accuracy that depend on the use of flexible or sequential development models*?

RQ 2.2: *Are there differences in schedule estimation bias that depend on the use of flexible or sequential development models*?

4. Software Development Models

It is outside the scope of this paper to provide a complete description of all development models used in the software industry. This is especially true, given that the different models are combined in different ways, and often are tailored for specific purposes, projects or

companies. There exist several variants of the different sequential and flexible development models. An example of this is how Royce's description of what would become the waterfall model was (mis)interpreted and used in a fashion stricter than that which he originally suggested [26].

We will, for the support of the classification process described in Section 5, provide a brief description of the main properties of the most commonly employed development models used by the software companies observed in our study. The main development models observed were the following:

- Waterfall models
- Incremental models
- Evolutionary models
- Agile models

The traditional waterfall model is a sequential model [32]. It separates system development into distinct phases that are supposed to be completed in sequence, i.e., one phase should not be started before the preceding phase is completed. The phases are, typically, analysis, design, programming, testing and integration. Waterfall-based development models are widely used. Possible reasons for this are that a sequential development model is: 1) Easy to explain and recall, 2) Gives the impression of an orderly, accountable and measurable process, 3) Has been the standard development procedure in many communities, and 4) Has, until recently, been implemented by government regulations [26]. Opponents of such models, however, state that they only work well when technology, product features and competitive conditions are predictable or evolve slowly [22]. Such preconditions are, however, rarely present in software projects.

Incremental development is based on a division of the systems into parts (increments). The increments of the system are developed in sequence or in parallel [19]. Analysis, design, programming and testing are performed for each increment. Each of the increments provides a subset of the product's functionality. To minimize risks, the most critical part of the system may be delivered first.

The introduction of evolutionary project management in software engineering has been attributed to Tom Gilb [33]. In evolutionary development, the system is developed cyclically, with system deliveries in versions [25]. This is in contrast to the "big bang" delivery provided in the traditional waterfall development model. The delivered versions are adjusted according to client response and delivered again for a new assessment. Tom Gilb

states that "*You have the opportunity of receiving some feedback from the real world before throwing in all resources intended for a system, and you can correct possible design errors…*"[26]. Many software companies combine incremental and evolutionary development models.

Abrahamson, Salo *et al.* [34] define an agile development model as having the following properties: incremental (small software releases with rapid cycles), cooperative (client and developers working constantly together with close communication), straightforward (the model itself is easy to learn and to modify, and is well-documented), and adaptive (last minute changes can be made). Light and agile development addresses only the functions most needed by the client (the functions that deliver most business value to the client). These functions are then delivered as quickly as possible, and feedback collected, which will then be used to prompt further development.

As indicated in the description of the development models, a project may use more than one development model, e.g. a project may use a combination of incremental and evolutionary development. In addition, it may not be easy to decide whether a project follows an agile or evolutionary development model. These factors motivate our decision to separate the development models into two categories only: Sequential and Flexible. Sequential development models include waterfall-based development models, while flexible development models include all development models based on increments and evolution.

5. Survey Method

The survey was conducted between February and November 2003. In addition to studying the possible effects of the development model, the intent was to compare estimation practices and performance in the Norwegian software industry with findings from other countries.

5.1. The Participating Companies

In order to ensure a representative sample, stratified random sampling [35] from the population of Norwegian software development companies was used. This is a reasonable approach, since we were going to investigate a limited number of companies. It was necessary to ensure that we had companies that represented different types of organization, such as software houses developing products for the mass market, contractors who develop for clients and the internal development departments of large companies. We also wanted

companies of different sizes, both small (<25 employees), medium (between 25 and 100 employees) and large (>100 employees). The classification was based on different Norwegian sources, e.g. business magazines [36].

Each company was contacted by phone and the study was presented to them. If they agreed to participate, they were given time to prepare before they were visited by our researchers. The unit of investigation was either the entire company, or a specific department in cases where the company had more than 1000 employees. We will, however, use the term company for our unit of research in this paper. The eighteen companies (departments) that participated had between 10 and 750 employees, with an average of 141. Five of the companies developed projects to be used in-house, while two developed products for sale to the mass market. Out of the eleven companies that developed solutions for clients, nine had mainly private clients, while two had mainly public clients.

Each company submitted from one to four of their projects for scrutiny. The criteria that the projects needed to meet in order to be included in the study were that they should be over 100 hours (to exclude trivial tasks), be finished (either completed or abandoned), be the most recent cases (to avoid biased selection), and that we had access to the managers of the projects. This resulted in a repository of 52 projects.

5.2. Data Collection and Analysis

We collected data via personal semi-structured interviews, based on a predefined set of questions and interview instructions. Each interview lasted between 30 and 70 minutes, and all of them were taped. This approach yields data of high quality and ensures that ambiguities are resolved [35]. This was especially important in our survey, since there may be variations in the use and interpretation of terms related to development models and estimation approaches. It also allowed the respondents to add valuable information that did not fit into a predefined questionnaire. Another point in favour of this approach is that face-to-face interviews may increase the likelihood of serious contributions from the companies. The main limitation of the approach is that it is time-consuming, and hence prevents us from investigating as many companies and projects as would be possible by using questionnaires sent by post.

The interviews at company level were mainly concerned with background information on such general matters as number of employees, business segment, types of client, general aspects of development models, estimation approaches and process improvement efforts. At

project level, we collected detailed information about each project. This information included the type of project, type of client, estimation approach, development process and persons involved. Most important, however, was the collection of estimate(s) and actual(s) for effort and schedule. This was always based on recorded data from the participants, so that these results were not affected by hindsight bias.

The managers had the opportunity of defining their development model as being sequential (waterfall) or flexible (evolutionary, incremental and agile). They could also specify the extent to which component-based development and prototyping was used, and the extent to which combination models, e.g. evolutionary and incremental development, were used, and how the estimation had been conducted. In cases where the participants reported that they followed a company-defined development model or estimation approach, we asked them to provide descriptions of the process.

The terminology used in the context of software estimation is often ambiguous [13]. Different respondents may, for example, interpret estimated most likely effort differently. We were aware of such problems of interpretation and tried to ensure common interpretations of important terminology used in the interviews.

The written interview notes and tapes were registered into databases and the development models and estimation processes were classified by two independent researchers who had no stake in the research. This is especially important, because it ensures that there are no biases regarding how the development models or estimation approaches are classified. There was no indication that the researchers had disagreed on the classifications for any of the projects, thus indicating a high level of inter-rater reliability.

5.3. Measuring estimation accuracy

In order to assess estimation accuracy, both related to effort and schedule, the Balanced Relative Error (BRE) [37, 38] was used. It is calculated as:

$$BRE = \frac{|x - y|}{\min(x, y)}, \qquad \text{x = actual, y = estimated.} \qquad (1)$$

A more common measure of estimation accuracy is the MRE (Magnitude of Relative Error) measure [39]. MRE is calculated as:

$$MRE = \frac{|x - y|}{x}, \quad \text{x = actual, y = estimated.} \tag{2}$$

However, the MRE has unfortunate properties [37, 40, 41]. The main concern for our case is the fact that underestimated and overestimated projects are weighted unevenly. Assume, for example, that for two projects, A and B, the estimated effort is 1000 work-hours. Project A spends 500 work-hours, while project B spends 2000 work-hours. The MRE of Project A is 100%, while the MRE of Project B is only 50%. The BRE, as its name indicates, is a more balanced measure leading to a BRE of 100% for both projects.

BREbias measures both the size and direction of the estimation error, i.e., whether there is a bias towards an effort over- or under-run:

$$BREbias = \frac{(x - y)}{\min(x, y)}, \quad \text{x = actual, y = estimated.} \tag{3}$$

In order to investigate whether the development model influenced the completeness of delivered functionality, we interviewed the project managers about the delivered functionality on the finished products. They were asked about the extent to which the delivered functionality met the original specification on which the estimates were based. Since we did not have access to the opinions of the users of the products, there is a potential problem regarding the objectivity of the answers. In addition, there may be a hindsight bias. However, we have no reason to believe that such factors would affect one of the survey groups differently from the other.

5.4. The problem of multiple estimates

Previous surveys on software estimation have tended to treat a software estimate as a single fixed value. During the course of our research, however, we noticed that software projects often have several effort estimates. This problem, and how it poses challenges to estimation models, has also been addressed by Edwards and Moores [42].

Part of the problem is that the effort estimate often changes over the course of a project, depending on the *stage* at which the estimate is made. For example, a project might have an early estimate, based on vague requirements, a planning estimate based on a detailed requirement specification, and one (or more) re-estimates during the course of development.

Another factor contributing to the problem is the different receivers of the effort estimate. A project may, for example, have two estimates at the planning stage: one that is used internally in the project team and another communicated to the client.

Projects can consequently operate with one, two or even more different effort estimates. In our survey we found that a project could have as many as six different effort estimates. This poses a significant challenge to estimation surveys. In this paper, the goal is to investigate the estimation ability of professionals in software projects in relation to the development model used. For this reason, we found it meaningful to compare the *most likely* estimates at the *planning stage,* i.e. the estimate used internally by the developers at the stage where the decision to start implementation was made. This is in accordance with research on estimation accuracy in other areas [17].

5.5. Measuring Estimation Accuracy in Flexible Development Projects

There are a number of hazards related to comparing the estimation accuracy of sequential and flexible projects. In some guidelines for flexible models, it is stated that one should set a deadline for effort and schedule, and then simply produce whatever output is possible within those boundaries, with the most important functionality first. Obviously, with this approach, one will end up on target each time. It would be meaningless to do a comparison if we were just comparing the flexibility of flexible development models to the accuracy of sequential development models. However, in real life projects, especially when one develops for a client, stakeholders often expect some kind of predefined and agreed-upon functionality to be delivered. In all projects in this survey, regardless of development model, there existed schedule and effort estimates for the project total, along with the functionality expected when the decision to start the project was made.

Another problem that is related to the estimation accuracy of projects with a flexible development model is that many of them encourage estimation revisions throughout the project. An estimate given closer to the end of a project will probably be more accurate than one provided in the beginning. As discussed in the last subsection, this does not raise any problems for this survey, since we based the estimation accuracy calculations on the *most likely* estimates at the *planning* stage. This was done for both flexible and sequential projects.

6. Results

Out of the 52 project interviews, we excluded eight projects; either because they lacked information, or because most of the estimation and implementation work had been conducted by external sub-contractors. Two of the projects were abandoned before completion. This left 42 projects for the analysis. A complete record of the most relevant data is presented in Appendix II. Nineteen projects were classified as using a flexible model. The other 23 projects followed a sequential development model. Effort is measured in work-hours, and schedule in calendar days. For the projects, the mean actual effort was 3124.5 man-hours, while the median was 1175 man-hours. The average schedule was 177 calendar days, while the median was 131 calendar days. A short summary of the estimation results of projects with different development models is presented in Table 1.

		Mean		*Median*		*Standard deviation*	
		Flexible	*Sequential*	*Flexible*	*Sequential*	*Flexible*	*Sequential*
Effort	**Estimate (hours)**	2659	2373	1125	914	6035	2936
	Actual (hours)	3296	2983	1242	1150	6463	3679
	Accuracy (BRE)	0.36	0.59	0.14	0.60	0.54	0.44
	Bias (BREbias)	0.24	0.55	0.01	0.60	0.61	0.50
Schedule	**Estimate (days)**	145	152	122	117	72	132
	Actual (days)	164	187	122	140	83	19
	Accuracy (BRE)	0.14	0.35	0.06	0.14	0.20	0.66
	Bias (BREbias)	0.14	0.34	0.06	0.11	0.20	0.67
Functionality	**Delivered (%)**	106	106	100	100	13	12

Table 1: Estimation Results by Development Model

An Anderson-Darling test [43] on normality and a visual inspection revealed that none of the samples were normally distributed. We therefore applied the more robust non-parametric statistical Kruskal-Wallis [44] test of difference in median BRE and BREbias.

The median effort BRE was 0.14 for the flexible group, and 0.60 for the sequential group. The Kruskal-Wallis test on difference in median values resulted in a p-value of 0.017. To measure the magnitude of the observed accuracy difference in effort BRE, we included Cohen's size of effect measure (d) [45]. The size of effect (d) was calculated as: d = (mean value sequential group– mean value flexible group) / pooled standard deviation amongst the groups. The result was d=0.5, which is considered a medium effect [45].

Figure 1: Effort Estimation Bias of Sequential Projects.

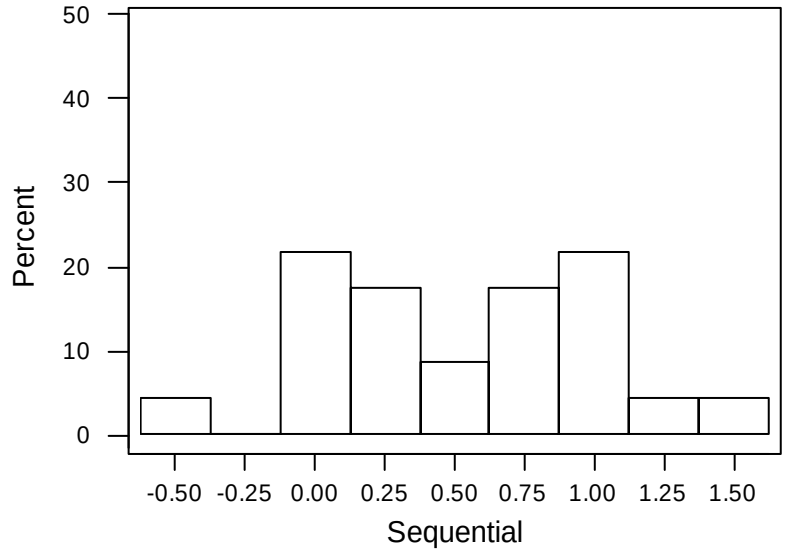
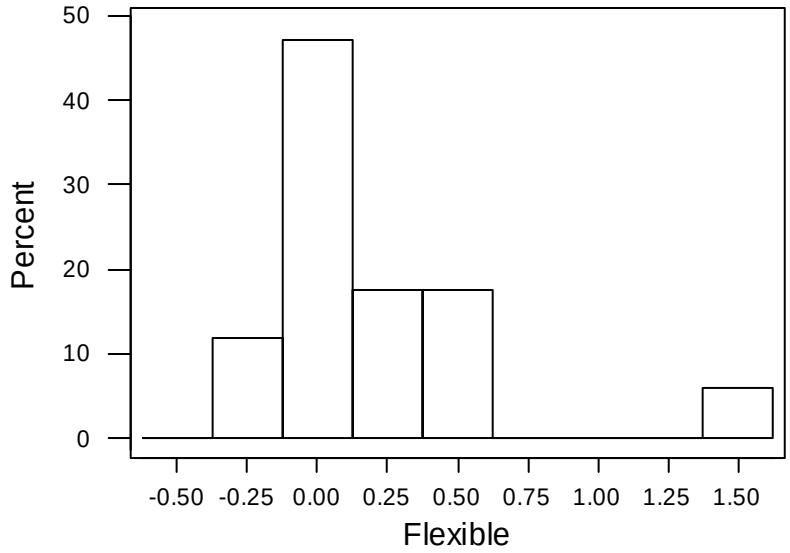


Figure 2: Effort Estimation Bias of Flexible Projects.



A graphical representation of the BREbias data is shown in Figures 1 and 2.

The median effort BREbias was 0.01 for the flexible group, and 0.60 for the sequential group. A Kruskal-Wallis test, performed on the median effort BREbias, resulted in a p-value of 0.07. As in the previous section, Cohen's size of effect measure was used. The result was d=0.6, which is considered a medium effect, since d is between 0.5 and 0.8 [45].

The overall mean effort BREbias was 0.41. A negative BREbias means that the estimate was too high, zero BREbias means that the estimate was on target, and a positive BREbias means that the estimate was too low.

The sequential projects are represented by the broad line, while the flexible projects are represented by the thin line. The overall mean effort BREbias of 0.41 is represented by the dotted line. A negative BREbias means that the estimate was too high, zero BREbias means that the estimate was on target, and a positive BREbias means that the estimate was too low.

The Kruskal-Wallis test on the median BRE of schedule estimation accuracy (flexible median BRE = 0.06 and sequential median BRE = 0.14) resulted in a p-value of 0.21. Cohen's d=0.4.

Since only one project finished ahead of schedule, the results for schedule BREbias are virtually identical (flexible median BRE = 0.06 and sequential median BRE = 0.11) to those for schedule BRE. The p-value from a Kruskal-Wallis test is 0.35. Cohen's d=0.4.

7. Discussion

Our results suggest that the choice of development model may affect effort estimation accuracy and bias. The results from the survey indicate that projects that are developed with a flexible development model may be less prone to effort overruns than projects that apply a sequential development model. The lack of significant difference in schedule estimation accuracy and bias is, we believe, a result of the low number of observations, i.e., low power of the study. We therefore need more observations to test the difference in schedule estimation accuracy.

The underlying reasons for the observed difference in estimation accuracy and bias are not obvious. The following sub-sections investigate possible explanations. We mainly address the explanations often mentioned as the advantages of flexible development, such as better estimation processes [20] and flexibility in delivered functionality [25]. In addition, we address alternative explanations, such as greater client involvement and smaller projects.

7.1. Better Estimation Process

The estimation process applied seems to be independent of the development model used. All the projects involved expert judgment-based, bottom-up estimation processes, and there were no differences among the groups related to the use of supporting estimation tools, such as checklists, experience databases and predefined work breakdown structures. Neither was there any difference in the amount of time spent on estimating the projects.

Proponents of flexible development models have specified several estimation revisions as the main reason for reduced overruns when these models are applied [20]. Surprisingly, we observed that the number of revisions of the estimates did not seem to increase for flexible projects. It is therefore unlikely that the observed difference in effort estimation accuracy and bias between the groups was caused by a systematic difference in estimation process.

7.2. More Flexibility in Delivered Functionality

We found no difference related to delivered functionality that depended on the development model used. The mean proportion of delivered functionality for both groups was 106% of estimated functionality. The managers were also asked to provide free-text responses to eventual overruns. None of the managers stated that the development of unnecessary functionality contributed to the overruns. This implies that, at least from the managers' point of view, differences in overruns between sequential and flexible projects are not due to differences in the amount of delivered functionality.

7.3. Smaller Projects

Project size may sometimes affect the effort overruns in software projects [46, 47], in that large projects seems to have larger effort overruns. If large projects more frequently follow a sequential development process this may explain the difference in estimation accuracy. In our survey, the 21 *largest* projects (measured in man-hours) had a BREbias of 0.52, while the 21 *smallest* projects had a BREbias of 0.30.

However, an analysis of the data suggests that this cannot explain the difference, since the choice of development model did not seem to be affected by the size of the project. Mean actual project effort was 3296 hours for projects that followed a flexible development model, as opposed to 2983 hours for the projects that followed a sequential development model.

7.4. More Client Involvement

An interesting observation in our study was that projects with public (government) clients had, on average, effort overruns three times larger than had projects with private clients, i.e., BREbias of 0.67 vs 0.21. However, there was no difference in the use of development models related to the type of client, i.e., differences in the type of client cannot explain the observed difference in estimation accuracy based on development model. A further analysis of the type of client data suggests that the client relationship, nevertheless, may explain the improved estimates from the use of flexible development models. Table 2 displays the mean effort estimation bias (BREbias) dependent on type of model and client.

	Flexible	*Sequential*	*Total*
Private	-0.02 (n=11)	0.40 (n=13)	0.21 (n=24)
Public	0.58 (n=8)	0.74 (n=10)	0.67 (n=18)
Total	0.24 (n=19)	0.55 (n=23)	*0.41 (n=42))*

Table 2: Mean BREbias based on client and development model type

Table 2 suggests that the benefit from flexible development models is much larger when the there is a private client. The private client projects differ by a magnitude of 0.42 based on choice of development model, while the public client projects differ by a magnitude of only 0.16.

It is believed, based on international and Norwegian studies, that public projects more frequently have confusing or contradictory goals (or, indeed, lack goals altogether), a diffusion of managerial responsibility, limited user involvement and are constrained by legislation [47, 48]. These problems may explain why public projects do not appear to receive the full benefits of flexible development models.

Consequently, it seems to be the improved client relations and communication, in cases where the client has the necessary maturity and skill that facilitates the improvements in estimation accuracy that are derived from the use of flexible development models.

7.5. Threats to Validity

We believe the most important threats to validity are the following:

- Small sample size
- Projects studied are not necessarily representative for other projects in other countries and in other situations
- Some of the data are subjective and subject to different interpretations, e.g., the proportion of delivered functionality
- No uniform terminology for description of development models

The small sample size was the result of a trade-off between quality of data and number of observations. We decided to focus on quality of data. The projects we studied may not be representative for other types of project. However, the accuracy results we report are similar to those reported by other surveys on effort estimation presented in a recent review [2]. The problems of subjective data and a lack of common interpretations of terminology are difficult to solve. We have tried to solve them through good interview processes and independent assessors.

8. Conclusion

We found that the use of a flexible development model, (e.g., evolutionary or incremental), improved the effort estimation accuracy and reduced the effort estimation bias. The reason for this improvement is not obvious and many of the common explanations, e.g., more flexible deliveries, different estimation processes and smaller projects, may not be valid.

It appears as if Harlan D. Mills was correct, when we as far back as 1976, he stated that *"The evolution of large systems in small stages, with user feedback and participation in goal refinements at each step is a way of going from grandiose to grand software system development"*. The most likely explanations supported by our observations are related to the relationship and communication with the client, i.e., to issues such as end-user involvement, client feedback loops, negotiations between developers and clients, and the decision processes in the client's organization. It may be that a flexible development model is better at facilitating a positive dialogue between client and developers than a sequential development model is, and that this is the most important difference.

We have started further work on understanding the findings reported in this paper. This work focuses on client-provider aspects of software projects, e.g., how involvement at an early stage and frequent communication between clients and developers, in combination with flexible development models, may lead to better control of software projects.

Acknowledgements

This research was funded by the Research Council of Norway under the project INCO. We thank Sinan Sigurd. Tanilkan, Hans Gallis, Anette Lien and Siw E. Hove for execution of the study, and Dag Sjøberg, Tom Gilb, Stein Grimstad and Erik Arisholm for valuable comments.

References

1. Yourdon, E., *Death March*. 1997, New Jersey: Prentice-Hall, Inc.
2. Moløkken-Østvold, K. and M. Jørgensen. *A Review of Surveys on Software Effort Estimation*. *2003 ACM-IEEE International Symposium on Empirical Software Engineering (ISESE 2003)*. 2003. Frascati, Monte Porzio Catone (RM), ITALY: IEEE. pp. 220-230.
3. Boehm, B., et al., *Software Estimation with COCOMO II*. 2000: Prentice-Hall.
4. Matson, J.E., B.E. Barrett, and J.M. Mellichamp, *Software development cost estimation using function points.* IEEE Transactions on Software Engineering, 1994. **20**(4): pp. 275-287.
5. Miyazaki, Y., et al., *Robust regression for developing software estimation models.* Journal of Systems and Software, 1994. **27**(1): pp. 3-16.
6. Jørgensen, M., *A Review of Studies on Expert Estimation of Software Development Effort.* Journal of Systems and Software, 2004. **70**(1-2): pp. 37-60.
7. Woodward, H., *Project Management Institute practice standard for work breakdown structures*. 2001, Newton Square: Project Management Institute, Inc.
8. Tausworthe, R.C., *The Work Breakdown Structure in Software Project Management.* Journal of Systems and Software, 1980. **1**: pp. 181-186.
9. Jørgensen, M. and K. Moløkken-Østvold. *A Preliminary Checklist for Software Cost Management*. *QSIC 2003*. 2003. pp. 134-140.

10. Shepperd, M. and U. Passing. *An Experiment on Software Project Size and Effort Estimation*. *2003 ACM-IEEE International Symposium on Empirical Software Engineering (ISESE 2003)*. 2003. Frascati - Monte Porzio Catone (RM), ITALY: IEEE. pp. 120-129.
11. Engelkamp, S., S. Hartkopf, and P. Brossler. *Project experience database: a report based on first practical experience*. *International Conference on Product Focused Software Process Improvement*. 2000. Oulu, Finland. pp. 204-215.
12. Linstone, H.A. and M. Turoff, *The Delphi Method: Techniques and Applications*. 1975, London: Addison-Wesley.
13. Grimstad, S., M. Jørgensen, and K. Moløkken-Østvold, *Software Effort Estimation Terminology: The Tower of Babel.* Submitted to Information and Software Technology, 2005.
14. Moløkken-Østvold, K. and M. Jørgensen. *Software Effort Estimation: Unstructured Group Discussion as a Method to Reduce Individual Biases*. *The 15th Annual Workshop of the Psychology of Programming Interest Group (PPIG 2003)*. 2003. Keele, UK. pp. 285-296.
15. Moløkken-Østvold, K. and M. Jørgensen, *Group Processes in Software Effort Estimation.* Empirical Software Engineering, 2004. **9**(4): pp. 315-334.
16. Taff, L.M., J.W. Borcering, and W.R. Hudgins, *Estimeetings: Development estimates and a front end process for a large project.* IEEE Transactions on Software Engineering, 1991. **17**(8): pp. 839-849.
17. Flyvbjerg, B., M.S. Holm, and S. Buhl, *Underestimating Costs in Public Works Projects - Error or Lie?* Journal of the American Planning Association, 2002. **68**(3): pp. 279-295.
18. May, E.L. and B.A. Zimmer, *The Evolutionary Development Model for Software.* Hewlett-Packard Journal, 1996. **47**(4): pp. 39-45.
19. Graham, D.R., *Incremental Development and Delivery for Large Software Systems*, in *Software Engineering for Large Software Systems*, B.A. Kitchenham, Editor. 1990, Elsevier.
20. Gilb, T., *Estimating Software Attributes: Some Unconventional Points of View.* ACM Sigsoft Software Engineering Notes, 1986. **11**(1): pp. 49-59.
21. Mills, H.D., *Software Development.* IEEE Transactions on Software Engineering, 1976. **2**(4): pp. 265-273.

22. Iansiti, M. and A. MacCormack, *Developing Products on Internet Time.* Harvard Business Review, 1997(Sept.): pp. 107-117.
23. Cockburn, A., *In Search of Methodology*, *Object Magazine*. 1994. pp. 52-56.
24. Cockburn, A., *Surviving Object-Oriented Projects*. 1998: Addison-Wesley.
25. Gilb, T., *Principles of Software Engineering Management*. 1988: Addison-Wesley Publishing Company.
26. Larman, C. and V.R. Basili, *Iterative and Incremental Development: A Brief History.* IEEE Computer, 2003(June): pp. 2-11.
27. Moløkken-Østvold, K., et al. *Does Use of Development Model Affect Estimation Accuracy and Bias? Product Focused Software Process Improvement, 5th International Conference, PROFES 2004*. 2004. Kansai Science City, Japan: Springer (LNCS 3009). pp. 17-29.
28. Mills, H.D., *The Management of Software Engineering, Part 1: Principles of Software Engineering.* IBM Systems Journal, 1980. **19**(4): pp. 414-420.
29. Hendrix, T.D. and M.P. Schnenider, *NASA's TReK Project: A Case Study in Using the Spiral Model of Software Development.* Communications of the ACM, 2002. **45**(4).
30. Royce, W. *TRW's Ada Process Model for Incremental Development of Large Software Systems*. *12th International Conference on Software Engineering (ICSE'12)*. 1990. Los Alamitos, CA: IEEE. pp. 2-11.
31. Lien, A.C. and E. Arisholm. *Evolutionary Development of Web-applications - Lessons learned*. *European Software Process Improvement Conference (EuroSPI'2001)*. 2001. Limerick Institute of Technology, Ireland.
32. Royce, W. *Managing the development of large software systems: Concepts and techniques. Proceedings of IEEE WESTCON*. 1970. Los Angeles. pp. 1-9.
33. Gilb, T., *Software Metrics*. 1976: Little, Brand and Co.
34. Abrahamson, P., et al., *Agile software development methods. Review and analysis.* 2002, VTT Publication. pp. 107.
35. Cozby, P.C., *Methods in behavioral research*. 5th ed. 1993, Mountain View: Mayfield Publishing Company.
36. HegnarOnline, *Kapital DATAs 1500 største*. 2000.
37. Jørgensen, M. and D. Sjøberg, *An effort prediction interval approach based on the empirical distribution of previous estimation accuracy.* Journal of Information and Software Technology, 2003. **45**(3): pp. 123-136.

38. Miyazaki, Y., et al., *Method to estimate parameter values in software prediction models.* Information and Software Technology, 1991. **33**(3): pp. 239-243.
39. Conte, S.D., H.E. Dunsmore, and V.Y. Shen, *Software Engineering Metrics and Models*. 1986, Menlo Park: Benjamin-Cummings.
40. Stensrud, E., et al. *An empirical validation of the relationship between the magnitude of relative error and project size*. *Eighth IEEE Symposium on Software Metrics.* 2002: IEEE. pp. 3-12.
41. Foss, T., et al., *A simulation study of the model evaluation criterion MMRE.* IEEE Transactions on Software Engineering, 2003. **29**(11): pp. 985-995.
42. Edwards, J.S. and T.T. Moores, *A Conflict Between the Use of Estimating and Planning Tools in the Management of Information Systems.* European Journal of Information Systems, 1994. **3**(2): pp. 139-147.
43. Christensen, R., *Analysis of variance, design and regression. Applied statistical methods*. 1998: Chapman & Hall/Crc.
44. Siegel, S. and N.J. Castellan, *Non-parametric Statistics for the Behavioral Sciences*. 2nd Edition ed. 1988: McGraw Hill College Div.
45. Cohen, J., *Statistical power analysis for the behavioral sciences*. 1969, New York: Academic Press, Inc.
46. Jørgensen, M. and K. Moløkken-Østvold, *Understanding Reasons for Errors in Software Effort Estimates.* Accepted for IEEE Transactions on Software Engineering., 2004.
47. Blaalid, J., *Country Report from Norway*, *OECD-PUMA expert meeting on management of large IT projects.* 2003, Statskonsult: Oslo.
48. West-Knights, L., Getting IT Right for Government - A Review of Public Sector IT Projects. 2000, Intellect.

Appendix I: Survey data

Nr	Model	Estimate (hrs)	Actual	BREbias	Estimate (days)	Actual	BREbias	Func.
1	Flex.	330.0	319.0	-0.03	235	235	0.00	100%
2	Seq.	560.0	1000.0	0.79	84	140	0.67	110%
3	Seq.	300.0	600.0	1.00	156	156	0.00	100%
4	Seq.	700.0	1400.0	1.00	182	224	0.23	100%
5	Seq.	4227.0	5170.0	0.22	98	98	0.00	110%
6	Seq.	1077.0	1150.0	0.07	42	49	0.17	110%
7	Seq.	4000.0	9000.0	1.25	293	335	0.14	110%
8	Flex.	1500.0	1512.0	0.01	70	70	0.00	100%
9	Flex.	1125.0	1000.0	-0.13	106	106	0.00	100%
10	Flex.	1249.0	1242.0	-0.01	153	214	0.40	98%
11	Seq.	1410.0	955.0	-0.48	74	64	-0.16	98%
12	Seq.	12000.0	14000.0	0.17	619	640	0.03	95%
13	Flex.	1249.0	1242.0	-0.01	92	106	0.15	100%
14	Seq.	487.5	562.5	0.15	103	106	0.03	110%
15	Seq.	640.0	1085.0	0.70	54	84	0.56	90%
16	Seq.	3937.5	4012.5	0.02	138	152	0.10	95%
17	Seq.	750.0	1200.0	0.60	117	457	2.91	100%
18	Flex.	533.5	466.5	-0.14	97	104	0.07	100%
19	Flex.	570.0	907.0	0.59	109	116	0.06	112%
20	Flex.	292.0	342.0	0.17	113	113	0.00	105%
21	Seq.	914.0	1903.0	1.08	196	217	0.11	120%
22	Seq.	400.0	432.0	0.08	42	56	0.33	110%
23	Flex.	705.0	1000.0	0.42	60	70	0.17	105%
24	Flex.	2265.0	2732.0	0.21	210	245	0.17	120%
25	Flex.	1932.0	5631.0	1.91	220	281	0.28	120%
26	Flex.	2340.0	3454.0	0.48	140	245	0.75	150%
27	Seq.	650.0	696.0	0.07	49	49	0.00	100%
28	Flex.	27241.0	28645.0	0.05	296	336	0.14	90%
29	Seq.	7520.0	8063.0	0.07	395	395	0.00	99%
30	Aborted	6728.0	n/a	n/a	151	n/a	n/a	n/a
31	Seq.	5450.0	8910.0	0.63	212	304	0.43	110%
32	Seq.	90.0	180.0	1.00	21	56	1.67	100%
33	Aborted	2720.0	n/a	n/a	152	n/a	n/a	n/a
34	Seq.	145.0	195.5	0.35	79	84	0.06	100%.

35	Flex.	190.0	101.0	-0.88	60	85	0.42	100%
36	Flex.	593.5	593.5	0.00	152	152	0.00	100%
37	Flex.	506.0	506.0	0.00	70	70	0.00	100%
38	Flex.	3784.0	3746.0	-0.01	266	266	0.00	100%
39	Flex.	1030.0	1335.0	0.30	122	122	0.00	115%
40	Seq.	2170.0	3831.0	0.77	54	54	0.00	100%
41	Flex.	3086.0	7844.0	1.54	183	183	0.00	100%
42	Seq.	1982.0	3140.0	0.58	153	153	0.00	145%
43	Seq.	133.5	261.0	0.96	273	334	0.22	100%
44	Seq.	340.0	866.5	1.55	72	91	0.26	125%

Paper VII:

Ethical Concerns when Increasing Realism in Controlled Experiments with Industrial Participants.

Kjetil Moløkken-Østvold.
Simula Research Laboratory.

Accepted for HICSS38 (Hawaii International Conference on System Sciences), 2005.

Abstract. *The emerging interest in realistic controlled experiments in computer science has created a need to examine related research ethics. Increased realism and scale in experimental studies pose new challenges that have not been debated to a sufficient extent. Specifically, there can be conflicts between the ethical principles of scientific value and informed consent. This paper provides an account of related previous work in computer science research ethics. To illustrate, two large-scale software engineering experiments with industrial participants are described. Challenges and solutions in these experiments are discussed in the light of current ethical guidelines. Interviews and debriefing sessions with industrial participants from these, and other, experiments are also provided. These reveal that there will not necessarily be ethical problems with increased realism, provided that the researchers respect the principles of informed consent, benefice and confidentiality.*

1. Introduction

In research areas such as medicine, law and psychology, the ethical aspects of scientific experiments are constantly scrutinized, debated and reported in research papers. It is also common to have independent review boards to assess the ethical aspects of an experiment before a grant, an approval or equivalent is given [1].

By contrast, in computer science, research ethics is seldom debated in general and almost never in published papers. In a recent survey on software engineering research from 1993 to 2002 [2], 110 research papers describing controlled experiments were noted. Only one of these directly discussed ethical aspects!

However, there have at least been some recent efforts to discuss research ethics in software engineering. The reason for this development is perhaps a combination of the fact that research areas in computer science are maturing, as well as an increase in focus on *empirical* studies. An example of this attention is the special issue of Empirical Software Engineering, published in 2001, which was entirely devoted to research ethics [3].

In the IS community, there has also been recent effort to address this problem, both related to ethics for professionals [4] and practitioners [5].

Although debated ethical controversies in computer science generally are of a lesser magnitude than in, e.g., psychology, there are several topics that disquiet researchers focusing on ethical aspects. A survey conducted among all 94 computer science departments at universities in the UK found that only 36% of the 44 respondents (department heads) felt that ethical considerations were important [6]. The survey also reports that only 26% of the departments felt that their universities are monitoring the ethical aspect of the research reasonably (24%) or very (2%) effectively. It has also been stated in a review [7] that guidelines and regulations applying to research areas in computer science have not applied to entities such as organizations serving as research subjects.

A common criticism of studies in software engineering is that most of them are case studies [8]. Since there are many ongoing efforts to make use of more realistic experiments, there has emerged a concomitant need for a debate on how this may cause ethical concerns. There is a huge difference between passive observation, and actual manipulation of participants, processes and projects. This paper seeks to explore the challenges that researchers face when trying to increase the realism of controlled experiments in the industry, especially when actual projects and professionals are used as research entities. To

illustrate, we mainly use examples from software engineering research, since this area is currently experiencing a surge in more realistic experimentation. However, we feel that the factors discussed here are relevant for all researchers who seek to increase the realism of industrial experiments in computer science.

This paper summarizes a recent survey on controlled experiments in software engineering and provides an account of how the researchers treat ethical issues. We also describe the ethical challenges faced in two recent experiments (DES and BESTweb) conducted at Simula Research Laboratory. The participants' responses to ethical issues are also presented. We also assess how the process of debriefing may affect the sentiments of research subjects.

In section 2, previous related research is presented, while section 3 presents two general research questions. Section 4 presents the results from a survey on controlled experiments. Section 5 presents two actual studies to use as a basis for the discussion in section 6. Section 7 presents a general discussion, while section 8 summarizes.

2. Previous Research

Much of the previous research in this area has been aimed at adopting and adapting guidelines from medicine, psychology, law etc., for use in computer science. There has been a focus on the participants as individuals, and how to treat them, whether they are students or professionals. Some papers in the IS [5] and SE communities [9-11] (the special issue of Empirical Software Engineering [3]) have this focus, which is very important. There have also been contributions that discuss certain elements of the experimental process that one may face, such as the logging of user actions [12].

However, when one tries to increase realism in an experiment, the ultimate goal may sometimes be to have a setting similar to that which is found in the industry. This includes having realistic projects. Therefore, it is necessary to consider the participants as part of their company, and also to address the challenges of having a company as a research entity. Efforts to address ethical issues at a company level have, until now, focused on factors other than using employees or project teams as participants in a controlled experiment.

On a more general level, it has been stated that existing codes in research ethics are relevant, but not directly applicable to studies on empirical software engineering [7]. Such guidelines are often related to authorship, relations to students, and fraud, but not to the use

of professionals or companies as participants. Also, strict guidelines are said to impose too large a bureaucratic burden [5].

In order to discuss the ethical considerations described in the following sections, this paper will use the review of principles related to empirical software engineering as described by Singer and Vinson [7]. These four high-level principles are: informed consent, scientific value, benefice and confidentiality.

Informed consent is perhaps the primary ethical principle related to research on empirical software engineering [7]. Informed consent is made up of several elements, and there is an ongoing debate as to the extent to which they are necessary. The elements are as follows:

- Disclosure – How the participants are informed about research purpose, procedure, risks, benefits and a statement offering to answer the participants' questions.
- Comprehension - This requires the researchers to present the research in a way the subjects can understand.
- Competence – This element refers to the subject's ability to make rational decisions.
- Voluntariness – This indicates that informed consent must be obtained without pressure on the participant.
- Termination – The participants must be able to terminate their involvement at any time.

Scientific value is composed of the elements:

- Importance of research topic – This is related to potential risks and benefits to subjects and society.
- Validity of results – Unless the results of the study are valid, they do not represent reality, and are therefore unethical.

Benefice relates to a favourable balance between benefits and harm. The greater the possible benefits, the greater the risk allowed.

Confidentiality normally has two elements:

- Anonymity – The data can not be used to identify the participants.
- Confidentiality – This limits who will have access to the data, and how they will be described.

In addition to the general ethical principles presented by Singer and Vinson [7], it is also important to follow local standards. We investigated how the principles presented by Singer and Vinson [7] match up to research guidelines for social sciences, law and humanities in

Norway [13]. These local guidelines [13] also contain aspects that are omitted by Singer and Vinson, and therefore also acts as a supplement for researchers in Norway.

3. Research questions and method

This paper addresses two main issues. The first is the amount of attention that ethical issues have received in software engineering research. On the surface, it can seem as if the software engineering community has devoted little attention to research ethics. This leads us to the research question:

RQ1: How do research papers in experimental software engineering report and discuss ethical aspects?

In order to investigate this topic, we have conducted a thorough survey of papers in experimental software engineering [2]. How these papers treat research ethics is analyzed and reported in section 4.

The second issue is how an increase in experimental realism may pose new ethical challenges. The goal of more realistic experiments, through use of "real" companies and professionals, is ultimately a means to achieve the higher end (and ethical principle) of scientific value. However, such goals may not necessarily be in line with other ethical principles. From other areas, such as psychology, we know that informed consent is a principle that is frequently overlooked in the search for scientific value [1]. The most famous, and debated, example is Stanley Milgrams studies on conformity [14]. Informed consent in Psychological research includes that *participants must enter a study voluntarily and be permitted to withdraw from it at any time without penalty if they so desire.* No matter how "innocent" computer science studies may look, when compared to psychological experiments, we still face special challenges. As reported in the previous section, researchers in software engineering who have investigated ethical aspects, also regard *informed consent* as perhaps the most important ethical principle [7]. This leads us to:

RQ2: To what extent does increased realism in controlled experiments with industrial participants create conflicts between the ethical principles of "informed consent" and "scientific value"?

In order to explore this, we report in section 5 from the two latest experiments performed by Simula Research Laboratory. We believe that these are examples of experiments that try to achieve a high degree of realism in order to possess scientific value. These experiments are scrutinized with respect to ethical issues, with a focus on informed consent. This is presented in section 6. In order to augment the self-assessment, we provide responses from post-experiment interviews and debriefing sessions with the participants.

4. Survey Results

As part of a large scale survey to analyze properties of controlled experiments in software engineering, researchers at Simula Research Laboratory investigated 5453 different research papers [2].

The survey characterised the controlled software engineering experiments published in a sample of nine journals and three conference proceedings in the decade from 1993 to 2002: ACM Transactions on Software Engineering Methodology (TOSEM), Empirical Software Engineering (EMSE), IEEE Computer, IEEE Software, IEEE Transactions on Software Engineering (TSE), Information and Software Technology (IST), Journal of Systems and Software (JSS), Software Maintenance and Evolution (SME), Software: Practice and Experience (SP&E), and the proceedings of International Conference on Software Engineering (ICSE), IEEE International Symposium on Empirical Software Engineering (ISESE), and IEEE International Symposium on Software Metrics (METRICS).

Out of the 5453 research papers, 119 different controlled experiments were identified, described in 110 different papers. This is an interesting result in itself, but the lack of controlled experiments in software engineering lies beyond the scope of this paper, and is discussed elsewhere [2].

The subset of papers (n=110) identified as describing experiments in software engineering was then analyzed with respects to the research question (RQ1) presented in section 3. Only one out of 110 papers addresses ethical issues directly [15]. That paper relegates its comments on the topic to a footnote, and provides an account of debriefing and post experiment interviews with the subjects of the experiment. None of the papers discuss important ethical topics such as “informed consent and “scientific value”. However, 17 of the papers describe debriefing procedures, mainly through post-experiment questionnaires. Several of the experiments are interesting from an ethical point of view, but it is futile to

further analyze or rank them, since few details about ethical and other important experimental aspects are described in the papers.

The fact that ethical issues are not discussed does not necessarily mean that they were not regarded as part of the experimental process. It may just be the case that there is no tradition for reporting such aspects in software engineering research papers. In order to obtain more information, we can use related information, such as recruitment of subjects. This is frequently reported, and can provide us with an indication of ethical aspects. Such information can also provide us with valuable inputs when analyzing the experiments conducted by Simula Research Laboratory.

Most of the subjects reported in the papers were students, which is understandable out of practical and economic reasons. Nonetheless, a troubling observation was that for one third of them, participation was mandatory! Such mandatory participation has also been criticized in the IS community [5]

Equally concerning is the fact that participation directly affected their grades in nine cases. This figure might actually be higher than has been stated, since a description of the reward was only present for about 35% of the experiments. A complete overview is presented in table 1.

Reward	*N*	*%*
Unknown	78	65.5
Part of job	15	12.6
Grade	10	8.4
Extra credits	9	7.6
Paid	3	2.5
Unpaid	3	2.5
Other rewards	1	0.8
Total	119	100.0

Table 1: Incentives for Participants.

Out of the analyzed experiments, there were no reports of using paid professionals. In experiments with professionals, participation was either part of an ordinary project or training programme. It is therefore difficult to compare these experiments with the ones described in this paper.

5. Description of Studies

This paper uses as examples two controlled experiments conducted by Simula Research Laboratory that used industry professionals as subjects. One is a bidding and development study that was recently conducted with participants from 35 different companies, and the other is an implementation study with professional programmers. The reasons for selecting these experiments for analysis in this paper are twofold. The main reason is that these experiments aim for an increased degree of realism, when compared to previous experiments. In addition, information (e.g. selection procedures, incentives, confidentiality and debriefing interviews) about these experiments is described much more thoroughly than in most other experiments. A brief overview of the characteristics of the two experiments, as compared to most other experiments in software engineering [2], is presented in Table 2.

Experiment	***Regular SE experiment***	***DES***	***BESTweb***
Participants	Students	Practitioners	Practitioners
Incentives	Varies	Paid	Paid
Realistic tasks	Sometimes	Yes	Yes
Full project	No	Yes	No
Real work environment	Seldom	Yes	Yes
Product to be used after completion	No	Yes	Yes

Table 2: Experiment properties

It should be noted that the author of this paper is responsible for the BESTweb, but has no stakes in the DES study.

5.1. Team Bidding and Development Study (DES)

This study was conducted in 2003. We were planning on getting a web-based system developed by a professional contractor to register and systemize data from experiments we conduct. We also wanted to perform some studies as part of the project, mainly divided between the bidding phase and the development phase.

In the bidding phase, our purpose was to investigate whether, and how, the evolution of requirement specifications in the (fixed-price) bidding process would affect the bids of software contractors. The underlying hypothesis was that if a contractor first receives a

limited requirement specification, they establish a bid that acts as an anchor. If they later on receive a larger extended specification, this anchor will affect the magnitude of the final bid. During this phase, 35 different software companies were approached and invited to deliver a bid. We invited a range of small, medium and large companies, randomly selected from the Norwegian industry. None of the companies had previous relations to the client (Simula Research Laboratory) that could affect the bids. The software companies were randomly divided into two groups, with 17 companies in group A and 18 companies in group B. Those in group A received a small initial specification, and were invited to provide a non-binding bid and solution sketch on this basis. Each company was paid 5000 NOK (approximately 600 Euro) for this work. The companies in Group B did not participate in this initial bidding round.

Based on responses from the companies and internal needs, Simula then developed a larger and more complete requirement specification. This requirement specification included many features that were more complicated than the original specification, and would require more development effort from the companies.

In the next bidding round, the final requirements were sent to all companies, grouping both Groups A and B. This was our first contact with software companies in Group B. All companies then had to deliver fixed-price bids based on this specification. Again, each company was paid 5000 NOK (approximately 600 Euro) for this work. The companies received instructions similar to those issued in the first bidding round, with the exception that this was based on final requirements. The bidding companies were given the information that we would select *at least* one of them as provider, and perhaps more than one.

After the bidding rounds were completed, four providers were selected to develop the solution. This was the development phase of the study, in which different teams implemented the same system based on the same final requirement specification. The selection was based on relevant business criteria, based on our needs.

Those who did not win one of the contracts were contacted by Simula Research Laboratory and debriefed. The only element of control in the development phase was that we selected two companies that had specified the use of UML and two companies that had not. This was done in order to use the applications in further studies.

The four companies who won the contracts used development teams located at their own organizations to develop their projects. In the development phase of DES, the projects required the companies to follow a number of firm guidelines. These were government

standards (Statens Standard), which often are used in projects procured by public institutions. Once a week, the teams were interviewed by a researcher from Simula Research Laboratory who had no stake in the project. These teams were debriefed after project completion.

5.2. Individual Development Study (BESTweb)

The BESTweb study was conducted to investigate how the estimation performance of professional programmers is affected by the time available in the estimation phase of a project. The programmers had to estimate the required workload of several tasks, measured in hours. They were subsequently asked to perform the programming tasks. The purpose was to investigate whether more time spent on effort estimation will lead to less optimistic (and probably more realistic) effort estimates.

This study was also based on an actual project, required by our software engineering department. The experiment was based on the extension of an already existing system for organizing literature references called BESTweb. It is of smaller magnitude than DES, but nonetheless larger than similar studies, which have, in the main, used small student tasks. The BESTweb study has more comparable previous experiments than the DES study, which may allow us to evaluate our procedures with those of other researchers. There are, however, certain aspects that differentiate this study from most similar controlled experiments. Principally, BESTweb used paid professionals as subjects, which is very unusual [2]. The study was recently concluded, and is currently under analysis.

The experiment recruited ten professional programmers from different software companies. They were hired on an hourly basis. The participants were encouraged to behave as in a normal assignment. They could take breaks when they felt like it, and they could use any tools available. The participants were required to make several extensions to the system, divided into different tasks. The tasks required about one work-week to complete. The goal was to make the tasks as realistic as possible. This is an actual system, which is in use at Simula, and the extensions implemented were based on real requirements as requested by the researchers who use the system.

Even though we tried to make the experiment as close to a real software project as possible, there were some factors that we controlled in the experiment. The participants were given a limited amount of time to estimate the tasks before they started programming. This amount of time was varied between the tasks and participants, and how this factor influences

estimation performance is the main research interest of the experiment. The experiment also required the tasks to be developed sequentially, and a new task could not be started before the existing task was solved according to predefined requirements. All participants received the same tasks in the same order.

Prior to the start of the study, the participants were informed that they would be working on an actual project, but that we would like to use the study as an opportunity to study software estimation. They were informed that we wanted to elicit how experts estimate. They were, however, encouraged to perform as in a real project, since the project was a realistic assignment, and we were bona fide clients.

They were also informed that a logging tool was installed. The purpose of the logging tool was to monitor how participants interact with the programming environment. This data may be used by researchers who wish to extend the study. They were also informed that no personal data was stored, and that no information would be sent from us to the employers of the participants or other parties. All researchers signed a confidentiality agreement. After the end of the assignment, they were debriefed and interviewed with respect to ethical issues.

6. Ethical Considerations and Debriefing Results

This paper will not go through every element of the stated guidelines or principles [7, 13] and see how our, or other researchers, experiments adhere to these. Instead, we will focus on how ethical principles may be affected when experimental realism is increased.

This section presents how the ethical notions of *informed consent, benefice and confidentiality* may be affected when one tries to attain *scientific value.* We also provide examples of how we have managed to achieve results of scientific value while satisfying the demands of informed consent, benefice and confidentiality in our most recent experiments. The main emphasis will be on the different elements of informed consent (disclosure, comprehension, competence, voluntariness and termination). Our experiences with debriefing sessions are also discussed.

6.1. Informed Consent: Disclosure

Disclosure is always problematic in controlled experiments [1]. Often, it is essential that the participants do not know or try to guess the hypothesis. When dealing with software companies as research entities, there are often two levels of disclosure: one to the

employers, and one to the participants. The amount of disclosure required to be given to the participants is different for each experiment, and must be balanced by the researchers.

In the bidding phase of the DES study, we informed the companies and participants that it was an experiment, but that this was just one small part of an otherwise normal project. They were told that *at least* one provider would be selected to implement the solution, and that the selection would be based on business evaluations. This was also the case. In the development phase of the study, they were also encouraged to work as if on a normal project. They were informed that the selection of application(s) to be implemented would be based on normal business evaluations. The only aspects that differed from a normal project were that it was four similar projects, and that the teams were interviewed each week (further elaborated upon in subsection 6.6).

In the BESTweb study, the participants were informed about the purpose of the experiment, including most development aspects. The only thing that remained hidden was how and why we controlled the amount of time they received in the estimation phase.
The developers in BESTweb were monitored closely. The developers used workstations provided by Simula Research Laboratory. These workstations were fitted with advanced logging tools [12], which allowed a high degree of monitoring of the participants in order to perform a detailed analysis of their development method. The professionals are probably used to having their code scrutinized, but not at this granularity. However, the participants were informed about the monitoring, and they had the opportunity to decline, or turn the tool off.

6.2. Informed Consent: Voluntariness and Termination

As described in the review in section 4, the use of paid software professionals in controlled experiments is very rare. Often, when using software professionals, it is sensible to recruit them through their employers. This may make it difficult for the subjects to abstain from participation or to withdraw from the experiment if they feel uncomfortable. The element of voluntariness and the opportunity for participants to terminate the experiment at any time are two key elements of informed consent [7]. Since hired consultants normally are required to generate billable hours for their companies, the threshold for declining may be increased compared to experiments that recruit participants as individuals, e.g. through advertisement.

In order to secure voluntariness and the ability to terminate for our participants, we always encourage a close dialogue. All our researchers have signed confidentiality agreements (elaborated in subsection 6.3). These agreements state that no sensitive information from experiments or other studies will be handed to third parties, such as employers.

In the second phase of the DES study, there was a possibility that we would encounter *individuals* who would like to withdraw from the experiment. However, as participants in a team, they might have felt pressure from peers or management to stay on the project, even though it might have been uncomfortable for them. To cope with this eventuality, there were weekly anonymous interviews with all participants. These were conducted by a researcher from Simula Research Laboratory who had no stake in the project. None of the participants indicated any desire to leave the project.

Since only individuals were used in the BESTweb study, possible problems related to voluntariness and terminations were of a lesser magnitude than in the DES study. The participants were informed that they could withdraw at any time, for any reason. If they chose to do so, we would only inform their employers that the participant did not have the right profile for the study, and pay compensation for the effort spent.

6.3. Informed Consent: Comprehension and Competence

Comprehension and competence are often linked. When one recruits software professionals for an assignment, one must first assess the level of complexity of that assignment. Normally, this can be done by independent experts. Recruitment through companies can then be based on the skill levels required for that task. These requirements must be explained in a plain manner. This is also the case for all other tasks performed and instructions issued during the experiment.

In the DES experiment, we sent requirement specifications to the companies, which then had the responsibility of selecting competent people. The requirements were presented in a clear and readily comprehensible manner.

For the BESTweb study, we sent a list of the technologies involved to the companies, and a description of the level of complexity. It was then their responsibility to select participants. In order to make sure that the participants had the right kind of knowledge, a pre-experiment interview was conducted by one of our researchers.

6.4. Benefice

It is important to ensure that any possible harm done to the companies does not outweigh the benefits. Since benefits are often difficult to measure, one should strive for situations that will not generate any harmful effects.

For the bidding phase of the DES study, it would be pertinent to make comparisons with other actual software projects. Unfortunately, due to the high grade of realism in the study, there are no meaningful research comparisons to make. While in our study only four out of the 35 companies that delivered bids got the assignment, the common industry practice is that only one company wins the contract. In addition, those who delivered bids received 5000/10000 NOK as a participation fee. This is *not* the case in the industry when you fail to win a contract.

In the BESTweb study, it was a possibility that the control elements of the experiment might render the performance of the developers sub-optimal. The developers were required to complete the tasks in a given sequence, and were not allowed to plan ahead. This could require a larger effort measured in man-hours to complete the project. The developers might have been hampered by these restrictions, resulting in a failure to perform at an optimum level. Nonetheless, they were paid by the hour, so a longer development cycle only meant more billable hours. Most important, the tasks were not linked, which indicates that the order of development should not affect performance.

6.5. Confidentiality

Protecting the identity of participants is crucial in all kinds of studies. It is also important to ensure that no sensitive information is leaked to other parties.

As mentioned earlier, all researchers at Simula Research Laboratory have signed confidentiality agreements. More specific agreements can also be provided to companies and participants on request, as was done in a recent survey [16].

Both in DES and BESTweb, all information provided to Simula about sensitive issues was kept confidential. An example is the fine granularity logs from the logging tool in BESTweb. Those data were collected from the computer automatically, and analyzed by a researcher who had no means of backtracking and linking the data to any specific participant.

6.6. Debriefing Experiences

Debriefing was prominent both in DES and BESTweb. In the first phase of the DES study, after selecting four providers out of the 35, we debriefed all companies. Each company was contacted via phone by a representative of Simula. There were no indications that the companies regretted participation in the study. Nonetheless, a few of the companies were disappointed because they were not selected as a provider. This is, however, not uncommon in a bidding situation such as this.

In the second phase of the study, debriefing and follow-up of the participants was more thorough. During the whole development period, each project team was interviewed once a week. All project members had a personal interview session with an independent researcher from Simula, who was not a stakeholder in any of the study phases. Each semi-structured interview lasted between 15 and 75 minutes, and a total of 98 interview sessions were conducted. The interviewer did not receive any indication of problems related to informed consent or other ethical issues. No persons or teams wanted to pull out of the experiment, and for all practical purposes, the study was treated as a regular project. None of the participants raised ethical objections, and none of them felt pressured to participate by their management. No participants felt, at any stage, that they were being misled or lied to. They were satisfied with the information about the study received from Simula. In fact, two of the participants were a little "disappointed" by the relatively low amount of research and manipulation. They had actually been looking forward to a project that departed from the norm.

The participants in the BESTweb study were also debriefed after all development assignments were performed. During the debriefing session, any criticism could be stated anonymously to a neutral researcher. These data are currently under analysis, but no problems related to ethics have been uncovered so far.

7. Discussion

As previously described by other authors [3], and seen in our survey presented in section 4, there has not been much focus on research ethics in the software engineering literature. Whether this is due to lack of awareness or just poor reporting, we do not know. It is most likely a combination. Previous papers [6] have revealed a low degree of attention to research ethics, and a recent survey has found much of the reporting related to controlled

experiments in software engineering to be inadequate [2]. The reporting problem is a general issue that relates not only to ethical issues, but to all aspects of experiments [2]. In short, it may well be that the answer to our first research question is that reporting and discussion of ethical concerns in the software engineering literature is virtually nonexistent.

Lack of reporting also makes it difficult for us to answer our second research question. This is principally due to the lack of similar cases with which to compare our own. Research studies normally recruit participants as individuals, e.g. through advertising or by using students. This also applies to many studies in computer science. However, when one seeks to increase realism in such studies, it is necessary to recruit real companies, project teams or professionals. Often, the only sensible way to achieve this is by recruitment through the companies themselves. Since participation (and payment) is then often linked to the company, withdrawal may not be so easy to handle, both for participants and researchers.

The DES study has several properties that make it interesting as a starting point for discussion. Mainly, it has tried to increase, to a significant degree, the level of realism in software engineering research. Previously, the greatest difference regarding experiments in empirical software engineering was that between using students and professionals in controlled experiments. These experiments were typically small tasks, lasting from a few hours to a number of days. The DES experiment has done has increased the level realism by departing from small arbitrary tasks, performed by individuals, and instead studying large scale realistic industry projects, developed by professional companies. DES is a study in which different companies "compete" against each other in an experimental setting, but with a realistic project, and such very realistic incentives as contracts, money and prestige.

In the BESTweb study, we used individual software professionals in a controlled experiment. Use of professionals in experiments has been debated to some extent, but not in our setting, where several professional developers from different companies develop the same project.

From the examples, it seems as though informed consent is the most prominent ethical concern, in studies at both individual and company levels. However, with an open procedure between the researchers and the participants, it is possible to keep issues raised by informed consent at an acceptable level. Such procedures include statements, confidentiality agreements, explanations (including debriefing), descriptions and the possibility to refuse participation [7, 10]. The main problematic issue related to informed consent in studies similar to the ones presented is, perhaps, that of voluntariness; not particularly between the researchers and the participants, but between the participants and their employers. If the

participants are paid professionals, it may be that the companies pressure them to participate in order to get billable hours from research institutions. This may be difficult for the researchers to discover, and such possibilities should be monitored closely.

Even though both benefice and informed consent were, taken individually, only partially problematic in the DES study, it is the combination of those elements that may be alarming. Many small problems may add up in a large study, and every part needs to be analyzed. It is difficult to approach this area, since research on companies has scarcely been debated, and even the more general guidelines [13] give vague statements: *"Researchers should respect the legitimate grounds private businesses, professional and industrial bodies and the like may have for not wanting information about themselves, their members or their plans published".*

Another possible problem is the pride of the developers, and the effort put into the product. Even though the *benefice* of the participants is preserved, since they get paid, they may also have other motives for development. If a participant's product is not chosen for further use, it is therefore important for the researchers to give a full account of the reasons behind such choices.

The presence of logging tools may also be unfamiliar to developers, and could cause both stress and reduced performance. However, a recent study [12] showed that as long as there is informed consent regarding the logging tools, the participants do not find them problematic. The study had 13 participants in programming tasks, similar to those in our BESTweb studies. The participants were surveyed about their feeling towards the experiment afterwards. On a seven-point Likert scale (1 – fully agree, 7 – fully disagree) the participants were asked if they worked differently because they knew that everything was being logged. The responses did not indicate that the participants felt that they had been affected, with a mean response of 6.1. The participants also expressed similar positive attitudes towards others parts of the experiment [12]. Since a normal software project can often be chaotic and stressful [17] it is extremely important that this stress is not augmented by the researchers.

There is little empirical data on the research ethics in computer science with which to compare our described debriefing sessions. There is, however, much information on debriefing from other research areas. One of the most discussed experiments in psychology, both because of its results and ethical controversy, is Stanley Milgrams experiment on obedience [14]. The experiment tricked participants into believing that they were administering potentially lethal electric shocks to what they thought was a learner in an

experiment on memory. In reality, the whole experiment was a cleverly devised illusion, and no shocks were delivered. However, Milgram showed that ordinary people were capable of delivering lethal shocks to other human beings if encouraged by an experimenter. The experiment was criticized for both its lack of informed consent and its possible harmful effects on the participants. Differing opinions regarding the experiment are described in a host of sources, e.g. textbooks in psychology and behavioural research [1, 18]. Milgram was very thorough in his debriefing efforts and surveyed his participants systematically. Despite the possible severe consequences of the experiment, a full 84% of the subjects were glad they participated, and 74% said they had benefited from participation. Only 1% regretted participation. The subjects were also interviewed by a psychiatrist one year after participation, and no ill effects were found.

No suffering appears to have been endured by the participants in our DES study. Even though we did not deem it necessary to contact a psychiatrist, the participants in both phases were debriefed and the teams in the second phase were interviewed regularly. No ethical concerns were voiced by the participants. It is possible that the participants lied to the interviewer about sensitive issues, such as withdrawal or pressure from management. However, our previous experiences with interviews from other studies, such as a recent survey [16], indicate that interview subjects are often quite frank about criticizing their management or company. Such a situation, when the interviewer has signed a confidentiality agreement, offers a rare opportunity for the interview subjects to vent possible anger and frustration in front of a neutral third person.

There are vast differences between Milgrams' studies on obedience, and seemingly harmless software engineering experiments. However, it is important to note that for the professionals who act as subjects, any misconduct on the part of researchers could be damaging for their professional careers. Thus, it is essential that care is taken to ensure that problems due to ethical issues do not arise.

It seems, in summary, as though the participants in the DES study were satisfied with the type and amount of information received. It did not appear as though the increased realism breached any aspects of informed consent. Our experiences from the BESTweb study are currently too limited to draw any conclusions.

It might be that the principle of *scientific value* will, in more realistic experiments, challenge the principle of *informed consent*. However, such problems can be kept to a minimum with an open dialogue and the extensive use of debriefing sessions.

8. Summary

It is not meaningful, or possible, to draw conclusions in this kind of paper. However, we feel confident that there is currently a lack of emphasis on ethical issues when increasing realism in controlled experiments. We have tried to illustrate this with examples from our own studies. It is important to note that this paper only describes possible problematic areas of *two* non-randomly selected studies, which were carried out at the author's institution.
However, the purpose of this paper is mainly to bring to attention issues that researchers face when trying to increase realism. Another purpose was to revitalize the debate on an important, but neglected, topic in our field. It is obvious that increased realism in experimental studies calls for extended and specialized ethical guidelines. It is therefore important for the research community to increase the focus on ethical aspects. This applies both to the planning, conducting and, especially, the reporting of experiments. There is almost no mention of ethical issues in many of the contemporary papers on empirical software engineering, and no common *de facto* guidelines are used.

Even though our experience comes principally from software engineering studies, we believe that the aspects discussed in relation to increased realism in controlled experiments are applicable to all researchers in the IT/IS area. It is hoped that such initiatives as initiated in the special issue of Empirical Software Engineering [3] will continue, and there may, perhaps, be an increased focus on research ethics in papers in conferences and journals, so that it may be possible to establish guidelines such as those used in other disciplines.

Acknowledgements

This research was funded by the Research Council of Norway under the project INCO. Thanks to all members of the SE group at Simula Research Laboratory and Chris Wright for valuable comments.

References

1. Cozby, P.C., *Methods in behavioral research*. 5th ed. 1993, Mountain View: Mayfield Publishing Company.
2. Sjøberg, D.I.K., et al., *A Survey of Controlled Experiments in Software Engineering.* Submitted to IEEE Transactions on Software Engineering., 2004.

3. Singer, J. and N. Vinson, eds. *Empirical Software Engineering*. ed. V.C. Basili and W. Harrison. Vol. 6. 2001, Kluwer Academical Publishers.
4. Davison, R., *Professional Ethics in Information Systems: A Personal Perspective.* Communications of the AIS, 2000. **4**.
5. Davison, R., et al., *Research Ethics in Information Systems: Would a Code of Practice Help?* Communications of the AIS, 2001. **7**.
6. Hall, T. and V. Flynn, *Ethical Issues in Software Engineering Research: A Survey of Current Practice.* Empirical Software Engineering, 2001. **6**(4): pp. 305-317.
7. Singer, J. and N. Vinson, *Ethical Issues in Empirical Studies of Software Engineering.* IEEE Transactions on Software Engineering, 2002. **28**(12): pp. 1171-1180.
8. Sjøberg, D.I.K., et al., *Challenges and Recomendations when Increasing the Realism of Controlled Software Engineering Experiments*, in *ESERNET Method Book*. 2002, LNCS.
9. Storey, M.-A.D., B. Phillips, and M. Maczewski, *Is it Ethical to Evaluate Web-based Learning Tools using Students?* Empirical Software Engineering, 2001. **6**(4): pp. 343-348.
10. Sieber, J., *Protecting Research Subjects, Employees and Researchers: Implications for Software Engineering.* Empirical Software Engineering, 2001. **6**(4): pp. 329-341.
11. Davis, M., *When is a Volunteer Not a Volunteer?* Empirical Software Engineering, 2001. **6**(4): pp. 349-352.
12. Karahasanovic, A. *Is it Ethical to Log Users' Actions in Software Engineering Experiments? Informing Science and Information Technology Education Joint Conference.* 2003. Pori, Finland. pp. 1211-1214.
13. Kalleberg, R., *Guidelines for research ethics in the social sciences, law and the humanities*. 2001, The National Committee for Research Ethics in the Social Sciences and the Humanities.
14. Milgram, S., *Behavioral Study of Obedience.* Journal of Abnormal and Social Psychology, 1963. **67**: pp. 371-378.
15. Jørgensen, M. and D.I.K. Sjøberg, *Impact of effort estimates on software project work.* Information and Software Technology, 2001. **43**: pp. 939-948.
16. Moløkken-Østvold, K., et al. *A Survey on Effort Estimation in Norwegian Software Industry*. *10th International Symposium on Software Metrics*. 2004. Chicago, Illinois, USA: IEEE Computer Society. pp. 208-219.

17. Yourdon, E., *Death March*. 1997, New Jersey: Prentice-Hall, Inc.
18. Atkinson, R.L., et al., *Hilgard's Introduction to Psychology*. 12th ed. 1996, Orlando: Harcourt Brace College Publishers.

Paper VIII:

How Large Are Software Cost Overruns? Critical Comments on the Standish Group's CHAOS Reports

Magne Jørgensen and Kjetil Moløkken-Østvold.
Simula Research Laboratory.

Submitted to Information and Software Technology.

---

1. Introduction

The Standish Group (www.standishgroup.com) claims that the results of their CHAOS research, i.e., their large-scaled surveys conducted in 1994, 1996, 1998, 2000 and 2002, are the most widely quoted statistics in the IT industry. This may very well be true. Quoted with particular frequency are the results described in the 1994 CHAOS report [1], probably because a summary of the 1994 CHAOS report is free and is the only one that can be downloaded from the web. The results of that report have been used in several recent governmental reports, project reviews, and research studies. Examples are the PITAC 1999 report [2] and the cost estimation study described in [3]. An important result from the 1994 CHAOS research is the reported 189% average cost overrun of so-called challenged projects, i.e., projects not on time, not on cost, and not with all specified functionality. In this paper we argue that the 189% average cost overrun number, as it is commonly interpreted, is not consistent with results of other cost accuracy surveys and probably far too high to reflect the average cost overrun in that period.

2. The Importance of the Correctness of 189% Average Cost Overrun Result

Among the users of the cost overrun numbers in the 1994 CHAOS report are scientific researchers, software process improvement groups, and government advisors. The main use seems to be to argue for more research, better estimation processes and improved project management methods. These are all laudable goals, well supported by a 'software cost estimation crisis' implied by a 189% average cost overrun. Unfortunately, there are several examples of situations where the 189% result may have hindered progress. The following three real-world examples illustrate this.

Example 1: A project had a 146% cost overrun, i.e., the actual cost was about 2.5 times the estimated cost. A report on the project's performance stated that the cost overrun was not that bad, because it was better than the industry average of 189%. There are several examples of this type of use.

Example 2: A consultancy company claimed to be in the elite class of software development companies, based on a comparison of its own numbers with the 189% cost overrun number.

Example 3: A recent UK study of software projects [4] report an average cost overrun of 18%. Here an adjusted version of the 189% cost overrun number from the 1994 CHAOS report is used to support an argument for an enormous improvement in estimation performance. Readers of that report may get the impression that the improvement of cost estimation processes does not require a great deal of systematic work and focus.

3. What Does 189% Average Cost Overrun Mean?

Before we compare the CHAOS Report results with those of other studies it is important to clarify the meaning of '189% average cost overrun'. This turned out to be more difficult than expected. In fact, we were unable to find an explicit definition of the cost overrun measure applied in the CHAOS reports. Only informal, inconsistent descriptions were presented. The following quotations provide typical examples of how the CHAOS reports describe the 189% average cost overrun:

- "*… 52.7% of projects will overrun their initial cost estimates by 189%*", page 41 in [5].
    - **Comment**: 52.7% is identical to the percentage of so-called challenged projects. Even the definition of challenged projects is not easy to interpret. It is defined in [1] as "*The project is completed and operational but over-budget, over the time estimate, and offers fewer features and functions than originally specified.*" The problem here is the use of "and" instead of "or", combined with the following definition of successful projects: "*The project is completed on-time and on-budget, with all features and functions as initially specified.*" Consider a project that is on-time, and on-budget, but not with all specified functionality. Is this project to be categorized as challenged or successful? Our guess is that it would be categorized as challenged, but this is not consistent with the provided definition of challenged projects.

- "*The average cost overruns for combined challenged and cancelled projects is 189%.*", page 42 in [5].
    - **Comment**: Here, the cancelled projects are included, i.e., there are two inconsistent descriptions of cost overrun in the same document. The method by which the cost overrun of a cancelled project might be included is not given.
- "*… 52.7% of projects will cost 189% of their original estimates*", page 14 in [6].
    - **Comment**: As we interpret it, if the cost is 189% of an estimate there is an 89% cost overrun, i.e., this is not the same as the first description.

To see whether we were the only ones confused by these descriptions we conducted a simple survey of the actual use of the 1994 CHAOS report cost overrun number. We examined 50 randomly sampled web-documents applying the search term: *((Standish Group) AND (189% OR 89%))* using the search engine www.yahoo.com. We found the following:

- 50% of the documents described the result as "*189% cost overrun*", 40% as "*189% of original estimate*", and 10% as "*89% cost overrun*". "189% of original estimate" and "89% cost overrun" seem to reflect the same understanding of the result, i.e., we found two different interpretations of cost overrun that were used with almost the same frequency.
- 70% of the documents related the result to "*53% of the projects*" (without explicitly pointing out that this 53% referred to challenged projects only), 16% to "*all projects*", 8% to "*challenged and cancelled projects*", and 6% explicitly pointed out that the average cost overrun is based on "*challenged projects*" only.

Generally, in recent reports and press releases the Standish Group seams to apply the interpretation "*189% average cost overrun of challenged projects*", see for example the press release March 2003 [7]. This means that many, perhaps the majority, of the users interpret the results differently from the Standish Group.

4. A Comparison with Other Cost Estimation Accuracy Studies

All surveys of cost estimation accuracy in the relevant period and countries that we were able to find suggest average cost overrun in the range of about 30% (see Table 1), i.e.,

values far from 189% cost overrun. The surveys in Table 1 have all been subject to scientific review of research method and results, as opposed to the Standish Group CHAOS reports. For a more detailed description of these and other cost estimation surveys, see [8].

Study	*Jenkins [9]*	*Phan [10]*	*Bergeron [11]*
Year	1984	1988	1992
Respondents	23 software organizations	191 software projects	89 software projects
Country of Respondents	USA	USA	Canada
Average Cost Overrun	34%	33%	33%

Table 1: Cost Overrun Surveys

These values are not directly comparable with those in the CHAOS reports. The studies in Table 1 include successful as well as challenged projects, as opposed to the CHAOS report where the successful projects are excluded from the cost overrun calculation. However, the proportion of successful projects in the 1994 CHAOS report was only 16% and cannot explain the huge difference in the results. The question is therefore: Are there other differences between the three studies in Table 1 and the CHAOS 1994 survey that can explain the huge difference in average cost overruns? Are there, for example, reasons to believe that the cost accuracy performance was so much worse in 1994 than in the period 1984-1992, or that the three studies in Table 1 are bias towards too low cost overruns? We can find no such reasons explaining the difference in results.

Interestingly, the Standish Group's CHAOS surveys for the years 1996, 1998, 2000, and 2002 report strongly decreasing numbers, i.e., 142%, 69%, 45%, and 43% average cost overruns. Adjusted for differences in how cost overrun is measured, we find that the numbers for 2000 and 2002 corresponds well with the average cost overrun of about 30% in the studies in Table 1, i.e., it seems as if it is mainly the early (1994, 1996 and 1998) CHAOS report cost overrun numbers that are unusual. The strong decrease in average cost overrun, as measured by the Standish Group, is a reason to doubt the research method in itself. For example, do we believe that the average cost overrun improved from 142% to 69% in only two years?

5. Why Are the Cost Overruns in the 1994 CHAOS Report So High?

To investigate reasons for the high 1994 cost overrun number, we asked the Standish Group how they selected the projects to be included in their CHAOS studies and how we should interpret 'cost overrun'. The response to our research method question was that providing this type of information would be like giving away their business for free, and we got no response on how to interpret 'cost accuracy'. This unwillingness to reveal research method and measurement definitions would have been an unacceptable response in an academic context, but is, as far as we have experienced, not uncommon in commercial companies conducting research studies.

This lack of research method and measurement definition information leaves us with no choice but to speculate about potential reasons. We have identified the following potential reasons explaining the '189% cost overrun' reported in the 1994 CHAOS research report:

- *Incorrect interpretation of own results*: Re-calculations of the average cost overrun based on the Standish Groups 1994 distribution of cost overrun per overrun category results in a cost overrun close to 89%, i.e., there may be an inconsistency between the two presentations (average cost overrun and cost overrun distribution) of the overrun data. When the Standish Group present the 1994-study as 189% instead of 89% cost overrun, it may have been misled by its own confusing description of the results. However, even a reduction from 189% to 89% does not lead to results on the level of the comparable studies.
- *No category for cost under-run:* In most studies on software cost estimation accuracy there is a proportion of projects with cost under-runs. For example, in the recent UK study on project performance [4] as many as 15% of the projects were completed *ahead* of budget. Even challenged projects may have cost under-runs, since they may be challenged only regarding time or functionality. We find no reference to, or description of, treatments of cost under-run in the Standish Group's reports. It is therefore possible that cost under-runs are not included as cost under-runs, but perhaps as 0% cost overrun.
- *Unusual definition of cost overrun:* The Standish Group may have used an unusual definition of cost overrun, e.g., the definition may include cost on cancelled projects as indicated in one of the three informal descriptions of cost overrun. A possible cost

overrun definition that includes cancelled projects (possibly consistent with an 89% cost overrun) is for example: *Cost overrun = TIM / (TIM – WM)*, where *TIM* = Total investment on software projects, and *WM* = Wasted money on cancelled projects and on cost overruns. Then, cost overrun is interpreted as how much more a company should expect to pay more than initially estimated when considering the possibility of both cost overruns and cancellation of projects. An example: If a company initially estimates an expenditure of $1.000.000 on IT-application development, the expected cost is $1.890.000 (189% of $1.000.000) to complete these projects. The strength of this measure is that it includes money spent on cancelled projects, which arguably is connected to cost overruns. The weakness is that no-one else uses this cost overrun measure, the measure is difficult to interpret, and it conflicts with our intuitive understanding of project cost overrun. In any case, this discussion demonstrates the necessity for a clear definition of "cost overrun" in the Standish Group's research reports.

- *Non-random sampling of projects*. Unusual results are sometimes caused by non-random samples. A thorough reading of the version 3.0 of the CHAOS report [6] provides some support for this explanation. On page 13 the selection process of the 1994-study is described as follows: *"We then called and mailed a number of confidential surveys to a random sample of top IT executives, asking them to share failure stories* [!!!]*. During September and October of that year, we collected the majority of the 365 surveys we needed to publish the CHAOS research.*" The decreasing average cost overrun numbers of more recent CHAOS research may therefore be a consequence of an increasingly more representative (and less failure-story related) selection of projects.

6. What Should We Learn From This?

This paper does *not* prove that the 189% average cost accuracy reported in the 1994 CHAOS report is biased and unrepresentative for the situation in 1994. Such a proof would require the availability of information that the Standish Group will not release. It is possible that the results are valid, and merely very difficult to interpret, given the lack of measurement definitions and research study description. Bearing in mind the above cautionary note as to what we are able to establish, we have attempted to provide reasons to *doubt* the 189% average cost overrun value as it is interpreted by most of its users. In

particular we believe that the unusually high average cost accuracy number and the lack of research method description are valid reasons for doubting that number. As software cost estimation researchers, we (the authors of this paper) and many others have uncritically applied the 1994 CHAOS Report cost overrun numbers to several studies, e.g., in [3, 12]. We therefore believe that there are lessons to be learned:

Lesson 1: When something does not correspond with own experience and other studies, doubt it. A 189% *average* cost overrun, as reported by the CHAOS research, is an extremely high number in relation to numbers reported in other studies. Consequently, we should require a detailed description of research method applied, or independent studies that replicate the result before believing it.

Lesson 2: The number of observations, which is higher in the CHAOS report than in the comparable studies, is not always a good indicator of the validity of the results. We should be just as concerned about the selection process as with the number of observations. If the selection process is not properly described, we should doubt the results regardless of the number of observations. Bias is not removed with an increased number of observations.

Lesson 3: Studies that do not precisely define their measures, as is the case with the CHAOS research, should be interpreted carefully. The confusion about the interpretation of the 189% average cost overrun illustrates this. For example, many documents referring to the CHAOS report did not notice that the average cost accuracy only referred to challenged (and maybe cancelled) projects, not to all projects.

7. Summary

The Standish Group reported in 1994 that the average cost overrun of software projects was as high as 189%. This cost overrun number is still of great importance. It is, for example, used as input in recent governmental reports, as benchmark for the estimation performance of recent projects, and to support the claim that there has been an immense improvement in cost estimation performance the last 10 years. We found several reasons to believe that an average cost overrun of 189% is much too high to reflect the situation in 1994. The number is not consistent with cost overrun results of other surveys in that period, and, there may be serious problems with how the Standish Group conducted their research.

Unfortunately, the Standish Group provides an incomplete description of how they conducted their studies, e.g., how they selected the projects to be included in the study, and does not include a description of the measure 'cost overrun'. This makes it difficult to evaluate the validity of their 1994 study. Even worse, the lack of precise definition of 'cost overrun' seems to have created much confusion. Many, perhaps the majority, of the users interpret the cost overrun results differently from the Standish Group. Our main conclusion is that we should doubt the validity of the 189% average cost overrun reported by the Standish Group in 1994 until such time as the Standish Group disclose how they measure cost overrun and how they conduct their research. Currently, the validity and comprehensibility of that number is highly questionable and may create the impression (a) that the IT-industry has improved strongly since 1994 and (b) that even very inaccurate projects are "better than average".

References

1. *The CHAOS Report.* www.standishgroup.com/sample_research, 1994.
2. *The PITAC REport.* www.hpcc.gov/pitac/report, 1999.
3. Jørgensen, M. and D.I.K. Sjøberg, *The impact of customer expectation on software development effort estimates.* International Journal of Project Management, 2004. **22**(4): pp. 317-325.
4. *The state of IT project management in the UK 2002-2003.* www.computerweeklyms.com/pmsurveyresults/surveyresults.pdf, 2003.
5. Johnson, J., *CHAOS: The dollar drain of IT project failures.* Application Development Trends, 1995(January): pp. 41-47.
6. *Chaos Chronicles Version 3.0*. 2003, The Standish Group: West Yarmouth, MA.
7. *Press release, The Standish Group.* http://www.standishgroup.com/press/article.php?id=2, March 2003.
8. Moløkken, K. and M. Jørgensen. *A Review of Surveys on Software Effort Estimation*. *IEEE International Symposium on Empirical Software Engineering (ISESE 2003)*. 2003. Rome, Italy. pp. 223-230.
9. Jenkins, A.M., J.D. Naumann, and J.C. Wetherbe, *Empirical investigation of systems development practices and results.* Information and Management, 1984. **7**(2): pp. 73-82.

10. Phan, D., D. Vogel, and J. Nunamaker, *The search for perfect project management*, *Computerworld*. 1988. pp. 95-100.
11. Bergeron, F. and J.Y. St-Arnaud, *Estimation of information systems development efforts: A pilot study.* Information & Management, 1992. **22**: pp. 239-254.
12. Jørgensen, M. and D.I.K. Sjøberg, *Impact of experience on maintenance skills.* Journal of Software Maintenance and Evolution: Research and practice, 2002. **14**(2): pp. 123-146.

Biographies:

Magne Jørgensen is a professor at University of Oslo and Simula Research Laboratory. His research interests include software cost estimation, uncertainty assessments in software projects, expert judgment processes, and, learning from experience. He received the Dr. Scient. degree in informatics from the University of Oslo, Norway in 1994. He has 10 years industry experience as consultant and software project manager. His publications on software cost estimation and related topics are listed at: http://www.simula.no/people_one.php?people_id=36. *magne.jorgensen@simula.no*.

Kjetil Moløkken-Østvold is a Ph.D. student at University of Oslo. His research interests include software cost estimation, expert estimation in groups and surveys of software estimation performance. He received the Cand. Scient. degree from University of Oslo in 2002. kjetilmo@simula.no.