Empirical Research on
Software Effort Estimation
Accuracy

Master thesis
60 credit

Kristian Marius Furulund

May 2nd 2007
Summary

To improve the software development process is named by both the European Union and the United States government as an important task for society. The constant problem with effort overruns and estimation inaccuracy is a main part of the software development problem. Empirical research on software effort estimation is a key part of the continuing effort by researchers and practitioners to improve the way in which software development projects are carried out.

As part of this effort, a study on eighteen of the latest projects at a Norwegian software consultancy was done. The study was done by interviewing the project managers responsible for the projects, having them provide key project data, and their assessments of different project properties related to effort estimation. The study focused on answering research questions related to:
- The effect the contractor-customer relationship and customer properties have on estimation accuracy
- The effect utilizing experience data has on estimation accuracy
- The role of estimation accuracy when assessing software project success

The analysis of the collected empirical data showed that reduced effort overruns was associated with increased contact frequency with the customer and contracts that share the risk between contractor and customer.

Utilization of experience data, and the use of checklists, was also found to have a positive impact on estimation accuracy.

There was not found any strong correlation between project managers’ project success assessment and estimation accuracy, indicating that estimation accuracy and project manager success assessment contribute with two different, but important viewpoints when software project success is to be assessed.

In addition to the empirical study and its results, the thesis presents a review of existing group combination techniques for software effort estimation. The review was motivated by recent studies that have suggested that to do software estimation as a group is beneficial. The review presents techniques that vary largely as to how they structure the interaction among the group members, and how their opinions are aggregated. A thorough discussion on the argumentation behind the techniques, and the consequences they have is given in the review.

The empirical data collected during the work with this thesis suggests different ways in which software contractors could improve their estimation ability and reduce their effort overruns.

The conclusions of this thesis is, that to increase estimation accuracy, software contractors should: (i) involve the customer, and nurture the customer relationship, (ii) add some repeatable structure to the estimation process, but be careful not to add too much structure, (iii) gather and utilize experience data in the estimation process and (iv) evaluate projects when they are done. In doing the evaluation both objective data on effort, schedule and functionality compliance and subjective assessments of project success from key stakeholders, as customer, user, project manager, developers and management should be gathered.
Acknowledgments
The work with this thesis has been made challenging, interesting, educating and entertaining thanks to my competent, enthusiastic and available supervisors Doctor Kjetil Moløkken-Østvold and Professor Magne Jørgensen.

I would also like to extend a warm thanks to Christian Stensholt at Bouvet. Without him the study would have been impossible to do.

Finally I would like to thank the students and employees at Simula Research Laboratory for creating a stimulating environment which has made my work with the thesis easier to endure.

Organization of Thesis
This thesis presents empirical research on software effort estimation accuracy. At the end of this document four papers are enclosed.

The first paper, which is found in Appendix A, is on the effect the contractor-customer relationship and customer properties have on estimation accuracy. The second paper, which is found in Appendix B, analyses the effect utilization of experience has on estimation accuracy. The third paper, which is found in Appendix C, looks at the role estimation accuracy plays when software project success is to be assessed. The fourth paper, found in Appendix D, presents a review of different group techniques for doing software effort estimation.

The main document is divided into five sections. In Section 1 a background on process improvement in general and for the software development context especially is given. The challenges of effort estimation and the case for doing process improvement through empirical studies are also presented.

In Section 2 the research topics and the research questions associated with them are presented.

Section 3 presents the research method that has been used to answer these research questions.

In Section 4 the results are presented. The results are organized according to the research topics presented in Section 2 and the four enclosed papers.

In Section 5 conclusions and suggested future research is presented.

The focus when compiling this thesis has been put on the four enclosed papers. The main document gives a general background and describes the research method in more detail than what is done in the papers. Besides that most of what is found in the main document should be seen as a summary of what is stated in the enclosed papers. Therefore the reader is advised to look at the papers for detailed background, results and discussion on the different research topics.
# TABLE OF CONTENTS

1 INTRODUCTION .......................................................................................................................... 9  
1.1 Process improvement .................................................................................................................. 9  
1.2 Software Process Improvement .................................................................................................. 10  
1.3 The estimation problem ............................................................................................................. 11  

2 RESEARCH TOPICS AND RESEARCH QUESTIONS ............................................. 13  
2.1 RT1: The effect customer collaboration have on estimation accuracy .................................... 13  
2.2 RT2: The effect utilizing experience data has on estimation accuracy .................................... 13  
2.3 RT3: The role of estimation accuracy when assessing software project success .................... 13  
2.4 RT4: Group techniques for software effort estimation ............................................................... 14  

3 RESEARCH METHOD .............................................................................................................. 15  
3.1 Definitions ................................................................................................................................ 15  
3.2 The Study .................................................................................................................................. 16  
3.3 Measures .................................................................................................................................. 18  
3.4 Isolation Strategy ....................................................................................................................... 19  
3.5 Treats to validity ....................................................................................................................... 20  

4 RESULTS ................................................................................................................................. 21  
4.1 General results .......................................................................................................................... 21  
4.2 The Relationship between Customer Collaboration and Software Project Overruns” – Conditionally accepted for Agile 2007, Washington DC – Appendix A (RT1) ......................................................... 22  
4.3 “The Impact of Utilizing Experience on Software Effort Estimation Accuracy-An Empirical Study” – to be submitted to a Conference - Appendix B (RT2) ................................................................................. 24  
4.4 “The Role of Estimation Accuracy in Assessing Software Project Success - An Empirical Study” – to be submitted to a Journal – Appendix C (RT3) ........................................................................................................... 26  
4.5 “A Review of Group Techniques for Software Effort Estimation” – Appendix D (RT4) ........... 29  

5 CONCLUSIONS AND FUTURE RESEARCH ....................................................... 33  
5.1 Future research ......................................................................................................................... 33  

REFERENCES .............................................................................................................................. 35
1 Introduction

The goal of all research should be to contribute to the improvement of a given situation. The improvement should come as a result of a change that has been empirically proven to have a positive effect.

"There is nothing more difficult to handle, more doubtful of success, and more dangerous to carry out than initiating changes..." – Machiavelli

Hopefully the empirical research presented in this thesis will make the task of initiating the changes it suggests for improving software effort estimation accuracy somewhat easier.

1.1 Process improvement

In the software engineering discipline improvements are often manifested through improvements in the development process. In that respect it is interesting to have a look at process improvement in a general and historical perspective.

In his book from 1911, “The Art of Scientific Management”, Fredrick W. Taylor pioneered process improvement [1]. Taylor looked at how one could improve the efficiency in the manufacturing industry, and his theories had a vast impact on the organization and utilization of factory workers early in the 20th century. Taylor emphasized the need for structure and repeatability. A worker should be trained for one specific task, optimize how he does that task, and do not worry about the rest of the development process. Taylor’s theories have also received a lot of critiques for the depersonalization of the factory employees. His theories did not take the well-being and personal development of the workers into consideration. His theories represent an extreme top-down management style, where the employees are given very little freedom as to how they wish to carry out their work tasks.

Despite the faults of his theories, Taylor is interesting in a process improvement setting as he was the first who looked at the current situation, found possible improvement factors, tested if they helped, and then argued for applying those who did. This train of thought was further developed by W.E. Deming, who is seen as the father of scientific process improvement [2]. Deming was a strong advocate for measurements and collection of experience data. According to Deming the way in which to do process improvement was through application of possible positive measures, and statistical analysis of their effects. This is the main argumentation behind the plan-do-check-act cycle presented in his process improvement framework, Total Quality Management (TQM) [2].

1.1.1 Freedom and structure in the development process

A wide reaching discussion in process improvement and application is how much structure the process should provide. By adding structure one ensures repeatability and enables learning from experience. However, too much structure limits the freedom and adaptability of the process.

The well-known and widely used waterfall process [3] is an example of a development process that provides a lot of structure. In recent years the waterfall process has received a lot of critic for its limited ability to adapt to the dynamic context in which a software development project is executed. In the software development industry the discussion between structure and freedom has become a discussion between sequential development and more flexible development processes (agile, evolutionary and/or incremental).
Among the supporters for a more flexible development process we find Tom Gilb who in a textbook from 1976, stated 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... [4].”

The same year, Harlan D. Mills wrote in a paper 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 [5].”

The members of agile community are strong supporters for less structure and more freedom in the development process.

“Welcome changing requirements, even late in development. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale [6].”

Despite all these critics the waterfall model continues to be widely used indicating that it has benefits that can cancel out the loss of freedom and adaptability [7].

The conflict between freedom and structure, or freedom and security, are as with development processes, also valid for most other areas in the world. As early as in 1651 Thomas Hobbes pointed to the need for some structure in order to avoid chaos [8]. Humans are willing to give away some of their freedom to a ruler or an authority in order to achieve some structure and security in society. How much freedom one should sacrifice in order to achieve the sufficient level of security is a constant debate. In George Orwell’s classic book “1984”, a society where the individual freedom was reduced to a minimum is described [9]. Being extremely inhumane and having many obvious faults, the society was extremely secure if you followed the strict structure and guidelines given by the authorities.

The discussion on freedom and security has many similarities with the discussion on what is the right amount of structure to have in a development process. The aim is to find the best mix of freedom and structure which lies between the chaotic state of nature described by Hobbes, and the totalitarian society described by Orwell.

1.2 Software Process Improvement

To improve the software development process is named as a key challenge in society by both the Unites States government and the European Union [10]. To take on this task by collecting empirical data has many advocates within the software engineering field. In accordance with the theories presented by Deming, Watts Humphrey underlines the need for data collection and feedback when suggesting how software developers could improve their performance [11]. He emphasizes that both to produce better outcomes and to learn for the future is difficult if one does not gather and utilize empirical data in a systematic manner. Conradi, Dybå et. al argue that measurements are essential when doing software process improvement [10].

Several process improvement frameworks specially designed for the software development setting exists. The most complex and comprehensive frameworks are the Capability Maturity Model Integrated (CMMI)[12] and the ISO 9000-framework [13]. The Quality Improvement Paradigm (QIP) is a lighter framework, which is more tailored towards the dynamic setting
and volatile environment in which software development projects are carried out [14]. All these frameworks carry with them the ideas in the plan-do-check-act cycle introduced by Deming, where focus is put on measurement and iteration.

In a research setting it is interesting to note that collection of empirical data on the current situation in order to carry out process improvement has strong historical support. The research presented in this thesis builds on that tradition, and wishes to contribute in the constant effort by practitioners and researchers to optimize the software development processes. Within the general field of software process improvement this thesis has its focus on software effort estimation accuracy.

1.3 The estimation problem
Estimation is defined by the Oxford English Dictionary as “The action of appraising, assessing, or valuing“ or “The process of forming an approximate notion of (numbers, quantities, magnitudes, etc.) without actual enumeration or measurement [15]. From this definition it follows that the task of estimation is not easy to do precisely.

The problem with estimating the scale of a software project is, like with all other activities concerning guessing what will happen in the future, that there is no way to be certain.

Project overruns is a major and persistent challenge in software development [16]. Research has shown that the average effort overrun in software development projects is about 30%-40% [16]. The task of estimation is an important part of all software engineering projects, and the ability to produce accurate estimates has an impact on key economic processes as budgeting and bid proposals. Inaccurate estimates may results in the wrong projects being selected. [17-19], poor resource allocation and poor quality software [20]. Accurate effort estimates are also an important tool in project planning and resource allocation [21]. Traditionally compliance to estimated cost and schedule has also been seen as important to achieve project success [22].

According to Rowe and Wright, the best way to estimate, is to use previously recorded data from a similar setting [23]. The problem is that many software companies fail to learn from their mistakes, and do not record data which could help them become less dependent of human judgment. As statistical data rarely is available, expert judgment is most frequently used [24]. Barry Boehm defines expert judgment as

“Consulting with one or more experts who use their expertise and understanding of the proposed project to arrive at an estimate of its cost [25].”

1.3.1 The human factor
Reasons for the continuing cost overrun problems could be lack in estimation ability and the use of inefficient techniques when producing the estimates. As expert judgment estimation is still the predominate way of doing effort estimation, one also has to consider the human factor in deriving estimates [26].

Humans have a general overconfidence in own abilities, which also has its effect on software effort estimation. Overconfidence leads to an underestimation of effort needed to complete a task. It is also dangerous to view software estimation as a rational exercise. The estimation process is not completely rational, where the only objective is an accurate estimate. We have to recognize that goals such as pleasing managers influence the estimation process. [26]. It is
likely that the actual estimates are biased in order to fulfil other goals than reaching the most accurate estimate.

From the estimation process to the humans involved, it is obvious that there is an improvement potential in the way effort estimation is done in the software development industry. A way in which to help fulfilling this potential is by, in line with suggested software process improvement frameworks, doing empirical research on different estimation factors and approaches, and to use collected data as a contribution in the effort to develop best practices for different software effort estimation settings.
2 Research topics and research questions
This section gives a presentation of the research topics and the research question associated assessed in this thesis.

2.1 RT1: The effect customer collaboration have on estimation accuracy
The growing focus on client involvement in the software development process, e.g. the agile movement [27], makes it interesting to investigate the possible impacts of contractor-client relationship and client properties on effort estimation accuracy.

RQ1.1: Do projects with daily contact between contractors and customers have a lesser magnitude of effort overruns than do other projects?

RQ1.2: Do projects that employ contracts facilitating risk-sharing have a lesser magnitude of effort overruns than do other projects?

RQ1.3: Do projects where the customer is perceived to have a good procurement capability have a lesser magnitude of effort overruns than do other projects?

2.2 RT2: The effect utilizing experience data has on estimation accuracy
The utilization and gathering of experience data is widely suggested as a mean to help software process improvement. In that respect it is interesting to investigate the possible impact of experience data on estimation accuracy. The impact of two related, and frequently used, tools for improving estimation accuracy, estimation models and checklist, has also been looked at.

RQ2.1: Do projects in which data from experience is utilized have a lesser magnitude of effort overrun than do other projects?

RQ2.2: Do projects in which an estimation model is utilized experience a lesser magnitude of effort overrun than do other projects?

RQ2.3: Do projects in which checklists are utilized experience a lesser magnitude of effort overrun than do other projects?

2.3 RT3: The role of estimation accuracy when assessing software project success
This thesis has its focus on software estimation accuracy. In that respect it is interesting to take a step away and have a look at which role estimation accuracy plays in determining project success. In addition to effort and schedule estimation accuracy, the role of delivered functionality has been investigated.

RQ3.1: Is there a correlation between the project managers’ assessment of project success and effort estimation accuracy?

RQ3.2: Is there a correlation between the project managers’ assessment of project success and schedule estimation accuracy?

RQ3.3: Is there a correlation between the project managers’ assessment of project success and delivered functionality?
2.4 RT4: Group techniques for software effort estimation

This thesis will have a look at a very specific area in which the estimation process could be improved to gain higher estimation accuracy. To utilize groups in solving complex tasks is advocated by many. Research has also shown that it might be a good idea to do software effort estimation as a group, and therefore it has been seen as important to have a look at different group techniques for software effort estimation.

*RQ4.1:* Which group techniques for software effort estimation exist?
3 Research Method

In answering the research questions, empirical data has been gathered through interviews with project managers at a software consultancy.

In this section a presentation on how this data was collected, how it was analyzed and potential weaknesses in the data will be given. In subsection 3.1 several terms have been defined. Subsection 3.2 presents the study and the explored properties. The different measures used in the analysis are explained in subsection 3.3. An accord for the applied isolation strategies are given in subsection 3.4. Subsection 3.5 presents potential treats to validity.

The framework for analyzing Software Cost Estimation accuracy suggested by Jørgensen and Grimstad [28] will to as large an extent as possible be followed. The framework is a good fit as Jørgensen and Grimstad claim that it will be suited for small data sets where random allocation of treatment is not possible, as is the case in the study presented here.

3.1 Definitions

As the main focus of this thesis is software effort estimation, it is necessary do define what is understood with the term estimate.

As part of RT1 the impact contract types have on estimation accuracy is looked at. Target price contracts were used in many of the project. As this is a relatively new contract type that few are familiar with a definition of it is given below.

When considering RT2 the terms experience data, checklists and model based estimation are encountered, and therefore they have been defined below.

A main part in a RT3 is to have a critical look at what constitutes software project success. However a definition of project success is not provided. The reason for that is that it is one of the goals of this study to investigate which factors project success is comprised of. In the collection of data relating to project success it has been up to the respondents to apply their personal interpretation of project success. This is in accordance with previous research [21] which also allowed the respondents to use their own implicit definition of project success.

Estimate

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 the research. If the goal is to compare the actual effort with the estimated effort, as it is here, it is meaningful to use the most likely estimates at the planning stage, instead of, for example, project bids. The latter may be affected by outside factors, such as market competition.

This definition given by Grimstad will be applied:

“The term software development effort estimate is understood as a prediction of the effort most likely required to implement a software development project [29].”
Target price contract

A target price contract is a contract that shares the risk of overruns between the contractor and the customer. The contract defines a target of effort (in hours), and often a date, for completing the project.

If the project is completed, and has used fewer hours than defined by the contract, the contractor and customer share the profit. If more effort than what is defined by the target is needed, the contractor and the customer share the extra costs. This is set up so that the customer pays a percentage, often 50%, of full price for the hours that make up the difference between the target and actual effort.

E.g., the contractor charges $150/hour and the target for completing the project is set to 1000 hours. If 800 hours is used to complete the project the customer has to pay full price for the 800 hours used and $75 for the remaining 200 hours (not used). If 1200 hours is used the customer pays full price for the 1000 hours set by the target, and $75 for the additional 200 hours.

A target contract can also be set up with floors and ceilings defining the minimum and maximum hours that the contractor could charge the customer.

Experience data

Experience data is understood as recorded data from previous projects that has been formally or informally systemized and made available for utilization in future estimation processes.

Checklist

A checklist is understood as a list that helps the estimator to remember tasks and other aspects to be considered when doing effort estimation.

Model based estimation

When differentiating between model-based, expert-based and a combination of the two the definitions presented in a recent paper by Jørgensen is applied.

“Judgment-based effort estimates (are) based on a tacit (intuition-based) quantification step and model-based effort estimates (are) based on a deliberate (mechanical) quantification step[30].”

3.2 The Study

All data presented in this thesis was gathered through a single study, which was performed at a medium sized Norwegian software consultancy. At the time of the study the company had about 300 employees. The company operates as an independent contractor and offers a wide range of complete software solutions to its various customers. By collecting data from only one company it is easier to better isolate company dependant factors [31]. On the flip side collecting data from only one company makes it possible for company specific properties to have a strong impact on the results, making them less likely to have external validity.

The data was collected through interviews with the project managers of 18 different projects. These projects were selected by the company, which was not informed of the research questions. The inclusion criteria were that relevant project data was stored and available, that we had access to interview the responsible project managers, and that the projects had a
workload of at least 100 man-hours. The latter criteria is in line with previous surveys, in which “trivial tasks routinely handled without effort estimation” were also filtered out [18].

The data was collected via personal interviews, which yields data of high quality, and ensures that ambiguities are resolved [32]. It also allows for the respondents to add valuable information that is not possible to include when completing a predefined questionnaire. Another point in favour of this approach is that 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 investigating as many projects as would be possible by using mailed questionnaires.

Each interview lasted between 20 and 80 minutes. 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.

3.2.1 Explored properties

The goal of the interviews was to explore different properties related to the projects, their execution and the estimation process that had been used to estimate size. Some of the properties could be fairly objectively assessed, while others depended on the subjective assessments of the project managers.

Below a classification of the explored properties is presented. The properties listed under objective have been collected without any interpretation from the project managers being necessary. The properties listed under fairly objective, are objective in their nature, but as it involves an element of project manager interpretation they are divided from the purely objective measures. The subjective measures rely entirely on the individual project manager’s perceptions and assessments.

The reason for classifying the different properties is to make sure that their limitations and assumptions is considered when they are used in the analysis.

Objective
- estimated effort to complete project
- actual effort to complete project
- estimated delivery date
- actual delivery date
- type of contract

Fairly objective
- customer contact frequency
- the utilization of experience data in the estimation process
- the use of an estimation model to derive the estimates
- the use of checklist in the estimation process

Subjective
- the customer’s procurement capability
- perception of reasons for estimation inaccuracy
- perception of what would be the ideal estimation process
• assessment of project success
• perception of reasons for project success/failure

In addition, several questions were aimed at general project properties. In addition to having their own value, these questions were also relevant for isolating effects and exploring confounding factors in the study. These possible confounding factors included project size and perceived technical knowledge.

3.3 Measures

In order to compare actual effort and estimated effort, and measure any differences in project overruns dependent on the studied properties, the BREbias measure has been used. It has previously used in related research, e.g., [31, 33], and is calculated as:

\[ BREbias = \frac{(x - y)}{\min(x, y)}, \]

\[ x = \text{actual and } y = \text{estimated value}. \]

The BREbias measures both the magnitude and direction of effect when comparing the actual effort to estimated effort. BREbias is based on the Balanced Relative Error (BRE) [34, 35].

Even though MRE has been the most widely used measure of estimation accuracy [36], one must be aware that it has unfortunate properties [34, 37]. The main concern for our case is the fact that underestimated and overestimated projects are weighted unevenly.

Another take on estimation accuracy is to analyze how well a project complies with its initial schedule. In assessing a project’s schedule compliance a delay factor, which is hoped to give a comparable measure of a project’s relative schedule compliance, has been calculated. The delay factor is computed by looking at the delivery delay in days relative to actual effort. As actual effort has been registered in hours, the actual effort in hours has been divided by eight to get person days, which is a more comparable figure.

\[ Delay \text{ factor} = \frac{Delivery \text{ delay (calendar days)}}{Effort (person days)} \]

The delay factor may not be a very robust measure. However, it gives a good enough picture of the relative delivery delay, for it to be applied in the analysis of schedule compliance.

To measure the size of any difference, we used Cohen’s size of effect measure (d) [38], where:

\[ d = \frac{sample1 - sample2}{pooledStdDev} \]

Statistical Kruskal-Wallis test[39] and t-test[40] for differences have been done where those are appropriate.

The Kruskal-Wallis test is a non-parametric method for testing equality of population medians. The test does not assume a normal population, and the groups do not have to be
equal in size [39]. A p-value from a Kruskal-Wallis test below the significance level indicates a not random difference between the compared populations.

A t-test is based on the means and standard deviation of the different groups. As the Kruskal-Wallis test it results in a p-value, which if is below the level of significance indicates that the difference between the compared populations are not random. A t-test builds on a number of assumptions for the populations that are be compared. Among them are a normal distribution of data and quality of variance [40]. This is seldom the case when one has a small data sample as is the case in this thesis. The results of t-test presented here should therefore be used and interpreted with strong caution. However, it provides a useful insight on the relationship between the means and distributions of the compared groups.

3.4 Isolation Strategy
In accordance with the framework presented by Jørgensen and Grimstad [28] different isolation techniques have been discussed and those possible to apply have been applied. The reason for doing isolation is to check for possible confounding factors that may have biased the results. All the four isolation strategies purposed by the framework, randomization, grouping, adjustment and exclusion have been considered.

Randomization
Since the treatment was already applied before the study was started it was impossible to do randomization of treatment. This is an unfortunate, but implicit property of doing empirical research on actual software development projects. In order to apply treatments randomly, one would have to do a controlled experiment, which is likely to result in a substantial loss of realism.

Grouping
Grouping has been done for the variables where this is possible, and a thorough analysis has been done. The possible effect of confounding factors has been discussed with regards to all the four research topics and the research questions associated with them. These discussions are presented in Appendix A, B, C and D.

Adjustment
It is hard to measure the difference between planned outcome, and actual outcome. There has been done no adjustment for any differences. It is assumed that the differences between planned and actual outcome are relatively evenly distributed on the different categories.

Exclusion
Projects giving outliers in one or more categories have been considered for exclusion. Here the framework lists different factors that should be analyzed before including a project in the dataset. In other words if a project suffer from either inconsistent use of terminology, logging problems, difference between planned and actual outcome or measurement selection bias it should be considered to be excluded from the sample. Such a consideration has been done for all of the projects, but there has not been found a need to exclude any of them from the sample. Projects have however been excluded from specific analysis as a result of lack of data due to project managers’ wish to withhold information, or the lack of access to the necessary information.
3.5 Treats to validity

In general, the sample size of this study is too small for statistical analysis, and should be used with caution. As in a previous related study [31], collecting in-depth, high-quality data on a wide range of properties has been selected instead of using mailed questionnaires. Regarding internal validity, one must also be aware that many of the measures are subjective perceptions of the respondents, and not objective facts. In a study like this, which is not a controlled experiment, cause and relationship effects are impossible to pinpoint.

Regarding external validity, the size of the overruns is similar to previous surveys and case studies, and indications that the sample is not particularly biased. However, in other environments, such as in-house development, other factors have to be taken into account. Therefore, external validity is limited to contractors developing projects of a similar size and complexity.

As with all research on effort estimation there is the problem of difference in estimated functionality and implemented functionality [17]. As mentioned under isolation strategy no active measure to preempt or handle this problem has been done.

In this study all data and viewpoints come from project managers. By only collecting data from one of the several project stakeholders it is likely that the view presented here is incomplete and somewhat biased [41].

Regarding the use of satisfaction assessments there are issues that one should be aware of. When asked to assess success, one should not downplay the human urge to feel that something one took part in was a success [42]. If one could manage to believe that something was a success one would feel better about oneself, and therefore one tends to be optimistic about things in which one has played a part. Theories have been presented that claim that human’s satisfaction assessment will be close to constant, independent of the actual outcome, effort or behaviour that is assessed [43].
4 Results
In this section a presentation of the general results regarding effort overruns for the 18 studied projects will be presented. The findings regarding the four research topics and the research questions associated with them will be given in dedicated subsections.

4.1 General results
In this section general results regarding effort and schedule estimation accuracy and project size (measured in actual hours of effort) will be given. This results are presented in order to provide the reader with information about context in which the other, more specific, results are presented and discussed.

4.1.1 Effort Estimation accuracy
Of the 18 finished projects, three were overestimated, one was on target, while 14 were underestimated.

Table 1: BREbias for all projects

<table>
<thead>
<tr>
<th>N</th>
<th>Mean</th>
<th>Median</th>
<th>Min</th>
<th>Max</th>
<th>StDev</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>0.27</td>
<td>0.22</td>
<td>-0.230</td>
<td>1.70</td>
<td>0.46</td>
</tr>
</tbody>
</table>

The mean and median BREbias is 0.27 and 0.22 respectively (this corresponds to effort overruns of 27% and 27% respectively). This is in line with findings in previous surveys and case studies on software estimation, and indicates that with regards to overruns, the projects studied were fairly representative [16].

4.1.2 Schedule Estimation accuracy
For the analysis of adherence to schedule, one project was excluded due to a significant postponement which was out of the contractor’s control. Of the remaining 17 projects one was completed before schedule, nine after schedule and seven on schedule.

Table 2: Delay factor for all projects

<table>
<thead>
<tr>
<th>N</th>
<th>Mean</th>
<th>Median</th>
<th>Min</th>
<th>Max</th>
<th>StDev</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>0.11</td>
<td>0.09</td>
<td>-0.07</td>
<td>0.45</td>
<td>0.16</td>
</tr>
</tbody>
</table>

The mean delivery delay factor was 0.11, while the median was 0.09.

4.1.3 Project size
The 18 studied projects vary largely in size. The smallest project took 947 hours to complete, while the biggest took 16 000 hours.

Table 3: Project size

<table>
<thead>
<tr>
<th>N</th>
<th>Mean</th>
<th>Median</th>
<th>Min</th>
<th>Max</th>
<th>StDev</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>5590</td>
<td>2604</td>
<td>947</td>
<td>16000</td>
<td>5230</td>
</tr>
</tbody>
</table>

The mean project size was 5590, while the median was 2604.
4.2 The Relationship between Customer Collaboration and Software Project Overruns” – Conditionally accepted for Agile 2007, Washington DC – Appendix A (RT1)

The use of agile development methods are getting increasingly popular. In an agile project the relationship between the contractor and the customer is important. The assumption by the agile community is that a good relationship to the customer, and through frequent contact with the customer one increases the chances of project success. The effect of collaboration between contractor and customer on estimation accuracy has not been studied before. However, previous studies and different methodologies suggest that different customer properties may affect estimation accuracy.

The contractor-customer relationship and customers properties effect on estimation accuracy was investigated. The factors addressed were contact frequency, contract form and customer procurement ability. The research questions phrased under RQ1.1, RQ1.2 and RQ1.3 were discussed.

4.2.1 RQ1.1: Do projects with daily contact between contractors and customers have a lesser magnitude of effort overruns than do other projects?

The respondents stated how often they communicated with the customer. In this analysis, it has been differentiated between those who had daily communication (11 projects), and those who did not have daily communication (seven projects). The results are displayed in Table 4.

<table>
<thead>
<tr>
<th>Level</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daily</td>
<td>11</td>
<td>0.09</td>
<td>0.19</td>
</tr>
<tr>
<td>Not Daily</td>
<td>7</td>
<td>0.58</td>
<td>0.35</td>
</tr>
</tbody>
</table>

The projects which do not have daily communication with the customer appear to have larger overruns. A Kruskal-Wallis test for difference results in p=0.023. The corresponding size of effect is d=1.25, indicating a large size of effect [38].

4.2.2 RQ1.2: Do projects that employ contracts facilitating risk-sharing have a lesser magnitude of effort overruns than do other projects?

The following types of contracts were employed: 1) by the hour (time and material), 2) fixed price, 3) target price (risk sharing between contractor and customer), and 4) other. The results are displayed in Table 5.

<table>
<thead>
<tr>
<th>Level</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>By the hour</td>
<td>4</td>
<td>0.55</td>
<td>0.37</td>
</tr>
<tr>
<td>Fixed price</td>
<td>5</td>
<td>0.33</td>
<td>0.19</td>
</tr>
<tr>
<td>Target price</td>
<td>7</td>
<td>0.10</td>
<td>0.21</td>
</tr>
<tr>
<td>Other</td>
<td>2</td>
<td>0.13</td>
<td>0.13</td>
</tr>
</tbody>
</table>

There are relatively few observations in the different categories, so a statistical analysis is not included. However there appears to be some differences in overruns related to the different contract types.
4.2.3 RQ1.3: Do projects where the customer is perceived to have a good procurement capability have a lesser magnitude of effort overruns than do other projects?

The customers’ perceived procurement capability was defined by combining the interviewed project managers’ assessment of the customer’s collaboration ability, its IT-skills, its decision ability and its clarity of goals. Based in the assessments a customer score was calculated, where best possible score was 4 (1 on all four properties), while the lowest possible was 20 (5 on all four properties).

The projects were divided in those who received a total customer score of 8 or better (rated as good or better), and those with a total score of 9 or worse.

<table>
<thead>
<tr>
<th>Total score</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good or better</td>
<td>10</td>
<td>0.23</td>
<td>0.15</td>
</tr>
<tr>
<td>Less than good</td>
<td>8</td>
<td>0.33</td>
<td>0.24</td>
</tr>
</tbody>
</table>

There was no apparent, large difference when comparing the total customer score. A Kruskal-Wallis test resulted in p=0.35, Cohen’s d=0.21. However, there is a tendency towards a lesser magnitude of overruns with better customer procurement capability.

4.2.4 Findings

The main finding was that good collaboration with customers, facilitated by frequent communication, was associated with projects that experienced a lesser magnitude of effort overruns. It might also be that risk-sharing contracts can reduce overruns.

In addition, it was not observed a clear relationship between customer procurement capability and magnitude of overruns. This might be due to the small sample size, but it may also be that the results constitute evidence that this frequent explanation of overruns is somewhat exaggerated.

Taken together, the findings presented in this paper indicate that contractors can implement a few key practices to facilitate collaboration with their customers in order to reduce overruns and achieve greater project success.

In addition to what has been observed related to overruns, frequent communication and risk-sharing contracts, may also have benefits that extend beyond the current project. These benefits include fostering a contractor/customer relationship that is beneficial in the long-run and that will bring returning business.

For a more thorough analysis see Appendix A.
4.3 “The Impact of Utilizing Experience on Software Effort Estimation Accuracy–An Empirical Study” – to be submitted to a Conference - Appendix B (RT2)

It is not easy to decide how to approach the task of software effort estimation. Throughout the (albeit short) history of software development, new estimating techniques have emerged continually. Attempts have been made to compare these techniques and derive best practices [17]. However, there seems to be little consensus on which elements and factors increase estimation accuracy.

To collect and utilize experience to improve situations in general has strong historical support [1, 2]. Also within the field of software development and software effort estimation, building on previous experience has been widely suggested as a mean to improve the software estimation ability [11, 18, 44]. To do estimation based on experience from previous project has been labelled estimation by analogy [25]. The basis of estimation by analogy is to base the estimates for a new project on data from one or more completed projects [25]. The strengths in such an approach is that you base your estimates on actual experience [45]. The problem is the often very unique nature of software development projects, which makes it hard to assess how similar a new project is to a previous one. Doing estimation by analogy is, or at least has been, widely utilized in the software industry [46].

There have been studies supporting the use of estimation by analogy [45, 47], and studies finding that estimation by analogy do not lead to more accurate estimates [48]. There also exist diverging views on whether the estimates should be computed directly based on previous projects [47], or if some human consideration should be included in the estimation process [49].

When studying the use of data from experience to enable more accurate software effort estimation, it is interesting to consider two related, and frequently used, tools for improving estimation accuracy. Estimation models and checklists are two specific tools that could be used in the estimation process. Both estimation models and checklists could be customized on the basis of data from experience. However, it is also possible to utilize these tools without the use of data from experience.

The research questions discussed are phrased under RQ2.1, RQ2.2 and RQ2.3.

4.3.1 RQ2.1: Do projects in which data from experience is utilized have a lesser magnitude of effort overrun than do other projects?

The yes category, containing six projects, includes those projects were recorded data were utilized in the estimation process. The no category contains the remaining twelve projects were no experience data was used in the deriving of effort estimates.

Utilizing data from experience seems to lead to more accurate estimates. However, one has to be aware of the small sample size for these kinds of analyses. The d-value indicates that, according to Cohen’s effect classification, the application of data from experience has a large effect [38].

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>6</td>
<td>0.06</td>
<td>0.13</td>
</tr>
<tr>
<td>No</td>
<td>12</td>
<td>0.37</td>
<td>0.27</td>
</tr>
<tr>
<td>K-W</td>
<td>0.160</td>
<td>T-test</td>
<td>0.052</td>
</tr>
</tbody>
</table>
4.3.2 RQ2.2: Do projects in which an estimation model is utilized experience a lesser magnitude of effort overrun than do other projects?

None of the projects relied entirely on a model when deriving the estimates. Most of those that used a combination of a model and experts utilized elements from a company customized model or a self-developed model. A similar distribution was found in [50].

Table 8: BREbias based on estimation model utilization

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model-based</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Combination</td>
<td>8</td>
<td>0.34</td>
<td>0.26</td>
</tr>
<tr>
<td>Expert-based</td>
<td>10</td>
<td>0.20</td>
<td>0.20</td>
</tr>
<tr>
<td><strong>K-W</strong></td>
<td></td>
<td><strong>0.48</strong></td>
<td></td>
</tr>
<tr>
<td><strong>T-test</strong></td>
<td></td>
<td><strong>0.550</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Cohen's d</strong></td>
<td></td>
<td><strong>0.30</strong></td>
<td></td>
</tr>
</tbody>
</table>

The findings here show that the overruns are lower for those projects in which the estimating is expert-based than for those where a combination of expert-based and model-based methods is applied. These results are in agreement with the findings in [30, 48], but are contrary to those in [50] where it was found that a combination approach outperformed expert-based estimation.

4.3.3 RQ2.3: Do projects in which checklists are utilized experience a lesser magnitude of effort overrun than do other projects?

Checklists were used when estimating in seven projects and not used in 10. For one of the projects, the project managers were uncertain and did not answer the question on checklists.

Table 9: BREbias based on checklists

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>7</td>
<td>0.10</td>
<td>0.19</td>
</tr>
<tr>
<td>No</td>
<td>10</td>
<td>0.38</td>
<td>0.29</td>
</tr>
<tr>
<td><strong>K-W</strong></td>
<td></td>
<td><strong>0.17</strong></td>
<td></td>
</tr>
<tr>
<td><strong>T-test</strong></td>
<td></td>
<td><strong>0.09</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Cohen's d</strong></td>
<td></td>
<td><strong>0.61</strong></td>
<td></td>
</tr>
</tbody>
</table>

The use of checklists seems to have a positive impact on estimation accuracy. Both the mean and median are substantially lower for the group of projects where a checklist has been used in the estimation processes.

As with the use of data from experience, the t-test gives a relatively low p-value and the Cohen’s d-value that indicates a large effect [38].

4.3.4 Findings

It was found that the use of data from experience increased estimation accuracy. Regarding the use of estimation models the results showed that it did not improve estimation accuracy. The results show that there is reason to believe that estimation accuracy can be improved by the use of checklists.

Fortunately for the software development community, it might be easy to implement properties that lead to an increase in estimation accuracy. As found in previous studies, the solution to the challenge of estimating accurately is, perhaps, not to add complex and intricate models and methods for deriving the estimates [17, 24].

For a more thorough analysis see Appendix B.
4.4 “The Role of Estimation Accuracy in Assessing Software Project Success - An Empirical Study” – to be submitted to a Journal – Appendix C (RT3)

What constitutes software project success is not precisely defined. Traditionally, project success has been measured on three axes: effort, schedule and functionality. Increasingly, factors such as outcome satisfaction have also been emphasized when assessing project success.

According to Procaccino and Verner, the traditional view of what encompasses software project success is as follows [22]:

“The success of any software development project has traditionally been ‘defined’ from the organizational perspective, whereby a project should deliver agreed upon functionality on time and within budget”.

In a recent paper, Agarwal and Rathod present this definition [51], originally by Baker et al., of project success [52]:

“The project is considered an overall success if the project meets the technical performance specification and/or mission to be performed, and if there is a high level of satisfaction concerning the project outcome among key people in the parent organization, key people in the project team and key users or clientele of the project effort.”

Agarwal and Rathod would like to see project success defined by a combination of the two definitions given here. Subjective assessments, mainly from people not directly involved in the development of the outcome, and objective data as schedule overruns and economic profit should be combined to assess software project success [51].

As other factors emerge in the assessment of project success it is interesting to look at the role of estimation accuracy when assessing project success.

The issue of investigating the role of effort and schedule estimation accuracy when determining project success has here been approached by trying to determine how project managers’ assessment of project success correlate with these factors.

When analysing the project managers’ assessment of project success, one has to rely on data based on the project managers’ subjective perceptions. Analyzing and utilizing subjective data poses challenges. This is especially true when humans’ satisfaction or success rating is to be measured. The psychological phenomenon cognitive dissonance leads humans to focus on the positive aspects of our own actions [42]. This results in an overrepresentation of high satisfaction assessment ratings, which is independent of the actual outcome, effort or behaviour that is assessed [43]. A detailed description of the cognitive dissonance phenomena and the constant satisfaction theory is given in Appendix C.

When investigation the role of estimation accuracy the research questions phrased under RQ3.1, RQ3.2 and RQ3.3 were discussed.
4.4.1 RQ3.1: Is there a correlation between the project managers’ assessment of project success and effort estimation accuracy?

The respondents rated the success of the project on a five-point Likert scale with the categories: very, high, medium, low and very low. The project managers were, in general, satisfied with their projects, with as many as 14 projects being assessed with a very high or high success rating. This corresponds well with the cognitive dissonance and constant satisfaction theories.

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>St.Dev.</th>
<th>Min</th>
<th>Median</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very high</td>
<td>8</td>
<td>0,31</td>
<td>0,61</td>
<td>-0,30</td>
<td>0,24</td>
<td>1,70</td>
</tr>
<tr>
<td>High</td>
<td>6</td>
<td>0,15</td>
<td>0,22</td>
<td>-0,25</td>
<td>0,19</td>
<td>0,36</td>
</tr>
<tr>
<td>Medium</td>
<td>2</td>
<td>0,10</td>
<td>0,15</td>
<td>0,00</td>
<td>0,10</td>
<td>0,21</td>
</tr>
<tr>
<td>Low</td>
<td>1</td>
<td>0,96</td>
<td>0,96</td>
<td>0,96</td>
<td>0,96</td>
<td>0,96</td>
</tr>
<tr>
<td>Very low</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 10 shows that effort overruns are somewhat higher for the projects that are assessed as having a very high success rating compared to the ones that are assessed as having a high success rating. The overruns are even a bit lower for the two projects that are assessed as being a medium success.

Seemingly, effort estimation accuracy is not an important factor for project managers when they are assessing project success. This is in agreement with findings in [22, 53] but goes against findings in [21, 51, 54].

4.4.2 RQ3.2: Is there a correlation between the project managers’ assessment of project success and schedule estimation accuracy?

Perceived project success and schedule estimation accuracy are compared in Table 11. Due to a significant postponement that was out of the contractor’s control, one project was excluded from the analysis.

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>St.Dev.</th>
<th>Min</th>
<th>Median</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very high</td>
<td>7</td>
<td>0,09</td>
<td>0,18</td>
<td>-0,07</td>
<td>0</td>
<td>0,45</td>
</tr>
<tr>
<td>High</td>
<td>6</td>
<td>0,16</td>
<td>0,18</td>
<td>0</td>
<td>0,10</td>
<td>0,47</td>
</tr>
<tr>
<td>Medium</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Low</td>
<td>1</td>
<td>0,26</td>
<td>0,26</td>
<td>0,26</td>
<td>0,26</td>
<td>0,26</td>
</tr>
<tr>
<td>Very low</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 11 shows that the delay factor is slightly less for the very high category than for the high category. In general, the projects are more in compliance with the schedule estimates than with the effort estimates, and the differences in delay factor between the different satisfaction levels are not large.

However, it is interesting to note that project managers seem to be more satisfied with projects that are delivered according to, or at least closer, to schedule. There seems to be a slight positive correlation between adherence to schedule and project managers’ success assessment.
4.4.3 RQ3.3: Is there a correlation between the project managers’ assessment of project success and delivered functionality?

The assessment of the percentage of delivered functionality is subjective, and will therefore have several potential weaknesses. We do, however, think that it will give some indication of the importance of delivered functionality when assessing project success.

Table 12: Delivered functionality based on project manager satisfaction

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>St.Dev.</th>
<th>Min</th>
<th>Median</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very high</td>
<td>8</td>
<td>128.1</td>
<td>45.2</td>
<td>90</td>
<td>115</td>
<td>230</td>
</tr>
<tr>
<td>High</td>
<td>6</td>
<td>104.2</td>
<td>12.01</td>
<td>90</td>
<td>100</td>
<td>125</td>
</tr>
<tr>
<td>Medium</td>
<td>2</td>
<td>92.50</td>
<td>3.54</td>
<td>90</td>
<td>92.5</td>
<td>95</td>
</tr>
<tr>
<td>Low</td>
<td>1</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Very low</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 12 shows that for the very high project success category, the project managers claimed, on average, that they delivered 128% of the specified functionality. Although the percentage was lower for the high category it was more than 100%.

The results indicate that project managers are more satisfied with projects that they believe to have delivered more functionality. This is in line with the findings in [22] and the results presented in the next subsection.

4.4.4 Findings

Progress has been made regarding what constitutes software project success and the role that estimation accuracy has in determining it. The investigation of the relationship between estimation accuracy, delivered functionality, and project managers’ assessment of success showed a slight negative correlation between project manager assessments and effort estimation accuracy, and a slight positive correlation for schedule estimation accuracy and project manager assessments. A slight positive correlation was also found for delivered functionality and assessments of project success.

In general, little correlation was found. This might be due to the problems with subjective assessments and the theory of constant satisfaction, which implies that one should not put much faith in the assessments of project success made by project managers.

Another view is that estimation accuracy and the measure associated with the traditional iron triangle is less important, or at least not significant, for achieving project success.

The conclusion is that both the objective data on estimation accuracy and the subjective assessment of project managers have value. An even better determination of project success would be achieved if assessments and data from a wide range of project stakeholders where gathered, placing special emphasis on the clients’ and users’ assessments of the project outcome, as these are the stakeholders the product is developed for.

Future project evaluations should strive to gather information from all stakeholders: clients, management, developers, and project managers. In addition, estimation accuracy and delivered functionality should be measured. The different factors should be combined and the relationship between them investigated. This would lead to a better framework for both defining and determining project success.
4.5 “A Review of Group Techniques for Software Effort Estimation” – Appendix D (RT4)

Reasons for the continuing cost overrun problems [16] could be lack in estimation ability and the use of inefficient techniques when producing the estimates. Many argue that the use of groups, where several individual estimates are combined, is a good plan for optimizing the estimates [24, 25, 55-57].

The importance of dealing with biases is central in the argument for combining estimates. To prevent over optimism or over pessimism one wishes to cancel out biases. Boehm claims that using more than one expert will lead to less biased estimates [25]. Interacting groups also have positive attributes like knowledge from a variety of sources and creative synthesis [23].

In addition to the positive effects of utilizing groups, several potential undesired effects are identified. Here the focus will be on coordination issues and social dynamics. In addition groupthink and polarization, which can be seen as special and typical effects of the social dynamics in a group, will be discussed. A detailed presentation of these effects is given in Appendix D.

Several suggestions on how to structure a group process in order to utilize the positive attributes, while at the same time preempting the undesired effects have emerged. As utilizing a group to do software effort estimation is supported by several studies, and intuitively seem to be a good idea, a review on which group techniques for effort estimation exist was done.

4.5.1 RQ4.1: Which group techniques for software effort estimation exist?

The basic way of reaching a common estimate through group interaction is using what is labelled as an unstructured group. An unstructured group is what we normally refer to as a group, meaning several people coming together, sharing their viewpoints and reaching a common decision. The word unstructured is used as the group is not given any instructions to work according to a specified structure.

Four different structured group processes will be presented. The processes vary largely in how much structure they provide and how much face-to-face interaction between the group members they allow. The techniques that will be discussed are:

- Delphi
- Wideband Delphi
- Planning Poker
- Decision Markets

In [33] Moløkken-Ostvold and Haugen categorize different group techniques based on structure, anonymity, interaction and overhead. In Table 13 the categorization has been extended to also include Decision Markets. A full presentation of the different group techniques is given in Appendix D.
Table 13: Group technique overview

<table>
<thead>
<tr>
<th></th>
<th>Structure</th>
<th>Anonymity</th>
<th>Interaction</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unstructured group</td>
<td>None</td>
<td>No</td>
<td>Yes</td>
<td>Limited</td>
</tr>
<tr>
<td>Delphi</td>
<td>Heavy</td>
<td>Yes</td>
<td>No</td>
<td>Major</td>
</tr>
<tr>
<td>Wideband Delphi</td>
<td>Moderate</td>
<td>Limited</td>
<td>Limited</td>
<td>Moderate</td>
</tr>
<tr>
<td>Planning Poker</td>
<td>Light</td>
<td>No</td>
<td>Yes</td>
<td>Limited</td>
</tr>
<tr>
<td>Decision Markets</td>
<td>Heavy</td>
<td>Yes</td>
<td>No</td>
<td>Major</td>
</tr>
</tbody>
</table>

The table shows that Delphi and Decision Markets have many of the same properties. Wideband Delphi and Planning Poker are also quite similar. The processes’ take on anonymity seems to be defining for the rest of the properties. To have anonymity no face-to-face interaction can take place, in order to achieve that one needs more structure in the process, which leads to more overhead.

The table below show a presentation of the different group processes’ ability to preempt undesired effects of doing problem solving as a group. All the group processes encompasses a trade-off between preempting undesired group effects, increasing the probability and ability to take advantage of potential positive group effects and generated overhead through a high need for structure.

Table 14: The group techniques preemption abilities

<table>
<thead>
<tr>
<th></th>
<th>Preempts</th>
<th></th>
<th>Social and political conflicts</th>
<th>Groupthink and polarization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coordination</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unstructured Group</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Delphi</td>
<td>Yes</td>
<td>Yes</td>
<td>Limited</td>
<td></td>
</tr>
<tr>
<td>Wideband Delphi</td>
<td>Yes</td>
<td>Limited</td>
<td>Limited</td>
<td></td>
</tr>
<tr>
<td>Planning Poker</td>
<td>Yes</td>
<td>Limited</td>
<td>Limited</td>
<td></td>
</tr>
<tr>
<td>Decision Markets</td>
<td>Yes</td>
<td>Yes</td>
<td>Limited</td>
<td></td>
</tr>
</tbody>
</table>

As mentioned above, how important anonymity is seen to be, has a significant impact on the group process. The focus on anonymity is a result of a belief in the need for independence between the different group members, in order to fully utilize the group’s potential [58].
Through anonymity one hopes that the different group members will voice their own opinion without being influenced by political, social or other considerations.

An unstructured group process does not do anything to try to protect the anonymity and independence of the group members, resulting in no preemption of any of the undesired group effects.

Wideband Delphi and Planning Poker represent a combination between protecting independence and allowing face-to-face communication. In different ways the two techniques make sure that the group members’ initial estimates are independent of each other. Wideband Delphi does it through making the initial estimates anonymous, while in Planning Poker it is done through the simultaneous showing of the estimates. However, after the initial estimates are revealed they allow face-to-face discussions. The hope is that the potential positive effects of face-to-face discussions will be worth the increase in the chances for political and social conflict, polarization and groupthink.

Both Delphi and Decision Markets have a stronger emphasis on the need for independence between the group members in order for them to produce a good outcome. That is why the two techniques in different ways sets up structure and generates overhead to protect the anonymity of the group members. By not allowing any face-to-face interaction one preempts the possibility for groupthink and political and social conflicts. As both techniques have a way in which the average or aggregated opinions are reported back to the group members, the possibility for polarization is not preempted as good as the other undesired effects.

Delphi, Wideband Delphi, Planning Poker and Decision Markets are all build on the assumption that to do software effort estimation as a group is a good idea. In different ways they try to make sure that the knowledge and information held by the group members are shared, so that they together can reach a good estimate.

Little research is done on the use of these different estimate combination techniques for software effort estimation, and there is an obvious need for more research on the different techniques abilities to produce accurate software effort estimates.

For a more thorough analysis see Appendix D.
5 Conclusions and future research

The findings in this thesis suggest several measures that could be implemented by software development contractors in order to improve their estimation ability.

The investigation of customer relationship and customer properties showed that it might be wise to have daily contact with the customer, and to use a risk-sharing contract.

Not surprisingly the analysis of the effect of utilizing experience data in the estimation process showed that gathering and utilizing experience could have beneficial effects. Regarding the use of estimation models the results showed that it did not improve estimation accuracy. The results show that there is reason to believe that estimation accuracy can be improved by the use of checklists.

The investigation of the relationship between estimation accuracy, delivered functionality, and project managers’ assessment of success showed a slight negative correlation between project manager assessments and effort estimation accuracy, and a slight positive correlation for schedule estimation accuracy and project manager assessments. A slight positive correlation was also found for delivered functionality and assessments of project success. The conclusion is that both the objective data on estimation accuracy and the subjective assessment of project managers have value. An even better determination of project success would be achieved if assessments and data from a wide range of project stakeholders where gathered.

To sum up, the suggestions to software contractors are to:

• Involve the customer, and nurture the customer relationship
• Add some repeatable structure to the estimation process, but be careful not to add too much
• Gather and utilize experience data
• Evaluate projects when they are done, and in doing the evaluation, apply an inclusion framework for what encompassed project success. Objective data on cost, schedule and functionality should be combined with assessments from different stakeholders as customer, users, project managers, developers and management.

5.1 Future research

There is still a strong need for more research on the software effort estimation process, and how one should go about to improve the ability to produce accurate estimates.

There has been limited research on the role of the customer, and hopefully the results presented here will inspire future research on this topic.

Although the data in this study suggests that to use a checklist as an instrument both to gather and utilize experience from previous projects, more studies are needed on other forms of experience utilization. Research on how a checklist should be set up and developed is also needed.

An essential part of future software process improvement should be to get a better understanding of what constitutes project success. In order to carry out improvements one should know which goals one is trying to fulfil. Future research and project evaluations should strive to gather information from all stakeholders: clients, management, developers,
and project managers. In addition, estimation accuracy and delivered functionality should be measured. The different factors should be combined and the relationship between them investigated. This would lead to a better framework for both defining and determining project success.

The effects of different group techniques have on software effort estimation are also poorly understood. The review in this thesis presents different techniques that all should be studied more closely.
References


Appendix A

“The Relationship between Customer Collaboration and Software Project Overruns”

by Kjetil Moløkken-Østvold and Kristian Marius Furulund

Conditionally accepted for Agile 2007 in Washington DC
The Relationship between Customer Collaboration and Software Project Overruns

Conditionally accepted for Agile 2007 in Washington DC

Kjetil Molokken-Østvold¹ and Kristian Marius Furulund².

¹ Simula Research Laboratory, P.O.Box 134, 1325 Lysaker, Norway kjetilmo@simula.no
² Department of Informatics, University of Oslo, P.O.Box 1080 Blindern, 0316 Oslo, Norway kristff@ifi.uio.no

Abstract

Most agile projects rely heavily on good collaboration with the customer in order to achieve project goals and avoid overruns. However, the role of the customer in software projects is not fully understood. Often, successful projects are attributed to developer competence, while unsuccessful projects are attributed to customer incompetence. A study was conducted on eighteen of the latest projects of a software contractor. Quantitative project data was collected, and project managers interviewed, on several issues related to estimates, key project properties, and project outcome. It was found that in projects where collaboration was facilitated by daily communication between the contractor and the customer, they experienced a lesser magnitude of effort overruns. In addition, employing a contract that facilitates risk-sharing may also have a positive impact.

The Agile Manifesto stresses the importance of customer collaboration. An excerpt taken from http://www.agilemanifesto.org/ reads:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

The importance of customer collaboration in agile projects is further explored and explained in central textbooks, e.g., on XP [3], Scrum [4] and agile estimating and planning [5]. Mike Cohn elaborates: “Customer collaboration is valued over contract negotiation because agile teams would like all parties to the project to be working towards the same set of goals” [5]. Beck and Fowler state that a development project must adjust its course constantly, and further write that “Communicating is the point. We’ve seen too many requirement documents written down that don’t involve communication” [3]. Frequent communication can be used to prioritize features, set focus on bug-fixing or include more functionality [3].

However, what constitutes good collaboration, how can it be achieved, and can it help to reduce overruns? This paper explores the possible relationship between one of the key agile principles, customer collaboration, and software project overruns. Section 2 provides an account of the previous research that motivated the research questions, which are presented in Section 3. Section 4 describes the methods used to collect and analyze the data, which is presented in Section 5. Section 6 discusses the data in the light of the research questions. Section 7 concludes.

1. Introduction

Project overruns pose a major and persistent challenge for software development [1]. However, a recent study found that projects that used flexible development processes (agile, evolutionary and/or incremental) had an average magnitude of overruns that was significantly less than for projects that employed sequential (waterfall) development processes [2]. The median effort overrun was 1% for the projects that used a flexible process, compared to 60% for those that used a sequential process. The projects were similar in other key respects, including estimation approach, project size and proportion of delivered functionality. In-depth interviews found that, from the perspective of the project managers, flexible development processes fostered good collaboration with the customer to a greater extent than sequential processes [2].

2. Background

Recommending a high degree of customer involvement and collaboration is nothing new. In his
book on Scrum [4], Ken Schwaber presents his experiences from the 1960s in the chapter about the role of the product owner: “When I started developing software, customer involvement and collaboration weren’t problems. ... I used short iterations of one or two days. I’d meet with the customer, and we’d sketch out on paper what he or she wanted. We’d discuss the problem until I understood it. ... We didn’t realize it at the time, but this was heaven.”

In a textbook from 1976, Tom Gilb, who introduced evolutionary development, stated 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...” [6].

The same year, Harlan D. Mills wrote 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” [7].

Alistair Cockburn has presented a case for seeing software development as a series of resource-limited, goal-directed cooperative games of invention and communication [8]. His proposal was a reaction to both the existing view, and the failure to find a correlation between project success and the use of tidy engineering practices. He stresses that in a resource-limited game, communication, whether within the team, or between the team and users, is paramount. Rapid feedback is deemed to be particularly important. He summarizes the issues known to be important to project success within his model as: cooperation, communication, cost-of-, rate-of-, and sufficiency in-communication.

Ken Schwaber also presents his view on how collaboration and communication was hampered by the introduction of more process [4], which widened the gap between developers and stakeholders. From his perspective, the waterfall methodology, as it came to be commonly applied, embodied all the flaws of sequential development.

When it comes to the research community, the impact of poor collaboration between contractor/developer and customer on software overruns has not been studied to any great extent. The study described briefly in Section 1 found that projects that used flexible development processes had an average magnitude of overruns that was significantly less than for projects that employed sequential development processes, and that customer collaboration was important [2]. In addition, other studies have found that from the perspective of software professionals, the behavior of customers affects the accuracy of software estimates.

A systematic review of previous studies found that customer characteristics were often mentioned as both preventing and facilitating overruns [9]. None of the studies explicitly investigated the influence of customers on estimation accuracy, but seven out of eight studies reported that customer characteristics are perceived as important for estimation accuracy. Some of the reviewed studies emphasized the impact of project flexibility and communication [10], some involvement and commitment [11], and others communication and understanding [12]. In addition, it was found that it is easy for managers on the developer’s side, to attribute failure to the customer, and correspondingly attribute success to themselves [9].

However, the findings from previous studies are diverging and incomplete, and the area is far from understood. The differences can probably be explained by variance in purpose of the studies, terminology ambiguousness, various roles of the respondents, the method of analysis and the differences in sample sizes.

To follow up on the review of previous studies on the customer role, a study with 307 Norwegian software professionals was performed, in which the perceptions of the professionals were gathered by questionnaire [9]. The three customer-related reasons most frequently perceived as contributing to project overruns were 1) frequent requirement changes and new requirements, 2) lack of well defined requirements and 3) lack of competent customers able to make decisions. The most important reasons perceived as preventing project overruns were 1) competent customers able to make decisions, 2) well defined requirements and 3) adequate project administration [9].

Changed and new requirements were perceived to be the customer’s most frequent contribution to overruns. Here, it is interesting to note the agile community’s attitude towards welcoming change, as opposed to treating changes as unwelcome.

Availability of competent customers and capable decision makers was found to be the most important success-factor [9]. However, terms such as customer competence and customer decision-making ability are ambiguous.

The respondents also reported the estimation accuracy of their last completed project, and the projects with large overruns differed most from the projects that had smaller overruns with regard to 1) less realism in plans and budgets, 2) less project flexibility and 3) poorer customer and vendor communication.

3. Research Questions and Definitions

The Agile manifesto emphasizes “Customer collaboration over contract negotiation”. This, and research on the possible importance of customers and collaboration that was presented in Sections 1 and 2 motivated the research questions in this section.

It is virtually impossible to measure good collaboration objectively in an industrial setting.
Instead, we have to investigate key properties that are relevant for collaboration, and rely on manager responses. The study that found a relationship between flexible development processes and lesser overruns, also uncovered that the respondents felt that good collaboration and communication could help reduce overruns [2]. In a study by Verner et. al. that explored factors that contribute to the success and failure of projects [10], the managers who responded, reported good communication with customers as the foremost success property. However, communication and other related concepts are, to the same extent as is, collaboration, not precisely defined terms. Hence, motivated in part by Cockburn [8], we explore the frequency of communication between the contractor and the customer. This is a more objective measure and is also, we believe, representative of good collaboration.

RQ1: Do projects with daily contact between contractors and customers have a lesser magnitude of effort overruns than do other projects?

Beck and Fowler state that a fundamental problem with fixed-price contracts is that they pit the interests of supplier and customer against each other. It also appears as if many types of contracts used in the public sector has favored, or even required, a sequential (i.e. waterfall) approach. This has been previously observed both in Norway [13], the United States [14] and Australia [15], and many contractors have not been allowed to use more agile approaches. A consequence of this, as described in a recent paper by Jamieson, Vinsen and Callender [16] is that “…it is probable that a supplier will have accepted terms and conditions relating to scope and price that are philosophically in opposition to agile principles, particularly in respect of an iterative elaboration of requirements”. However, there appears to have been a recent change away from the strict requirements of a waterfall approach [15].

Concerns regarding estimation and contracts are frequently omitted in the agile literature. As described by Jamieson, Vinsen and Callender [15]: “Software can be developed in-house, but is more often obtained from vendors either as a package or through bespoke software development services” and “contemporary agile methods of software development do not appear to consider the role of the procurement process in influencing success, although the agile Unified Process (UP) recommends running projects in two contract phases, each of multiple timeboxed iterations.”

We are aware of several organizations in Norway who are proponents of introducing a risk-sharing mechanism into contracts. In addition a new contract standard aimed at agile projects and risk-sharing in the public sector is currently being developed. However, no one has explored the effects of different types of contracts on overruns and other related aspects. Hence, we investigated differences in overruns that are related to procurement and type of contract.

RQ2: Do projects that employ contracts facilitating risk-sharing have a lesser magnitude of effort overruns than do other projects?

Regarding type of contract, we differentiated among the following: 1) by the hour (time and material), 2) fixed price, 3) target price (risk sharing between contractor and customer), and 4) other.

A target price contract is a contract that shares the risk of overruns between the contractor and the customer. The contract defines a target of effort (in hours), and often a date, for completing the project. If the project is completed, and has used fewer hours than defined by the contract, the contractor and customer share the profit. If more effort than what is defined by the target is needed, the contractor and the customer share the extra costs. This is set up so that the customer pays a percentage, often 50%, of full price for the hours that make up the difference between the target and actual effort.

E.g., the contractor charges $150/hour and the target for completing the project is set to 1000 hours. If 800 hours is used to complete the project the customer has to pay full price for the 800 hours used and $75 for the remaining 200 hours (not used). If 1200 hours is used the customer pays full price for the 1000 hours set by the target, and $75 for the additional 200 hours.

A target contract can also be set up with floors and ceilings defining the minimum and maximum hours that the contractor could charge the customer. Related to collaboration, the customers procurement capability (frequently labeled maturity) have been said to have an impact on overruns [9]. This suggested impact has typically been found in responses from managers that tend to explain overruns by citing vague goals, lack of IT competence and poor management. This motivated the following research question:

RQ3: Do projects where the customer is perceived to have a good procurement capability have a lesser magnitude of effort overruns than do other projects?

The customer procurement capability we investigate is comprised of: a) their ability to collaborate, b) their IT skills, c) their decision-making ability and d) their clarity of goals.

In the software engineering literature, the term customer is often not clearly defined; sometimes it refers to a user, at other times to those who finance a project. A customer is also frequently called product.
owner by many. Since there is no uniform definition of customer; we apply a coarse-grained definition here: as whoever it is that engages the services of professional software engineers to produce a piece of software.

Different types of projects include:

1. Contracting – A case in which a software contractor develops custom-built solutions for an outside party. In this case the role of customer resides with the outside party, and can be assigned to one of the eventual users of the end-product.

2. In-house development - A case in which a software department typically develops solutions for another department in the same company. In this case, the role of customer resides in another department, and can be assigned to one of the eventual users of the product.

3. Shrink-wrap software - A case in which a software department of a company develops software for sale to businesses and/or individuals. The role of customer is often assigned to a business manager.

This is not intended to be an exhaustive overview, but represent most typical cases. In the study reported herein, we explored projects in category 1.

4. Method

The study was conducted in Norway from March 14th to October 16th 2006.

4.1. The company studied

The company studied is a medium-sized Norwegian software consultancy that, at the time of the study, had about 300 employees. The company operates as an independent contractor and offers a wide range of complete software solutions to its various customers.

In order to account for limitations in previous surveys [2], we wanted to explore several projects within one company to better isolate effects such as communication and customer competence.

4.2. Data collection and analysis

We interviewed the project managers of 18 different projects. These projects were selected by the company, which was not informed of our research questions. The inclusion criteria were as follows (i) relevant project data was stored and available, (ii) we were able to interview the responsible project managers, and (iii) the projects had a workload of at least 100 man-hours. This last criterion is in line with previous surveys, in which “trivial tasks routinely handled without effort estimation” were also filtered out [17].

We collected data via personal interviews, which yields data of high quality and ensures that ambiguities are resolved [18]. 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 personal involvement on the part of the researchers 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 projects as would be possible by using mailed questionnaires.

Each interview lasted between 20 and 80 minutes. The respondents were informed that their responses were anonymous, and that no feedback about the respondents’ answers would be reported to outsiders or to company managers.

The following factors are explored in this paper: 1) communication frequency, 2) contract type, and 3) customer procurement capability. The measures of 3 are based on the subjective opinions of the project managers, while measures of 1 and 2 are more objective measures.

In addition, several questions were aimed at general project properties. In addition to having their own value, these questions are also relevant for isolating effects and exploring confounding factors in the study. These possible confounding factors included project size, perceived project complexity and technical knowledge.

When conducting research on the accuracy of estimates of software projects, it is necessary to differentiate between dissimilar types of estimates. What estimate(s) to use depends on the focus of the research. If the goal is to compare the actual effort with the estimated effort, as it is in this paper, it is meaningful to use the most likely estimates at the planning stage, instead of, for example, project bids. The latter may be affected by outside factors, such as market competition.

In order to compare actual effort and estimated effort, and measure any differences in project overruns dependent on the studied properties, we used the $BRE_{bias}$ measure, previously used in related research, e.g., [2, 19]. It is calculated as:

$$BRE_{bias} = \frac{x - y}{\min(x, y)},$$

where $x$ = actual and $y$ = estimated.

The $BRE_{bias}$ measures both the magnitude and direction of effect when comparing the actual effort to estimated effort. $BRE_{bias}$ is based on the Balanced Relative Error (BRE) [20, 21].

Even though MRE has been the most widely used measure of estimation accuracy [22], one must be aware that it has undesirable properties [20, 23].
The main concern for our case is the fact that underestimated and overestimated projects are weighted unevenly.

To measure the size of any difference, we used Cohen’s size of effect measure (d) [24], where:

\[
d = \frac{\text{sample}_1 - \text{sample}_2}{\text{pooled StdDev}}
\]

We include the representation of the data in tables and figures, and provide statistical Kruskal-Wallis test for difference where those are appropriate. The Kruskal-Wallis is more robust for unequal sample sizes [25], as was the case in this study.

5. Results

Out of the 18 completed projects, three were overestimated, one was on target, while 14 were underestimated. A full account of key project data is provided in Appendix 1.

The mean and median BREbias were 0.27 and 0.22, respectively (this corresponds to an effort overrun of 27% and 22%). This is in line with findings in previous surveys and case studies on software estimation [26], and indicates that in this respect, the projects studied were fairly representative.

5.1. Communication frequency

The respondents stated how often they communicated with the customer. In this analysis, we have differentiated between those who had daily communication (11 projects), and those who did not have daily communication (seven projects). The results are displayed in Figure 1 and Table 1.

The projects in which the developers did not have daily communication with the customer appear to have a larger magnitude of effort overruns. A Kruskal-Wallis test for difference results in \( p = 0.023 \). The corresponding size of effect is \( d = 1.25 \), indicating a large size of effect [24].

5.2. Type of contract

The following types of contracts were employed: 1) by the hour (time and material), 2) fixed price, 3) target price (risk sharing between contractor and customer), and 4) other.

The results are displayed in Figure 2 and Table 2.

5.3. Customer procurement capability

The managers were asked to rate the customer on different procurement capabilities, frequently presented as relevant in previous studies [9]. The managers rated the customer on a five-point Likert-
scale (1= very good, 3= average, 5=very poor) on four different properties: collaboration skill (CO), IT-competence (IT), decision making ability (Dec.) and clarity of project goals (Goals).

In general, the managers gave their customers a high rating on most factors; mean scores were all above the neutral level. The scores were summarized in order to calculate a total customer score for each project. The best possible score was 4 (1 on all four properties), while the lowest possible was 20 (5 on all four properties). The projects received total scores from 4 to 15, with mean and median results for all projects calculated to 8.7 and 8 respectively. The main results are presented in Table 3.

Table 3: Customer capability as rated by the managers

<table>
<thead>
<tr>
<th></th>
<th>Co</th>
<th>IT</th>
<th>Dec.</th>
<th>Goals</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>1.8</td>
<td>2.8</td>
<td>2.2</td>
<td>1.9</td>
<td>8.7</td>
</tr>
<tr>
<td>Median</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>

We differentiated between those who received a total score of 8 or better (rated as good or better), and those with a total score of 9 or worse. See Table 4 and Figure 3 for an overview.

There was no apparent, large difference when comparing the total customer score. A Kruskal-Wallis test resulted in \( p=0.35 \), Cohen’s \( d=0.21 \). However, there is a slight tendency towards a lesser magnitude of effort overruns when the customers are perceived as being more capable.

5.4. Possible confounding factors

In observational studies like this, where one does not have the same degree of control over variables as in an experiment, it is important to measure and control for possible confounding factors. These factors include differences in underlying project properties, and it is necessary to explore how they interact with the factors that are investigated. Typical project properties, as described in a framework for analyzing software overruns [27], that may have an impact on overruns include project size, project complexity, and familiarity with technology.

Regarding the effect of project size on overruns, we divided the projects into two equal samples, with a “small” and “big” group, based on actual effort. A representation of BREbias based on project size (measured in actual effort) is presented in Figure 4 and Table 5.

<table>
<thead>
<tr>
<th>Size</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Big</td>
<td>9</td>
<td>0.39</td>
<td>0.21</td>
</tr>
<tr>
<td>Small</td>
<td>9</td>
<td>0.14</td>
<td>0.22</td>
</tr>
</tbody>
</table>

As seen in Figure 4 and Table 5, the median values indicate that the typical case in both groups has a similar overrun. However, there are substantial differences in mean values. Taken together, this indicates a tendency for relatively large projects to have some incidents with a large magnitude of overrun.

Complexity is a more subjective measure. The respondents rated the project complexity on a scale of: high, medium and low. No projects received a “low” score. BREbias based on complexity is presented in Figure 5 and Table 6.
There appear to be smaller overruns in the projects where the respondents had good technical knowledge. It would seem that relatively large projects in which good technological familiarity is absent may be related to larger overruns. It is therefore necessary to explore these factors further in the discussion.

6. Discussion

In a coarse-grained categorization, we can say that overruns in software projects can be caused by characteristics related to the following factors:
1. Development organization properties
2. Customer properties
3. Properties that depend on the interaction between the developing organization and customers

Most previous research on estimation has focused on category 1. Here, we have tried to explore issues related to category 2 and 3.

6.1. Communication frequency

Daily frequency of communication was associated with fairly small overruns in our projects, and the difference from the projects that did not have daily communication has to be considered large and significant. However, there may be confounding factors that influenced these results.

Project size, measured in actual effort, based on the communication frequency, is presented in Table 8.

<table>
<thead>
<tr>
<th>Level</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daily</td>
<td>11</td>
<td>6623</td>
<td>2700</td>
</tr>
<tr>
<td>Not daily</td>
<td>7</td>
<td>3966</td>
<td>1830</td>
</tr>
</tbody>
</table>

The table shows that project size was larger for the group with daily communication with the customer.

Technical knowledge based on the communication frequency is presented in Table 9.

<table>
<thead>
<tr>
<th>Level</th>
<th>N</th>
<th>Good</th>
<th>OK</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daily</td>
<td>11</td>
<td>7</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Not daily</td>
<td>7</td>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Sum</td>
<td>18</td>
<td>9</td>
<td>8</td>
<td>1</td>
</tr>
</tbody>
</table>

There is an overrepresentation of favorable technical knowledge in the projects with daily communication with the customer. This implies that
some of the estimation accuracy gains in projects in which developers and customer communicated daily could actually be attributed to the technical knowledge, and not to the frequency of communication. However, the possible advantage of better technical knowledge is offset by the larger size of the projects in which developers and customer communicated daily.

However, frequent communication may only be a symptom of, and not a reason for, a good and trusting relationship. On the flip side, one may also surmise that in a project that has gone off-course, the customers would communicate frequently with the contractors. Nonetheless, this provides some indications that frequent communication with the customer may help to avoid overruns.

Good collaboration/communication is a frequent top-listed property that managers point to when explaining project-success, e.g. [2, 9]. It is likely that frequent communication helps to build trust and resolve differences, and helps the developers to focus on solving the customers’ problems, instead of wasting effort on developing functionality that is neither required nor correct. This should apply whether the customer is in-house or external.

### 6.2. Type of contract

There were three different main types of contracts in our sample, and thus it is difficult to draw any clear conclusions. Some key points were that, not surprisingly, contracts that were paid for by the hour had the largest overruns. This type of contract may often add incentives for gold-plating by the contractor, frequently in the form of unnecessary functionality for the customers.

Fixed-price contracts had smaller overruns, probably because overruns are paid for by the contractor.

On the other hand, target-price contracts, which involves risk-sharing, and technically should foster collaboration, fared even better in terms of small overruns.

However, confounding factors may have an impact on this issue. Project size, measured in actual effort, based on contract form, is displayed in Table 10.

#### Table 10: Actual effort grouped by contract form

<table>
<thead>
<tr>
<th>Level</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>By the hour</td>
<td>4</td>
<td>2693</td>
<td>2277</td>
</tr>
<tr>
<td>Fixed price</td>
<td>5</td>
<td>11235</td>
<td>12500</td>
</tr>
<tr>
<td>Target price</td>
<td>7</td>
<td>4288</td>
<td>1830</td>
</tr>
<tr>
<td>Other</td>
<td>2</td>
<td>1826</td>
<td>1826</td>
</tr>
</tbody>
</table>

The table shows a substantial variance in size for the different contract forms. It may be particularly relevant to note the difference in size between fixed-price and target-price contracts.

Technical knowledge based on the type of contract is displayed in Table 11.

#### Table 11: Technical knowledge grouped by contract form

<table>
<thead>
<tr>
<th>Level</th>
<th>N</th>
<th>Good</th>
<th>Ok</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>By the hour</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Fixed price</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Other</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Target price</td>
<td>7</td>
<td>5</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Sum</td>
<td>18</td>
<td>11</td>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>

The table shows a slight variance in technical knowledge for the different types of contract. Again, it may be especially relevant to note the difference in technical knowledge between fixed-price and target-price contracts.

It might be that contractors seek to use a target price only when they believe that they have the technical knowledge. At the same time, customers may seek a fixed-price contract in order to free themselves of risk in large, complex projects.

In recent years, risk-sharing contracts have been increasingly popular in Norway. However, we are not aware of any previous studies that systematically evaluate their possible impact on overruns and project success. This, together with our small sample and possible confounding factors, makes these results tentative and so they should only be applied as a stepping stone for further research.

### 6.3. Customer procurement capability

We did not find any large differences in effort overruns based on customer procurement capability regarding different factors. This may be due to the following:

1. Interviews are not an accurate instrument for measuring such properties, e.g., the managers are not able to assess customer IT skill accurately.
2. Over a certain threshold, customer competence only has a small effect, as we observed. Only in clear cases of incompetence do customers severely affect overruns. Overall, the customers in this sample received favorable ratings.
3. These factors are, in general, not important for project execution.

Previously, unsuccessful projects have frequently been attributed to customer incompetence, while successful projects have been attributed to good performance by the contractors [9].

This study paint a somewhat different picture, and it appears, as previously found [28], that who
and how you ask affects the responses you get. When asked to explain overruns in free-test responses, it might just be that respondents are too eager to attribute them to customers [9].

6.4. Threats to validity

In general, the sample size of this study is too small for statistical analysis, and should be used with caution. As in a previous related study [2], it was a priority to collect in-depth, high-quality data on a wide range of factors, instead of using mailed questionnaires. The work presented herein must be seen as a starting point for future research, and also as a framework for how different companies can analyze the performance of their projects.

Regarding internal validity, one must also be aware that some of the measures, e.g. customer competence, are subjective perceptions of the respondents, and not objective facts. This has been mitigated by having respondents rate their customers on a predefined scale.

In a study like this, which is not a controlled experiment, relationships of cause and effect are impossible to pinpoint. In addition, in accordance with the guidelines presented in [27], one must be aware of possible confounding factors. Nonetheless, if the results from the objectively gathered data and previous studies are combined, tendencies can be seen.

Regarding external validity, the size of the overruns is similar to those in previous surveys and case studies, which indicates that the sample was not particularly biased. However, in other environments, such as in-house development, other factors need to be taken into account. Therefore, the external validity of the study is limited to contractors developing projects of a similar size and complexity.

7. Summary

The main finding was that good collaboration with customers, facilitated by frequent communication, was associated with projects that experienced a lesser magnitude of effort overruns. It might also be that risk-sharing contracts can reduce overruns, but one must be aware of possible confounding factors with respect to this issue.

In addition, it was not observed a clear relationship between customer procurement capability and magnitude of overruns. This might be due to the small sample size, but it may also be that the results constitute evidence that this frequent explanation of overruns is somewhat exaggerated.

Taken together, the findings presented in this paper indicate that contractors can implement a few key practices to facilitate collaboration with their customers in order to reduce overruns and achieve greater project success.

These results should, perhaps, not be surprising to many in the agile community, given the commitment to customer collaboration over contract negotiation.

In addition to what has been observed related to reducing overruns, risk-sharing contracts and frequent communication may also have benefits that extend beyond the current project. These benefits include fostering a contractor/customer relationship that is beneficial in the long-run and that will bring returning business.

Future research should address the need for an extended set of project data for analysis, and should further seek to explore subtopics, such as different types and lengths of communication between customers and contractors, and various ways to introduce risk-sharing contracts.

Acknowledgements

This research was funded by the Research Council of Norway under the project INCO. We thank the studied company for providing us with data and access to their managers. Also thanks to Mike Cohn and Chris Wright for valuable comments.

References


## Appendix 1: Key project data

<table>
<thead>
<tr>
<th>ID</th>
<th>Est. effort</th>
<th>Actual effort</th>
<th>BREBias</th>
<th>Communication frequency</th>
<th>Contract form</th>
<th>Customer score</th>
<th>Project complexity</th>
<th>Tech. knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5660</td>
<td>7301</td>
<td>0.290</td>
<td>Not daily</td>
<td>Target price</td>
<td>13</td>
<td>Medium</td>
<td>Good</td>
</tr>
<tr>
<td>2</td>
<td>1810</td>
<td>1570.5</td>
<td>-0.152</td>
<td>Daily</td>
<td>Target price</td>
<td>4</td>
<td>Medium</td>
<td>Good</td>
</tr>
<tr>
<td>3</td>
<td>12000</td>
<td>14500</td>
<td>0.208</td>
<td>Daily</td>
<td>Target price</td>
<td>12</td>
<td>High</td>
<td>Good</td>
</tr>
<tr>
<td>4</td>
<td>990</td>
<td>1252</td>
<td>0.265</td>
<td>Not daily</td>
<td>Other</td>
<td>8</td>
<td>Medium</td>
<td>Bad</td>
</tr>
<tr>
<td>5</td>
<td>9900</td>
<td>12600</td>
<td>0.273</td>
<td>Daily</td>
<td>Fixed price</td>
<td>10</td>
<td>High</td>
<td>OK</td>
</tr>
<tr>
<td>6</td>
<td>5600</td>
<td>11000</td>
<td>0.964</td>
<td>Not daily</td>
<td>Fixed price</td>
<td>12</td>
<td>Medium</td>
<td>OK</td>
</tr>
<tr>
<td>7</td>
<td>2400</td>
<td>2400</td>
<td>0.000</td>
<td>Daily</td>
<td>Other</td>
<td>13</td>
<td>Medium</td>
<td>Good</td>
</tr>
<tr>
<td>8</td>
<td>12000</td>
<td>12500</td>
<td>0.042</td>
<td>Daily</td>
<td>Fixed price</td>
<td>7</td>
<td>High</td>
<td>Good</td>
</tr>
<tr>
<td>9</td>
<td>3430</td>
<td>4076</td>
<td>0.188</td>
<td>Daily</td>
<td>Fixed price</td>
<td>11</td>
<td>Medium</td>
<td>OK</td>
</tr>
<tr>
<td>10</td>
<td>3135</td>
<td>2507</td>
<td>-0.250</td>
<td>Daily</td>
<td>By the hour</td>
<td>5</td>
<td>Medium</td>
<td>Good</td>
</tr>
<tr>
<td>11</td>
<td>700</td>
<td>947</td>
<td>0.353</td>
<td>Not daily</td>
<td>Target price</td>
<td>11</td>
<td>Medium</td>
<td>Good</td>
</tr>
<tr>
<td>12</td>
<td>13500</td>
<td>16000</td>
<td>0.185</td>
<td>Daily</td>
<td>Fixed price</td>
<td>9</td>
<td>High</td>
<td>Good</td>
</tr>
<tr>
<td>13</td>
<td>1582</td>
<td>4268</td>
<td>1.698</td>
<td>Not daily</td>
<td>By the hour</td>
<td>7</td>
<td>Medium</td>
<td>OK</td>
</tr>
<tr>
<td>14</td>
<td>953</td>
<td>1164</td>
<td>0.221</td>
<td>Not daily</td>
<td>Target price</td>
<td>7</td>
<td>Medium</td>
<td>OK</td>
</tr>
<tr>
<td>15</td>
<td>1695</td>
<td>1830</td>
<td>0.080</td>
<td>Not daily</td>
<td>Target price</td>
<td>7</td>
<td>Medium</td>
<td>OK</td>
</tr>
<tr>
<td>16</td>
<td>1400</td>
<td>1950</td>
<td>0.393</td>
<td>Daily</td>
<td>By the hour</td>
<td>7</td>
<td>High</td>
<td>OK</td>
</tr>
<tr>
<td>17</td>
<td>3500</td>
<td>2700</td>
<td>-0.296</td>
<td>Daily</td>
<td>Target price</td>
<td>8</td>
<td>Medium</td>
<td>Good</td>
</tr>
<tr>
<td>18</td>
<td>1511</td>
<td>2047</td>
<td>0.355</td>
<td>Daily</td>
<td>By the hour</td>
<td>6</td>
<td>Medium</td>
<td>OK</td>
</tr>
</tbody>
</table>
Appendix B

“The Impact of Utilizing Experience on Software Effort Estimation Accuracy
An Empirical Study”

by Kristian Marius Furulund and Kjetil Moløkken-Østvold

To be submitted to a Conference
The Impact of Utilizing Experience on Software Effort Estimation Accuracy—An Empirical Study

To be submitted to a Conference

Kristian Marius Furulund and Kjetil Moløkken-Østvold

1 Department of Informatics, P.O.Box 1080 Blindern, 0316 Oslo, Norway
kristf@ifi.uio.no

2 Simula Research Laboratory, P.O.Box 134, 1325 Lysaker, Norway
kjetilmo@simula.no

Abstract

The use of experience data is frequently suggested as a mean to increase software effort estimation accuracy. However, there has been done limited empirical research on the subject. To investigate the effect of using experience data, we conducted a study on eighteen of the latest projects of a software contractor. Quantitative project data was collected, and project managers interviewed, on several issues related to estimates, key project properties, and project outcome. It was found that in projects where experience data was utilized in the estimation process, they experienced a lesser magnitude of effort overruns. The use of a checklist also increased estimation accuracy. However, the utilization of an estimation model in the estimation process appeared not to have any impact on the estimation accuracy. It appears as if utilizing and gathering experience data through the use of a checklist is a good way to improve estimation accuracy.

1 Introduction

Project overrun is a major and persistent challenge in software development [1]. Estimation is an important part of all software engineering projects, and the ability to produce accurate estimates has an impact on key economic processes, including budgeting and bid proposals. Inaccurate estimates may result in the wrong projects being selected. Projects estimated optimistically might be selected instead of a project that has been estimated pessimistically [2-4]. Accurate estimates have been found to be a major factor contributing to project success [5, 6].

Research has shown that the average effort overrun in software development projects is about 30%-40% [1]. Such large deviations from the estimates indicate that there is room for improvement in the way effort is estimated.

It is not easy to decide how to approach the task of software effort estimation. Throughout the (albeit short) history of software development, new estimating techniques have emerged continually. Attempts have been made to compare these techniques and derive best practices [2]. However, there seems to be little consensus on which elements and factors increase estimation accuracy.

In this paper, the focus is on factors internal to the project that are related to the estimation process. In addition to investigating the effect of using data from experience, we will look at utilizing the experience by using estimation models and checklists. These are simple estimation properties that, both intuitively and through research, have been seen as having a positive impact on estimation accuracy [7, 8].

Section 2 presents the effects of utilizing experience data for process improvement in general, and for obtaining more accurate effort estimates in particular. The use of models and checklists is also introduced. In Section 3, the research questions are defined. Section 4 presents the method for data collection. The results are presented in Section 5. Section 6 discusses the results on the basis of the background provided in Section 2. Section 7 summarizes and concludes.

2 Background

A recent study found that managers regard overlooked tasks and unexpected events as the most dominant causes of estimation inaccuracy [9]. The same study found that a lack of systematic feedback to enable learning was a severe cause of estimation inaccuracy. In a study by Lederer and Prasad [10], overlooked tasks was named as the third most common cause of estimation inaccuracy. A lack of method and guidelines for carrying out the estimation process, and lack of historical data, were also among the top ten reasons cited. Lederer and Prasad conclude that the method for estimating have a significant impact on estimation accuracy [10].

2.1 The use of experience data and estimation by analogy

Gathering and utilizing data from experience is important for all attempts to improve estimation processes.
In 1911, Fredrick W. Taylor presented his theories for scientific management. Despite their many faults, the theories contain many ideas that retain their value for improving the efficiency of processes. Taylor was the first to focus on the value of gathering data from the current situation, so that that data could be utilized for improving a process further [11].

W. E. Deming also had a strong focus on data collection and statistical analyses, and he has played an important role in the development of frameworks for improving processes. Deming introduced the term “statistical process control”, where he emphasized the need to collect data and to learn from experience. In order to learn from experience regarding a particular process, it is necessary to have some sort of a structure that ensures that the process under investigation is repeated fairly consistently. Deming argued that (i) by using data from experience, it is possible to see where there is potential for improvement, and (ii) by comparing results before and after a new measure has been implemented in a test case, it is possible to determine whether or not it is a good idea to introduce the measure on a more long-term basis [12].

Collecting data from experience and utilizing it to determine which measures to take to improve the process of software development is one of the basic tenets in the well-known process improvement framework CMMI [13].

While considering how software developers could improve their performance, Watts Humphrey underlines the need for data collection and feedback [14]. He emphasizes that producing better outcomes and learning for the future is difficult if data from experience is not gathered and utilized systematically.

The idea of recording and utilizing data from experience when estimating software development effort is not new. It was presented as early as 1967 by Nelson and Force. [15].

More than a decade later, in his book “Software Engineering Economics” Barry Boehm presented estimation by analogy as an estimation technique. The central idea of this technique is to base the estimates for a new project on data from completed projects [16].

The strength of this approach is that estimates are based on actual experience [17]. The problem is the often very unique nature of software development projects, which makes it difficult to assess how similar a new project is to a previous one. Estimation by analogy is, or at least has been, widely utilized in the software industry [18]. Findings regarding its utility are mixed. Some studies find evidence to support its use [17, 19], while others find that its use does not lead to more accurate estimates [20].

Jørgensen, Indahl et al. provide this definition of estimation by analogy [21]:

“Estimation by analogy is (...) the process of finding one or more projects that are similar to the one to be estimated and then deriving the estimate from the values of these projects.”

They represent a very inclusive view of what could be considered estimation by analogy, incorporating all of the following: (i) pure expert estimation (the “database” of previous projects is in the expert’s head), (ii) expert estimation informally supported by a database containing information about finished projects (iii) and estimation based on the use of a clustering algorithm to find similar projects [21].

There exist different views on how to utilize data from experience when deriving effort estimates, which fall into two basic categories: computing new estimates on the basis of data from experience, or deriving them by assessing the data subjectively and using it as a mental input to the estimation process.

Shepperd, Schofield et al. emphasise the need to compute new estimates based on previous ones. They provide a tool (ANGEL) to aid what they see as a complex, but necessary computational process [19].

Cowderoy and Jenkins view estimation by analogy as too casual to be the basis of good project manager decisions. They call for estimates based on the output from different estimation techniques [22].

Jørgensen, Indahl et al. found that estimation by analogy improves estimation accuracy if combined with the regression towards mean-technique. They conclude that there is need for some mathematical support in deriving estimates based on experience data from previous projects [21].

Ohlsson, Wohlin et al. found that utilizing experience data does not necessarily lead to more accurate estimates. The study found that experience data is best utilized if it is used together with human assessment [23].

Walkerden and Jefferey underline the need for human consideration of the recorded data as an important part of a sound strategy for selecting analogues. They note that such a strategy is more flexible than using more mathematical methods [17].

It is necessary to note that there is more to data from experience than just numeric data on actual effort. All experience gathered, recorded or not and numeric or not, might be utilized in future estimation processes.

When studying the use of data from experience to enable more accurate software effort estimation, it is interesting to consider two related, and frequently used, tools for improving estimation accuracy.

1. Estimation models
   A thorough and systematic way of utilizing and building experience that leaves reduced opportunity for human input. For a precise definition, see section 3.1

2. Checklists
   A simple, but limited way to utilize and build experience that, to a large extent, enables flexibility and human input.

Estimation models and checklists are two specific tools that could be used in the estimation process. Both estimation models and checklists could be customized on the basis of data from experience. However, it is also possible to utilize these tools without the use of
data from experience. It is also important to note that there are several settings in which data from experience could be applied and neither an estimation model nor a checklist is used.

2.2 Estimation Models

Most of the estimation done today is expert-based [1]. In [24], Jørgensen raises the issue of why estimation models are not applied by project managers more frequently. He argues that the lack of evidence for their efficacy may be the most significant reason.

In [14] Humphrey presents another view: “The absence of (...) a feedback cycle is a fundamental reason for the limited use of many software engineering methods [14].”

Recently, a thorough survey was conducted on studies that compared models to expert-based estimates. It was concluded that there was no substantial evidence to suggest that expert-based estimation should be replaced by models. On the other hand, the survey did not show that expert-based estimation is significantly better than using models [24]. Indeed, one only needs to look at the recurring problem of significant project overrun to see that expert-based approaches are flawed.

2.3 Checklists

A checklist is a simple way of utilizing experience, and it has been advocated by many as a good way to improve expert-based estimation processes. A checklist could help the estimators to avoid overlooking tasks, which phenomenon has been cited as a major reason for estimation inaccuracy [2, 7, 25].

In laying out principles for better effort estimation, Passing and Strahringer argue for the use of a transparent and repeatable procedure when doing expert-based estimation [25]. A study showed that the use of checklists had a positive impact on estimation accuracy. The use of a checklist reduced the estimators’ over-optimism and increased estimate transparency and consistency [8].

The need for more research on the use of checklists and the effect that such use has on estimation accuracy has been emphasized in several previous studies [7-9].

3 Definitions and research question

3.1 Definitions

When conducting research on the accuracy of estimates of software projects, it is necessary to differentiate between different types of estimate. What estimate(s) to use depends on the focus of the research. If the goal is to compare the actual effort with the estimated effort, as it is in this paper, it is meaningful to use the most likely estimates at the planning stage, instead of, for example, project bids. The latter may be affected by outside factors, such as market competition.

We understand “data from experience” to mean recorded data from previous projects that has been formally or informally systemized and made available for utilization in future estimation processes. By “checklist”, we mean a list that helps the estimator to remember tasks and other factors to be considered when doing effort estimation.

By “estimation model” we mean a model that, on the basis of different input variables describing a task, computes an estimate for the size of, or time or effort needed to complete, the task.

When differentiating between model-based, expert-based, and a combination of the two methods, we apply the definitions presented in a recent paper. “Judgment-based effort estimates (are) based on a tacit (intuition-based) quantification step and model-based effort estimates (are) based on a deliberate (mechanical) quantification step[24].”

3.2 Research question

The study reported herein investigated the effect of utilizing data from experience in the estimation process. It also investigated the possible effect that utilizing an estimation model and a checklist have on effort estimation accuracy. Models and checklists could be used as means for implementing experience gathering and utilization, and it is therefore also interesting to compare the two approaches.

RQ1: Do projects in which data from experience is utilized have a lesser magnitude of effort overrun than do other projects?

RQ2: Do projects in which an estimation model is utilized experience a lesser magnitude of effort overrun than do other projects?

RQ3: Do projects in which checklists are utilized experience a lesser magnitude of effort overrun than do other projects?

4 Method

The study was conducted in Norway from March 14th to October 16th 2006.

4.1 The company studied

The company studied is a medium-sized Norwegian software consultancy, which at the time of the study had about 300 employees. The company operates as an independent contractor and offers a wide range of complete software solutions to its various customers.

In order to account for limitations in previous surveys [26], we wanted to explore several projects within one company, in order to isolate effects better.

4.2 Data collection and analysis

We interviewed the project managers of 18 different projects. These projects were selected by the company,
which was not informed of our research questions. The inclusion criteria were that relevant project data was stored and available, that we had access to interview the responsible project managers, and that the projects had a workload of at least 100 man-hours. This last criterion is in line with previous surveys, in which “trivial tasks routinely handled without effort estimation” were also filtered out [3].

We collected data via personal interviews, which yields data of high quality and ensures that ambiguities are resolved [27]. It also allows for the respondents to add valuable information that it is not possible to include when completing a predefined questionnaire. Another point in favour 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; hence, we were prevented from investigating as many projects as would be possible by using mailed questionnaires.

Each interview lasted between 20 and 80 minutes. 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.

The following are explored herein: 1) the utilization of experience data in the estimation process, 2) the use of an estimation model to derive the estimates, 3) the use of checklists in the estimation process, 4) the managers’ perceptions of causes of estimation inaccuracy, and 5) the managers’ perceptions of what would be the ideal estimation process. The measures of 4 and 5 are based on the subjective opinions of the project managers, while measures of 1, 2 and 3 are more objective measures.

In addition, several questions were aimed at investigating general properties of projects. In addition to having their own value, these questions are also relevant for isolating effects and exploring confounding factors in the study [28]. These possible confounding factors included project size, contract form and perceived technical knowledge.

In order to compare actual effort and estimated effort, and to measure any differences in project overruns that depended on the studied properties, we used the BREbias measure, which has been used previously in related research, e.g., [26, 29]. It is calculated as:

\[
BREbias = \frac{(x - y)}{\min(x, y)},
\]

\[x = \text{actual and } y = \text{estimated value.}\]

The BREbias measures both the magnitude and direction of effect when comparing the actual effort to estimated effort. BREbias is based on the Balanced Relative Error (BRE) [30, 31].

Even though MRE has been the most widely used measure of estimation accuracy [32], one must be aware that it has unfortunate properties [30, 33]. The main concern for our case is the fact that underestimated and overestimated projects are weighted unevenly.

To measure the size of any difference, we used Cohen’s size of effect measure (d) [34], where

\[d = \frac{\text{sample1} - \text{sample2}}{\text{pooledStdDev}}\]

We include the representation of the data in tables and figures, and use a statistical Kruskal-Wallis test for difference. The Kruskal-Wallis test is a non-parametric method for testing equality of population medians. The test does not assume a normal population, and the groups do not have to be equal in size [35]. A p-value from a Kruskal-Wallis test below the significance level indicates a difference between the compared populations that is not random. One-sided T-tests have also been included [36]. In general, the sample size is small and the statistical results should be used with caution.

Additional data is provided so that readers can draw their own conclusions. A full account of project data is provided at folk.uio.no/kristf/APSEC2007/data.pdf.

5 Results

Of the 18 finished projects, three were overestimated, one was on target, while 14 were underestimated. The mean and median BREbias is 0.27 and 0.22 respectively (this corresponds to effort overruns of 27% and 27% respectively). This is in line with findings in previous surveys and case studies on software estimation, and indicates that with regards to overruns, the projects studied were fairly representative [1]. Key data is presented in Figure 1 and Table 1.

![Figure 1: BREbias for all the projects](image)

![Table 1: BREbias for all projects](table)
5.1 RQ1 – The use of experience data

The “yes” category includes those projects in which recorded data were utilized in the estimation process. The “no” category contains projects in which no experience data was used when deriving effort estimates.

Figure 2: BREbias based on experience data utilization

Table 2: BREbias based on experience data

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>6</td>
<td>0.06</td>
<td>0.13</td>
</tr>
<tr>
<td>No</td>
<td>12</td>
<td>0.37</td>
<td>0.27</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>K-W</th>
<th>T-test</th>
<th>Cohen’s d</th>
</tr>
</thead>
</table>
| Utilizing data from experience seems to lead to more accurate estimates. However, one has to be aware of the small sample size for these kinds of analyses. The d-value indicates that, according to Cohen’s effect classification, the application of data from experience has a large effect.

5.2 RQ2 - The utilization of estimation models

None of the projects relied entirely on a model when deriving the estimates. Most of those that used a combination of a model and experts utilized elements from a company customized model or a self-developed model. A similar distribution was found in [37].

Table 3: BREbias based on estimation model utilization

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model-based</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Combination</td>
<td>8</td>
<td>0.34</td>
<td>0.26</td>
</tr>
<tr>
<td>Expert-based</td>
<td>10</td>
<td>0.20</td>
<td>0.20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>K-W</th>
<th>T-test</th>
<th>Cohen’s d</th>
</tr>
</thead>
</table>
| The findings here show that the overruns are lower for those projects in which the estimating is expert-based than for those where a combination of expert-based and model-based methods is applied. These result are in agreement with the findings in [20, 24], but are contrary to those in [37] where it was found that a combination approach outperformed expert-based estimation.

5.3 RQ3 - The use of checklists

Checklists were used when estimating in seven projects and not used in 10. For one of the projects, the project managers were uncertain and did not answer the question on checklists.

Table 4: BREbias based on checklists

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>7</td>
<td>0.10</td>
<td>0.19</td>
</tr>
<tr>
<td>No</td>
<td>10</td>
<td>0.38</td>
<td>0.29</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>K-W</th>
<th>T-test</th>
<th>Cohen’s d</th>
</tr>
</thead>
</table>
| The use of checklists seems to have a positive impact on estimation accuracy. Both the mean and median are substantially lower for the group of projects where a checklist has been used in the estimation processes.

As with the use of data from experience, we have a relatively low p-value for the t-test and a Cohen’s d-value that indicates a large effect.

5.4 Project Manager Assessments

We also wanted to investigate the interviewed project managers’ attitudes towards key estimation factors. We asked them to cite reasons for estimation inaccuracy and describe how they saw the ideal estimation process.
5.4.1 Top reasons for estimation inaccuracy

The table shows the share of project managers that cited the different factors as a reason for estimation inaccuracy. Only the top five are presented.

Table 5: Reasons for estimation accuracy

<table>
<thead>
<tr>
<th>Reason</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Underestimated complexity</td>
<td>39%</td>
</tr>
<tr>
<td>Lack of technical skills</td>
<td>33%</td>
</tr>
<tr>
<td>Tasks overlooked</td>
<td>33%</td>
</tr>
<tr>
<td>Weak/ambiguous requirement</td>
<td>28%</td>
</tr>
<tr>
<td>New technology</td>
<td>28%</td>
</tr>
</tbody>
</table>

These findings are interesting, because they show that the project managers seem to agree with the notion that overlooked tasks are a major cause of estimation accuracy [9, 10]. However, tools such as checklists to help prevent overlooking tasks are rarely used.

5.4.2 Top factors in ideal estimation method

The table shows the share of project managers that named the different factors as part of what they see as the ideal estimation method. Only the top five are presented.

Table 6: Factors in ideal estimation method

<table>
<thead>
<tr>
<th>Factor</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer involvement?</td>
<td>53%</td>
</tr>
<tr>
<td>Use technical experts</td>
<td>33%</td>
</tr>
<tr>
<td>Build on experience</td>
<td>33%</td>
</tr>
<tr>
<td>Break down into activities</td>
<td>33%</td>
</tr>
<tr>
<td>Compose a good estimation team</td>
<td>20%</td>
</tr>
</tbody>
</table>

Interestingly, building on experience is considered by many to be a part of the ideal estimation method. This agrees well with our findings regarding the effect of utilizing data from experience.

5.5 Confounding factors

When collecting data in an industrial environment, as was the case in our study, there are likely to be many factors other than those studied that could affect the results. In compliance with proposed guidelines for research in the field of software effort estimation accuracy [28] we did a thorough check in order to identify likely confounding factors.

5.5.1 Comparison

The analysis done in [38] show that projects with large size and lack of good technological familiarity may experience larger overruns. The type of contract also seems to affect effort estimation accuracy. It is therefore necessary to explore these factors further.

In an attempt to isolate the different factors, they were checked for correlation with size of project, technical knowledge, type of contract, and the two factors that are found in this section to improve estimation accuracy: experience data and checklists. Only the comparisons that showed strong tendencies in any directions are presented.

5.6 Threats to validity

The size of the overruns was similar to previous surveys and case studies, and indicates that we did not have a particularly biased sample. However, there are several threats to the validity of our study:

- Small sample size
- Some subjective measures
- Only contractor perspective

6 Discussion

We determined to investigate the effect that selected properties of the estimation process have on estimation accuracy. It was found that the use of data from experience increased estimation accuracy. In the attempt to investigate what might be the best way of gathering and utilizing data from experience, results emerged that showed that using an estimation model did not improve estimation accuracy. It is apparently not enough to add a repeatable structure to the estimation process; it is also necessary to consider which elements and tools should be included in the process.

The results presented in section 5.3 show that there is reason to believe that estimation accuracy can be improved by the use of checklists.

6.1 RQ1 – The use of experience data

The key point is that that by building on systematic experience, it is possible to increase estimation accuracy. This is probably not surprising. The basis for almost all progress is to build on previous experience. What might be more surprising is the low percentage of project managers who utilize data from experience.
Humphrey provides the following explanation of why not more tools and methods are used by practitioners:

"Few software engineers are aware of the tools, and methods that are available, know how and where they apply, or are skilled in their use [14]."

A solution to the problem posed by Humphrey would be to have simpler tools, because they would be easier to make available and to use.

6.2 RQ2 - The utilization of estimation models

Jørgensen provides this insight on the characteristic and usability of estimation models:

"Models need to be simple, if their users are to understand them [24]."

The notion of that “the simplest is often the best” may be valid for producing such complex things as software effort estimates. However, both previous studies and common sense suggest that caution should be used when attempting to simplify the approach; it might be possible to simplify it too much. One study showed that relying only on personal memory, intuition and guessing increases overruns [2].

It is not easy to explain why using a combination of expert-based and model-based estimation appears not to improve estimation accuracy. It may be that by basing parts of the process of effort estimation on a predefined model, the structure is too restrictive. The result of providing so much structure is that it is the model and its steps and calculations importance come to have first priority, while the actual thought and consideration of the estimates become second priority.

6.3 RQ3 – The use of checklists

The results presented in section 5.3 suggest that using checklists could increase estimation accuracy.

The use of checklists facilitates the sharing of experience, and tries to preempt a dominant problem in software effort estimation, of overlooked tasks [9, 10]. In addition, it is possible to view a checklist as a very basic and simple experience database. The important point is that using checklists constitutes a simple method of increasing estimation accuracy.

Assuming that using checklists is a good idea, it is necessary to decide how to implement them. It is difficult to find a general checklist that would apply to all settings. Therefore, the use of a customized checklist for the setting or organization has been proposed. The elements of the checklist should be decided by the company, which decision should take into account the setting in which the checklist is to be used [9, 24].

To optimize experience sharing, Passing and Shepperd [8] propose that a common checklist should be used within a company. This will ensure that the checklist is as complete as possible, and that as many people as possible are able to input their knowledge and experience.

When proposing a checklist for software effort estimation, Jørgensen and Moløkken-Østvold argue that it is necessary to record assumptions, in order to make the estimates more transparent [7]. By recording assumptions, the ability to learn from previous successes and failures is increased.

7 Summary

It was found that the use of data from experience increased estimation accuracy. Regarding the use of estimation models the results showed that it did not improve estimation accuracy. The results show that there is reason to believe that estimation accuracy can be improved by the use of checklists.

Fortunately for the software development community, it might be easy to implement properties that lead to an increase in estimation accuracy. As found in previous studies, the solution to the challenge of estimating accurately is, perhaps, not to add complex and intricate models and methods for deriving the estimates [2, 39].

We recommend that some structure should be added to the estimation process so that learning from experience is enabled, but that care should be taken not to add too much. Too much structure could become a strait jacket for the estimators, and lead to a shift in focus from deriving good estimates to following the defined estimation process. Lack of freedom in the estimation process could also reduce the ability to adapt to the frequent changes in technology, type of product and methods of production.

Acknowledgements

This research was funded by the Research Council of Norway under the project INCO. We thank the studied company for providing us with data and access to their managers. Also thanks to Chris Wright for valuable comments.

References


Appendix C

“The Role of Effort and Schedule in Assessing Software Project Success – An Empirical Study”

by Kristian Marius Furulund and Kjetil Moløkken-Østvold

To be submitted to a Journal
The Role of Effort and Schedule in Assessing Software Project Success -
An Empirical Study

To be submitted to a Journal

Kristian Marius Furulund¹ and Kjetil Moløkken-Østvold²

¹Department of Informatics, P.O.Box 1080 Blindern, 0316 Oslo, Norway
kristf@ifi.uio.no

²Simula Research Laboratory, P.O.Box 134, 1325 Lysaker, Norway
kjetilmo@simula.no

Abstract

Traditionally, project success has been measured on three axes: effort, schedule and functionality. Increasingly, factors such as outcome satisfaction have also been emphasized when assessing project success. The role of estimation accuracy in defining project success was investigated by analyzing 18 of the latest projects of a software contractor. It was found that there was a slight negative correlation between project managers’ assessment of project success and effort estimation accuracy, and a slight positive correlation between project manager assessment of success and schedule estimation accuracy. It appears sensible to incorporate both subjective and objective data when assessing project success.

Keywords: Software effort estimation accuracy, software project success, project manager assessments

1 Introduction

Project overrun is a major and persistent challenge in software development [1]. Research has shown that the average effort overrun in software development projects is about 30%-40% [1]. The task of estimation is an important part of software engineering projects, and the ability to produce accurate estimates has an impact on key economic processes, such as budgeting and bid proposals.

Accurate effort estimates are also an important tool in project planning and resource allocation [2]. Inaccurate estimates may result in the wrong projects being selected [3-5], poor resource allocation, and poor quality software [6].

Given the above, it is important to consider the role that estimation accuracy plays in the assessment of project success, and whether it correlates with other, more subjective, criteria for success.

Traditionally, project success has been measured on the three axes of the iron triangle: effort (frequently labelled cost), schedule (frequently labelled time) and functionality [7]. Increasingly, other project factors, such as client and management satisfaction are coming to be emphasized when assessing project success [8].

A property of estimation accuracy is that it can be measured fairly objectively, although careful attention must be paid to definitions and measurements. The challenge with many other key project factors is that they can only be measured through subjective assessments. In various research disciplines, the problem of how to analyze and utilize subjective measures is a major challenge. Within the field of software engineering, few studies address this challenge and consider how to weigh objective data and subjective assessments. We have not found any previous studies in which subjective assessments of project success have been compared to objective data on the accuracy of effort and schedule estimations.

Given that most research focuses on the perception of estimation accuracy’s impact on project success, we wanted to look at objective estimation accuracy and perceived delivered functionality, and compare this to the assessment of project success. We see this as our contribution, and a first step, in identifying the possible relationship between objective measured criteria for project success and subjective assessments of project success.

Section 2 presents background material on measuring project success. In Section 3, the research questions are defined. Section 4 presents the method for data collection. The results are presented in Section 5. Section 6 discusses the
results based on the background given in Section 2. Section 7 concludes.

2 Background on measuring project success

Defining what constitutes project success is not a trivial task, and one can hardly say that there is a universal answer to the question of how to define it. Should it be defined by fairly objective measures, such as delivery according to budget and plan, or by subjective assessments of the project outcome? In fact, another major challenge is to actually determine whether a project was a success or not.

2.1 Diverging views on project success

According to Procaccino and Verner, the traditional view of what encompasses software project success is as follows [7]:

“The success of any software development project has traditionally been ‘defined’ from the organizational perspective, whereby a project should deliver agreed upon functionality on time and within budget”.

Traditionally, the success of a project has been measured on three axes: time (schedule), cost (effort) and functionality [10]. Schedule and effort are fairly objective data that can be measured. Functionality is more difficult to measure, and if one extends the scope of functionality to include quality, it becomes even more difficult. The difficulty in defining quality is that several different stakeholders in a project may differ as to the nature of quality [10]. An example of this is typical arguments between contractors and clients concerning whether a delivered product satisfies client standards.

In a recent paper, Agarwal and Rathod present this definition [8], originally by Baker et al., of project success [11]:

“The project is considered an overall success if the project meets the technical performance specification and/or mission to be performed, and if there is a high level of satisfaction concerning the project outcome among key people in the parent organization, key people in the project team and key users or clientele of the project effort.”

This definition of project success recognizes the need for subjective satisfaction among several, and diverse, groups of key project stakeholders. It is definitely another perspective on project success than that of the iron triangle. With this in mind, it is interesting to note that the definition says nothing about effort or schedule. It shifts the emphasis from the objective data of schedule and effort to the subjective assessments of people involved in, and affected by, the project outcome.

Procaccino, Verner et al. underline the need for collecting more than data on effort and schedule compliance to be able to assess project success [12]:

“There are important and productive measures of project success (and failure) not related to cost and schedule”.

The notion presented in the above quotation is supported by Tulnon, Jean et al. [13], in which the story of a successful project that did not meet either effort or schedule expectations is told. They present a framework in which client satisfaction is added to the three traditional dimensions of project success.

Agarwal and Rathod would like to see project success defined by a combination of subjective assessments (mainly from people not directly involved in the development of the outcome) and objective data, such as schedule overruns and economic profit [8]. They call for a more inclusive framework of criteria for assessing project success.

In addition, Procaccino and Verner [7] recognize that all stakeholders have important perspectives on project success. They studied both developers and project managers’ assessments of project success, and found that for the most part, they had the same view as to what constitutes project success.

The need for collecting information from different viewpoints is also underlined in previous estimation studies, in which the problem of getting balanced answers when collecting only the viewpoints of project managers is presented [14].

2.2 A wider perspective on project success

Increasingly, there is a tendency towards emphasising factors in addition to the three included in the traditional iron triangle when defining project success, for example client and user satisfaction, and management perception of success [8-10, 12]. This is in line with the larger movement towards a more client-oriented development process, e.g. the agile movement.

The idea seems to be that by including additional, and diverse, perspectives, the ability to assess project success might be increased. As these measures are more subjective than measuring effort, schedule, and even functionality compliance, several issues arise regarding the collection of subjective data. These challenges, and the relationship between objective data and subjective assessments, have been discussed and explored thoroughly by other researchers and lie beyond the scope of this paper.

However, it is important to explore some of the challenges of using subjective assessments made by project managers in software engineering studies. In agreement with Wohlin and Andrews [15], we believe that the data collected on subjective
assessments should not be discarded due to its weaknesses, and that a way should be found to analyse and utilize these data in the best possible way.

Analyzing and utilizing subjective data poses challenges. This is especially true when humans’ satisfaction or success rating is to be measured. The psychological phenomenon of cognitive dissonance [16] concerns the uncomfortable tension that comes from engaging in behaviour that conflicts with one’s beliefs. The theory of cognitive dissonance was first presented by Leon Festinger, and builds on one of the primal human instincts, our need to feel good and be satisfied with how we behave and perform [16].

According to Festinger, cognitive dissonance is experienced when we behave differently than we think we should. The dissonance causes us to feel physically unwell, and is a feeling that we automatically want to eliminate. This has the result that humans tend to avoid situations that make us feel uncomfortable and tasks we know that we do not perform well. Rather, we focus on situations and settings in which we can behave and perform to our satisfaction [16].

The effects of cognitive dissonance also influence our thoughts on our previous actions. If we have behaved in a way we wish we had not, we will try to neglect that in an effort to eliminate the cognitive dissonance. Instead we focus on positive aspects of our actions [16].

A consequence of the biased focus on the positive aspects of our own actions is an overrepresentation of high satisfaction assessment ratings. The lack of dependency between the normative quality of an action or outcome and perceived satisfaction, is the basis of the constant satisfaction theory [17].

A consequence of the biased focus on the positive aspects of our own effort is constant satisfaction.

The theory of constant satisfaction predicts that a human’s satisfaction assessment will be close to constant, independent of the actual outcome, effort or behaviour that is assessed [17].

### 2.3 Estimation accuracy and project manager assessment of project success

Previous studies that have provided insight into estimation accuracy, delivered functionality, and project success have often relied on project managers’ assessments of the importance of estimation accuracy relative to other criteria for success. We have found no studies where the project managers’ assessment of a project’s success had been compared to objectively measured estimation accuracy of that same project. Instead, most studies rely on the perceived importance of estimation accuracy.

Table 1 presents an overview of the findings of previous studies on project managers’ assessment of the role of estimation accuracy in determining project success. We grouped the views conveyed in the studies into two groups; those who supported the view that estimation accuracy is important for attaining project success, and those who did not.

We wish to demonstrate that there are conflicting views regarding the role of estimation accuracy in assessing project success.

**Table 1: The importance of estimation accuracy for attaining project success**

| Views supporting the view that estimation accuracy is important for attaining project success |
| Berntsson-Svensson and Aurum [9] and Procaccino, Verner et al. [18] found that accurate estimates are a major factor contributing to project success. |
| Brooks found that accurate estimates increased the possibility for project success [19]. |
| Verner, Evanco et al. found that accurate estimates are instrumental in perceived project success. Project success is often viewed in terms of staying within budget and meeting a schedule. Poor estimation will lead to deviations of results from the estimates, which will make it difficult to view a project as being successful [2]. |

| Views supporting the view that estimation accuracy is not important for attaining project success |
| In a study by Agarwal and Rathod, schedule and cost were the factors ranked with the lowest importance when assessing project success [8]. |
| Berntsson-Svensson and Aurum found that [9]: “not a single subject (…) considered “good estimates” as being an important factor”. |
| Procaccino, Verner et al. found that [12]: “Essentially, developers indicated that they value producing a quality system that meets customer/user requirements more than delivering that system on time and within budget” |
| Agarwal and Rathod found that focusing on estimated budget and schedule could divert attention from the crucial issue of completing the product’s functionality [8]. |
| Procaccino and Verner found that the most important issue for project managers was that the delivered system met requirements and worked as intended. Completion on time and within budget was ranked lowest [7]. Procaccino and Verner suggest that a reason for the low importance of estimation-related factors may be that project.
3 Definitions and research questions

3.1 Definitions

Various measures of the accuracy of effort and schedule estimation are used in this study. If the measures are to be valid, it is necessary to differentiate between different types of estimates. What estimate(s) should be used depends on the focus of the research.

If the goal is to compare the actual effort and schedule with the estimated effort and schedule, as it is in this paper, it is meaningful to use the most likely estimates at the planning stage, instead of, for example, project bids. The latter may be affected by outside factors, such as market competition.

Delivered functionality is a subjective measure, and was left to the respondents’ discretion. We do not provide a definition of project success. The reason for that is that it is one of the goals of this study to investigate which factors constitute project success. In the collection of data relating to project success it was left to the respondents to apply their personal interpretation of project success. This is in accordance with previous research that also allowed the respondents to use their own implicit definition of project success [2].

3.2 Research questions

The issue of investigating the role of effort and schedule estimation accuracy when determining project success has been approached by trying to determine how project managers’ assessment of project success correlate with these factors.

RQ1 examines the role of effort, which is closely related to cost, and RQ2 the role of schedule, or time as it is labelled by many. The role of delivered functionality is investigated in RQ3.

The reason for including a research question on functionality is two-fold. Firstly, it enables an analysis of all the traditional measures for assessing project success. Secondly, it makes it possible to investigate the relationship between the three measures.

RQ1: Is there a correlation between the project managers’ assessment of project success and effort estimation accuracy?

RQ2: Is there a correlation between the project managers’ assessment of project success and schedule estimation accuracy?

RQ3: Is there a correlation between the project managers’ assessment of project success and delivered functionality?

By answering these research questions we hope to:

• Investigate the role of estimation accuracy and delivered functionality when assessing project success.
• Take a first step in understanding the relationship between objective measured project success criteria and subjective project success assessments.
• Take a first step in exploring whether the notion of constant satisfaction is applicable to project manager assessments.

4 Method

The study was conducted in Norway from March 14th to October 16th 2006.

4.1 The company studied

The company studied is a medium-sized Norwegian software consultancy, which at the time of the study had about 300 employees. The company operates as an independent contractor and offers a wide range of complete software solutions to its various clients.

4.2 Data collection and analysis

We interviewed the project managers of 18 different projects. These projects were selected by the company, which was not informed of our research questions. The criteria for inclusion were that relevant project data was stored and available, that we were able to interview the project managers responsible, and that the projects had a workload of at least 100 man-hours. This last criterion is in line with previous surveys, in which “trivial tasks routinely handled without effort estimation” were also filtered out [4].

We collected data via personal interviews, which yields data of high quality and ensures that ambiguities are resolved [20]. It also allows the respondents to add valuable information that it is not possible to include when completing a predefined questionnaire. Another point in favour 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; hence, we were prevented from investigating as many projects as would be possible by using mailed questionnaires.

Each interview lasted between 20 and 80 minutes. 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.

We obtained both quantitative data, as well as qualitative responses from the managers, in order to
explore the research questions presented in Section 3.

Data on adhering to effort and schedule estimates was collected. The following properties were also explored: 1) the project managers’ assessment of project success and 2) the project managers’ assessment of delivered functionality. The latter measures are based on the subjective opinions of the project managers, being compared to the objective measures of estimation accuracy.

In order to compare actual effort and estimated effort, and to measure any differences in project overruns dependent on the studied properties, we used the \( \text{BREbias} \) measure, previously used in related research, e.g., [21, 22]. It is calculated as

\[
\text{BREbias} = \frac{(x - y)}{\text{min}(x, y)},
\]

\( x \) = actual and \( y \) = estimated value.

The BREbias measures both the magnitude and direction of effect when comparing the actual effort to estimated effort. BREbias is based on the Balanced Relative Error (BRE) [23, 24].

Even though MRE has been the most widely used measure of estimation accuracy [25], one must be aware that it has unfortunate properties [23, 26]. The main concern for our case is the fact that underestimated and overestimated projects are weighted unevenly.

In assessing a project’s schedule estimation accuracy, we calculated a delay factor. The delay factor is computed by considering the delivery delay (or premature delivery) relative to project size (measured in actual effort).

\[
\text{Delay factor} = \frac{\text{Delivery delay (calendar days)}}{\text{Actual effort (Person days)}}
\]

The delay factor represents the relative delivery delay, which enables comparison independent of project size.

5 Results

Of the 18 completed projects, one lacked a success assessment from the project manager and was excluded from the analysis. A full account of key project data is provided in Appendix A.

For effort overrun, the mean and median BREbias were 0.27 and 0.22, respectively (this corresponds to effort overruns of 27% and 22%). This is in line with findings in previous surveys and case studies on software estimation [27], and indicates that in this respect, the projects studied were fairly representative.

For the analysis of adherence to schedule, one project was excluded due to a significant postponement which was out of the contractor’s control (project ID 4 in the Appendix).

Of the remaining 16 projects one was completed before schedule, eight after schedule and seven on schedule. The mean delivery delay was 48 days, while the median delay was 7 days. For the delivery delay factor the mean was 0.11, while the median was 0.05.

The respondents rated the success of the project on a five-point Likert scale with the categories: very, high, medium, low and very low.

The project managers were, in general, satisfied with their projects, with as many as 14 projects being assessed with a very high or high success rating. This corresponds well with the cognitive dissonance and constant satisfaction theories.

5.1 RQ1: Project success versus effort estimation accuracy

A comparison of perceived project success and effort estimation accuracy is presented in Figure 1 and Table 2.

![Boxplot of BREbias vs PM satisfaction](image)

**Figure 1: BREbias based on project manager satisfaction**

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>St.Dev.</th>
<th>Min</th>
<th>Median</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very high</td>
<td>8</td>
<td>0.31</td>
<td>0.61</td>
<td>0.30</td>
<td>0.24</td>
<td>1.70</td>
</tr>
<tr>
<td>High</td>
<td>6</td>
<td>0.15</td>
<td>0.22</td>
<td>-0.25</td>
<td>0.19</td>
<td>0.36</td>
</tr>
<tr>
<td>Medium</td>
<td>2</td>
<td>0.10</td>
<td>0.15</td>
<td>0.00</td>
<td>0.10</td>
<td>0.21</td>
</tr>
<tr>
<td>Low</td>
<td>1</td>
<td>0.96</td>
<td>0.96</td>
<td>0.96</td>
<td>0.96</td>
<td>0.96</td>
</tr>
<tr>
<td>Very low</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1 and Table 2 show that effort overruns are somewhat higher for the projects that are assessed as having a very high success rating compared to the ones that are assessed as having a high success rating. The overruns are even a bit lower for the two projects that are assessed as being a medium success.

Seemingly, effort estimation accuracy is not an important factor for project managers when they are assessing project success. This is in agreement with findings in [7, 18] but goes against findings in [2, 8, 9].
5.2 RQ2: Project success versus schedule estimation accuracy

Perceived project success and schedule estimation accuracy are compared in Figure 2 and Table 3. As mentioned, due to a significant postponement that was out of the contractor’s control, one project was excluded from the analysis.

![Boxplot of Delay factor by PM satisfaction](image1)

**Figure 2: Delay factor based on project manager satisfaction**

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>St.Dev.</th>
<th>Min</th>
<th>Median</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very high</td>
<td>7</td>
<td>0.09</td>
<td>0.18</td>
<td>-0.07</td>
<td>0</td>
<td>0.45</td>
</tr>
<tr>
<td>High</td>
<td>6</td>
<td>0.16</td>
<td>0.18</td>
<td>0</td>
<td>0.10</td>
<td>0.47</td>
</tr>
<tr>
<td>Medium</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Low</td>
<td>1</td>
<td>0.26</td>
<td>0.26</td>
<td>0.26</td>
<td>0.26</td>
<td>0.26</td>
</tr>
<tr>
<td>Very low</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 3: Delay factor based on project manager satisfaction

Figure 2 and Table 3 show that the delay factor is slightly less for the very high category than for the high category.

In general, the projects are more in compliance with the schedule estimates than with the effort estimates, and the differences in delay factor between the different satisfaction levels are not large.

However, it is interesting to note that project managers seem to be more satisfied with projects that are delivered according to, or at least closer, to schedule. There seems to be a slight positive correlation between adherence to schedule and project managers’ success assessment.

5.3 RQ3: Project success versus perceived delivered functionality

The assessment of the percentage of delivered functionality is subjective, and will therefore have several potential weaknesses. We do, however, think that it will give some indication of the importance of delivered functionality when assessing project success.

![Boxplot of Delivered functionality (%) by PM satisfaction](image2)

**Figure 3: Delivered functionality based on project manager satisfaction**

Table 4: Delivered functionality based on project manager satisfaction

<table>
<thead>
<tr>
<th>Category</th>
<th>N</th>
<th>Mean</th>
<th>St.Dev.</th>
<th>Min</th>
<th>Median</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very high</td>
<td>8</td>
<td>128,1</td>
<td>45,2</td>
<td>90</td>
<td>115</td>
<td>230</td>
</tr>
<tr>
<td>High</td>
<td>6</td>
<td>104,2</td>
<td>12,01</td>
<td>90</td>
<td>100</td>
<td>125</td>
</tr>
<tr>
<td>Medium</td>
<td>2</td>
<td>92,5</td>
<td>3,54</td>
<td>90</td>
<td>92.5</td>
<td>95</td>
</tr>
<tr>
<td>Low</td>
<td>1</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Very low</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4: Delivered functionality based on project manager satisfaction

Figure 3 and Table 4 show that for the very high project success category, the project managers claimed, on average, that they delivered 128% of the specified functionality. Although the percentage was lower for the high category it was more than 100%.

The results indicate that project managers are more satisfied with projects that they believe to have delivered more functionality. This is in line with the findings in [7] and the results presented in the next subsection.

5.4 Manager free-text responses

The managers were also asked to name unsolicited success factors in free-text responses. Table 5 shows the share of project managers that named the different factors as important when assessing project success.

![Boxplot of Delivered functionality (%) by PM satisfaction](image3)

**Figure 3: Delivered functionality based on project manager satisfaction**

<table>
<thead>
<tr>
<th>Criteria</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Satisfied client</td>
<td>44</td>
</tr>
<tr>
<td>Knowledge increase</td>
<td>22</td>
</tr>
<tr>
<td>Build client relation</td>
<td>22</td>
</tr>
<tr>
<td>Delivered specified functionality</td>
<td>22</td>
</tr>
<tr>
<td>Making money</td>
<td>17</td>
</tr>
</tbody>
</table>

Table 5: Project success assessment criteria

The results in the table are in agreement with the viewpoint that estimation accuracy and adherence to effort and schedule is not seen as important for attaining project success [7-9, 12].

5.5 Threats to validity

The size of the overruns is similar to previous surveys and case studies [1], and indicates that we
do not have a particularly biased sample. However there are several threats to the validity of our study

- Small sample size
- Some subjective measures
- Only contractor perspective

The small sample size was a result of a trade-off between having a smaller set of observations with extensive information and high quality, and having a larger set of lower quality. As this study is exploratory in nature, the former was chosen.

Subjective measures are impossible to avoid when assessing delivered functionality and project success. One of the main goals of this study was to compare these subjective measures with more objective measures of estimation accuracy.

Having the perspective of only the contractor is an obvious weakness when measuring project success. However, we wished to measure precisely whether estimation accuracy correlated with perceived project success for the contractor. Later studies will include the perspective of the client, and how schedule delays and delivered functionality affect their perception of project success.

6 Discussion

The results showed a slight negative correlation between project manager assessments and effort estimation accuracy, and a slight positive correlation between schedule estimation accuracy and project manager assessments. A slight positive correlation between assessment of delivered functionality and project success assessments was also found. On the basis of the background given in Section 2, three different interpretations of the results are presented:

- The constant satisfaction phenomenon is valid for project manager’s assessment of project success, which entails that it is inappropriate to measure and analyze such subjective assessments when determining project success.
- The traditional measures for assessing project success (effort, schedule and functionality) are not as important as previously claimed. In addition, effort estimation accuracy is not as important as schedule estimation accuracy and delivered functionality.
- Subjective and objective measures tell different parts of the project outcome story, which indicates a need to study both subjective and objective factors when assessing project success.

6.1 Constant satisfaction and the human factor

As mentioned in Section 2, previous studies have found that the human satisfaction rate is constant. People tend to say that they are satisfied with their own previous actions, at least with those that there is apparently nothing to be done about [17]. In the study reported herein, it was found that a large majority of the project managers are satisfied with their projects. Fourteen out of 17 projects were rated with either very high or high satisfaction. Human beings emphasise the positive and remember the positive more easily than the negative [16]. These are the same mechanisms that result in our general over-optimism regarding our own abilities [5].

When asked to assess success, one should not downplay the human urge to overcome cognitive dissonance [16]. If one could manage to believe that something was a success one would feel better about oneself, and therefore one tends to be optimistic about things in which one has played a part. The implication of these theories is that whenever they are operative, one would make the same assessments whatever one is asked about, whenever one is asked about it, and by whomsoever one is asked.

It is also dangerous to view software estimation as a rational and mechanical exercise. As expert judgment estimation is still the predominant way of doing estimation, the human factor must be considered when deriving estimates [28]. The estimation process is not completely rational, where the only objective is an accurate estimate. It must be recognised that goals such as pleasing managers influence the estimation process [28]. It is likely that the actual estimates are biased in order to fulfil goals other than reaching the most accurate estimate. This makes the task of trusting; analyzing and utilizing effort estimates an even more complex endeavour.

Taken together, these factors make it difficult to study the effects of, and reasons for, satisfaction. The results of the study show that there might be reasons to think that constant satisfaction also applies to project managers’ assessment of project success.

Given all the limitations in subjective assessments, it might be concluded that no trust at all should be placed in these assessments. In the context of determining what encompasses project success, such a conclusion favours the viewpoint that the three axes of the iron triangle should still be predominant when assessing project success.
6.2 The role of estimation accuracy and delivered functionality

As presented in Section 2, many studies claim that the role of estimation accuracy is very limited when assessing project success. Most of these studies call for the collection of data on different stakeholders’ subjective assessments of project success. Obviously, the researchers have not focused on or considered the difficulties with subjective assessments that were presented in the previous subsection.

Our study found that schedule estimation accuracy and delivered functionality were correlated somewhat with project success, whereas effort estimation accuracy was not. This was also reflected in the free-text responses by the managers.

This might be due to the fact that schedule overruns and delivered functionality are more visible externally. Schedule overruns might result in fines and, in a worst-case scenario, negative press. Lack of delivered functionality might result in the project having to be redone. By contrast, effort overruns are often visible only to the manager and a financial officer. Although large effort overruns might be costly for the company, more moderate overrun appears to be an industry standard that does not have consequences [1].

Interestingly, a previous study by Lederer and Prasad reported that estimating the costs of system development was seen as a very important issue [4, 29]. Despite this, overruns are a persistent problem in software development projects. A possible reason for the lack of improvement might be, as Lederer and Prasad concluded; that project managers see estimates as important, but are not very good at estimating. It might also be a result of project managers having to choose between success criteria reducing overruns, with the latter, although seen as important, being given a low priority.

The reason for downplaying the importance of estimation accuracy may be found in human nature. Estimating is a complex and difficult task that few are very good at. According to the theory of cognitive dissonance, human nature is such that when we evaluate our own success, we focus on what we know we are good at [16]. Given that most project managers experience frequent overruns, and lack the ability to produce accurate estimates, they may wish to downplay the importance of estimation accuracy. This wish may be one reason why estimation accuracy seems to play a more limited role in the common perception of what encompasses project success.

Another reason for the lack of focus on estimation accuracy may be the common lack of confidence in estimates in the software development industry. Their accuracy is not trusted, leading to a mindset in which neither senior management, nor project managers, nor developers wish to measure estimation accuracy or be evaluated on it. As a result of this, it is not common for project managers or developers to suffer professionally or economically if estimates are not met.

Part of the case for reducing the importance of estimation accuracy is built on the notion that since it is so difficult to do, and since no schedule at all might be better than a poor schedule [2], we might as well focus on other factors, for example, subjective assessments of project outcome that both intuitively and through empirical study have been seen as better indications of project success.

The reduction in emphasis on estimation accuracy may also be a result of adding more factors to investigate, thus making the role of the estimation accuracy factor less significant. Before anyone emphasized client and user satisfaction, it was easier to look to estimation accuracy when determining project success.

6.3 Combining objective data and subjective assessments

One of the goals of this study was to investigate the relationship between subjective success assessments and objectively measured success factors.

The lack of correlation between estimation accuracy and project manager assessments can be interpreted as an indication that these factors paint different, but important parts of the complete picture regarding project success.

The case for including objective measures, such as estimation accuracy, when determining project success is based on two important notions. Objective data has general benefits over subjective data and it provides information on what still should be considered important factors with respect to achieving project success. There is no doubt that adherence to schedule and cost is important for achieving project success; if not for the project manager, at least for the company and other stakeholders. In addition to meeting economic and delivery goals, accurate estimates make it easier to manage a project, making it more likely to be successful.

On the other hand, there is a definite need for collecting and utilizing subjective assessments as well. To define and assess project success is a difficult task. Many opinions and factors must be considered. However, if one wishes to determine what constitutes project success, as much data as possible should be collected. Relying on subjective assessment by one group is not a good idea, because it only covers a narrow part of what should encompass project success. A study by Procaccino, Verner et al. showed that the viewpoint one has on the project determines what one considers when assessing project success [18].
The practice of asking project managers to assess the success of their own projects is strange, in that the success of an outcome is assessed by the developers, rather than by the recipient of the product. It would not be good practice for e.g. Coca-Cola, MacDonald’s or General Motors to only do internal studies on the quality or success of their products. The salient answers lie with the customer. This is true also for software projects, and one should ensure that the view of clients and users are represented.

The point is that one should not measure project success only by the assessment of project managers or by estimation accuracy. It should be determined by further factors. Developers, clients, managers and management emphasise different factors when assessing project success. It is probably necessary to gather the assessments of all these groups and combine them with objective data, such as adherence to schedule and budget.

7 Summary

Progress has been made regarding what constitutes software project success and the role that estimation accuracy has in determining it. The investigation of the relationship between estimation accuracy, delivered functionality, and project managers’ assessment of success showed a slight negative correlation between project manager assessments and effort estimation accuracy, and a slight positive correlation for schedule estimation accuracy and project manager assessments. A slight positive correlation was also found for delivered functionality and assessments of project success.

In general, little correlation was found. This might be due to the problems with subjective assessments and the theory of constant satisfaction, which implies that one should not put much faith in the assessments of project success made by project managers.

Another view is that estimation accuracy and the measure associated with the traditional iron triangle is less important, or at least not significant, for achieving project success.

The conclusion is that both the objective data on estimation accuracy and the subjective assessment of project managers have value. An even better determination of project success would be achieved if assessments and data from a wide range of project stakeholders where gathered, placing special emphasis on the clients’ and users’ assessments of the project outcome, as these are the stakeholders the product is developed for.

As mentioned by Agarwal and Rathod, a merging of internal objectives of development and the external interests of stakeholders would be welcomed [8]. Work for a broader and more inclusive perspective on what encompasses project success should be continued [30]. Both researchers and practitioners should take this seriously when assessing project success, and should ensure that they consider both different subjective assessments and objective data on adherence to schedule, cost and functionality.

Future research and project evaluations should strive to gather information from all stakeholders: clients, management, developers, and project managers. In addition, estimation accuracy and delivered functionality should be measured. The different factors should be combined and the relationship between them investigated. This would lead to a better framework for both defining and determining project success.

Acknowledgements

This research was funded by the Research Council of Norway under the project INCO. We thank the studied company for providing us with data and access to their managers. Also thanks to Chris Wright for valuable comments.

References


# Appendix A

<table>
<thead>
<tr>
<th>Project ID</th>
<th>Estimated effort</th>
<th>Actual effort</th>
<th>BREBias</th>
<th>Delivery delay (in days)</th>
<th>Delay factor</th>
<th>PM satisfaction</th>
<th>Delivered functionality</th>
<th>Type of contract</th>
<th>Technical knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5660</td>
<td>7301</td>
<td>0,29</td>
<td>-60</td>
<td>-0,07</td>
<td>1 - Very High</td>
<td>125 %</td>
<td>Target price</td>
<td>Good</td>
</tr>
<tr>
<td>2</td>
<td>1810</td>
<td>1570,5</td>
<td>-0,15</td>
<td>0</td>
<td>0,00</td>
<td>1 - Very High</td>
<td>100 %</td>
<td>Target price</td>
<td>Good</td>
</tr>
<tr>
<td>3</td>
<td>12000</td>
<td>14500</td>
<td>0,21</td>
<td>0</td>
<td>0,00</td>
<td>3 - Medium</td>
<td>95 %</td>
<td>Target price</td>
<td>Good</td>
</tr>
<tr>
<td>4</td>
<td>990</td>
<td>1252</td>
<td>0,26</td>
<td>0</td>
<td>1 - Very High</td>
<td>90 %</td>
<td>Other</td>
<td>Bad</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>9900</td>
<td>12600</td>
<td>0,27</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>5600</td>
<td>11000</td>
<td>0,96</td>
<td>360</td>
<td>0,26</td>
<td>4 - Low</td>
<td>100 %</td>
<td>Fixed price</td>
<td>OK</td>
</tr>
<tr>
<td>7</td>
<td>2400</td>
<td>2400</td>
<td>0,00</td>
<td>0</td>
<td>0,00</td>
<td>3 - Medium</td>
<td>90 %</td>
<td>Other</td>
<td>Good</td>
</tr>
<tr>
<td>8</td>
<td>12000</td>
<td>12500</td>
<td>0,04</td>
<td>0</td>
<td>0,00</td>
<td>1 - Very High</td>
<td>120 %</td>
<td>Fixed price</td>
<td>Good</td>
</tr>
<tr>
<td>9</td>
<td>3430</td>
<td>4076</td>
<td>0,19</td>
<td>46</td>
<td>0,09</td>
<td>2 - High</td>
<td>110 %</td>
<td>Fixed price</td>
<td>OK</td>
</tr>
<tr>
<td>10</td>
<td>3135</td>
<td>2507</td>
<td>-0,25</td>
<td>35</td>
<td>0,11</td>
<td>2 - High</td>
<td>100 %</td>
<td>By the hour</td>
<td>Good</td>
</tr>
<tr>
<td>11</td>
<td>700</td>
<td>947</td>
<td>0,35</td>
<td>0</td>
<td>0,00</td>
<td>2 - High</td>
<td>100 %</td>
<td>Target price</td>
<td>Good</td>
</tr>
<tr>
<td>12</td>
<td>13500</td>
<td>16000</td>
<td>0,19</td>
<td>0</td>
<td>0,00</td>
<td>2 - High</td>
<td>90 %</td>
<td>Fixed price</td>
<td>Good</td>
</tr>
<tr>
<td>13</td>
<td>1582</td>
<td>4268</td>
<td>1,70</td>
<td>86</td>
<td>0,16</td>
<td>1 - Very High</td>
<td>230 %</td>
<td>By the hour</td>
<td>OK</td>
</tr>
<tr>
<td>14</td>
<td>953</td>
<td>1164</td>
<td>0,22</td>
<td>14</td>
<td>0,10</td>
<td>1 - Very High</td>
<td>100 %</td>
<td>Target price</td>
<td>OK</td>
</tr>
<tr>
<td>15</td>
<td>1695</td>
<td>1830</td>
<td>0,08</td>
<td>102</td>
<td>0,45</td>
<td>2 - High</td>
<td>125 %</td>
<td>Target price</td>
<td>OK</td>
</tr>
<tr>
<td>16</td>
<td>1400</td>
<td>1950</td>
<td>0,39</td>
<td>109</td>
<td>0,45</td>
<td>1 - Very High</td>
<td>150 %</td>
<td>By the hour</td>
<td>OK</td>
</tr>
<tr>
<td>17</td>
<td>3500</td>
<td>2700</td>
<td>-0,30</td>
<td>0</td>
<td>0,00</td>
<td>1 - Very High</td>
<td>110 %</td>
<td>Target price</td>
<td>Good</td>
</tr>
<tr>
<td>18</td>
<td>1511</td>
<td>2047</td>
<td>0,35</td>
<td>74</td>
<td>0,29</td>
<td>2 - High</td>
<td>100 %</td>
<td>By the hour</td>
<td>OK</td>
</tr>
<tr>
<td>Mean</td>
<td>4543</td>
<td>5590</td>
<td>0,27</td>
<td>48</td>
<td>0,11</td>
<td></td>
<td></td>
<td></td>
<td>114 %</td>
</tr>
<tr>
<td>Median</td>
<td>2768</td>
<td>2604</td>
<td>0,21</td>
<td>7</td>
<td>0,05</td>
<td></td>
<td></td>
<td></td>
<td>100 %</td>
</tr>
</tbody>
</table>
Appendix D

“A Review of Group Techniques for Software Effort Estimation”

by Kristian Marius Furulund
Abstract

To use groups when deriving software effort estimates could be a good approach that will have beneficial effects leading to higher estimation accuracy. In that respect it is interesting to have a look at existing group techniques for software effort estimation. A brief background on the pros and cons of using groups for problem solving is given. Then the paper presents five different group techniques: 1) unstructured groups, 2) Delphi, 3) Wideband Delphi, 4) Planning Poker and 5) Decision Markets. Advantages and disadvantages for all the five techniques are discussed, with a special focus on their take on anonymity and face-to-face interaction. There exists very limited research on the suitability of these techniques for deriving software effort estimates.

1 Introduction

Estimation is defined by the Oxford English Dictionary as “The action of appraising, assessing, or valuing” or “The process of forming an approximate notion of (numbers, quantities, magnitudes, etc.) without actual enumeration or measurement [1]. From this definition it follows that the task of estimation is not easy to do precisely.

The problem with estimating the scale of a software project, is like all other activities concerning guessing what will happen in the future – There is no way to be certain.

According to Rowe and Wright, the best way to guess about the future, to estimate, is to use previously recorded data from a similar settings [2]. Rowe and Wright point out that human judgment compares poorly to statistical or computational models. The problem is that many software companies fail to learn from their mistakes, and do not record data which could help them become less dependent of human judgment [3].

As statistical data rarely is available, expert judgment is most frequently used [1]. Barry Boehm defines expert judgment as “Consulting with one or more experts who use their expertise and understanding of the proposed project to arrive at an estimate of its cost [4].”

Project overruns is a major and persistent challenge in software development [5]. Research has shown that the average effort overrun in software development projects is about 30%-40% [5]. The task of estimation is an important part of all software engineering projects, and the ability to produce accurate estimates has an impact on key economic processes as budgeting and bid proposals. Accurate effort estimates are also an important tool in project planning and resource allocation [6]. Inaccurate estimates may results in the wrong projects being selected. [7-9], poor resource allocation and poor quality software [10].

Reasons for the continuing cost overrun problems could be lack in estimation ability and the use of inefficient techniques when producing the estimates.

Many argue that the use of groups, where several human estimates is combined, is a good method for optimizing the estimates [4, 11-14]. Groups are utilized in many settings and have obvious positive features as division of labor, and more sources of knowledge and information.

According to Brown some tasks are too big and complex to be solved by an individual alone [15]. Brown claims that groups have clear advantages over an individual making all decisions. The division of labor, the cancelling out of biases and the motivational factors caused by the creation of a group identity are the most important improvements [15]. The importance of dealing with biases is central in the argument for combining estimates. To prevent over optimism or over pessimism one wishes to cancel out biases. Boehm claims that using more than one expert will lead to less biased estimates [4]. Interacting groups also have positive attributes like knowledge from a variety of sources and creative synthesis [2].

In his book “The Wisdom of Crowds” James Surowiecki argues for the superiority of groups in solving tasks [16]. His main point is that more
people will add more perspectives, making the sum of knowledge that the group could base their decision on more complete. With more knowledge it is easier to make a good decision. Surowiecki underlines the need for diversity, independence and decentralization within the group for it to fulfill its potential.

As mentioned earlier there are also strong support for the utilization of groups to do software effort estimation [4, 11-14]. Research within the software development field has shown that estimates made by groups are more accurate than those done by individuals [12, 17].

However, using a group to perform a common task also has different social and practical consequences. Brown presents theories claiming that solving problems with groups is not always better than doing it individually [15]. A group may perform better than one person, but not as many times better as the number of group members. This effect is known as the Ringelmann-effect [18].

Brown presents a theory by Steiner who, in compliance with the Ringelmann-effect, claims that a group’s actual productivity is never as good as its potential productivity [19]. The reason for this is coordination issues and social dynamics. Because of the need to coordinate actions and discuss different options a group becomes less productive than a mathematical aggregation of the team members’ individual potential productivity.

Groups are also often subject to confirmation bias, which means that they will only listen to and look for information that confirm the standpoint that they intuitively think is right [16]. This over focus on reaching consensus instead of paying attention to dissent is labeled by Janis as Groupthink [20].

Brown [15] presents research results showing that being exposed to others opinions causes polarization. This effect, known as the polarization effect, is in play when a group of individuals ends up with a more extreme standpoint than the most extreme of the initial individual standpoints.

In addition to the loss in group productivity due to coordination issues and social dynamics [19], Latene, Williams et al. identified another effect which they labeled, social loafing indicating that individuals become lazy when they are in a group, and do not put in as much effort in as they do when they solve a task alone [21]. Holt challenged this theory and found that through good preparation and motivation facilitating you could get a group to perform over its potential productivity [15].

In addition to the positive effects of utilizing groups, several potential undesired effects are identified. In this paper the focus will be on the following three:

**Coordination issues**

A group will experience loss of productivity due to coordination issues [19].

**Social Dynamics**

A group could fail to function properly due to potential social, personal and political conflicts [15]. One could also experience that group members get overly influenced by the more assertive members, by figures of authority or political considerations [4]. The main point here is that individual group members will not contribute with their knowledge and insight due to political and social consideration.

Groupthink and polarization, is closely related to social dynamics. They can be seen as special and typical effects of the social dynamics in a group.

**Groupthink and polarization**

As a result of a group’s over focus on reaching consensus [16], it will not accept or consider dissent or difference of opinion [20]. This effect is labeled groupthink. The desired effects of adding more perspectives and cancelling out of biases are largely reduced.

The polarization effect is the opposite of what one achieves through cancelling of biases. Instead of reducing a bias in a direction, it is further strengthen through a polarization of view points [15].

A presentation of the different group processes will be given in Section 3. A discussion about their positive and negative effects, based on the introduction in Section 1, will be given in Section 4. Section 5 concludes.

2 Research questions

To use groups when deriving software effort estimates could be a good approach that will have beneficial effects leading to higher estimation accuracy. In that respect it is interesting to have a look at existing group techniques for software effort estimation. The following research question will be addressed.

**RQ1: Which group techniques for software effort estimation exist?**

3 Group techniques

Most group techniques will utilize the potential positive effects of groups. However, what most often separates different group techniques is which negative effects they try to preempt, and how they try to do it. It is often a trade-off between preempting a negative effect, and reducing a positive effect.

3.1 Statistical groups

In a statistical group there is no interaction between the group members. They are only a group in
the sense that their individual estimates are combined statistically.

When considering how to combine estimates given by several individuals into an estimate, one can use well known statistical methods. Computing the mean or median of the different individual estimates will give us one estimate based on the multiple estimates. Jørgensen [13] claims that simple average often works as the best method for combining estimates.

3.2 Unstructured groups

The basic way of reaching a common estimate through group interaction is using what is labeled as an unstructured group. An unstructured group is what we normally refer to as a group, meaning several people coming together, sharing their viewpoints and reaching a common decision. The word unstructured is used as the group is not given any instructions to work according to a specified structure.

3.3 Structured groups

The structured processes vary largely in how much structure they provide and how much face-to-face interaction between the group members they allow. The techniques that will be discussed are:

- Delphi
- Wideband Delphi
- Planning Poker
- Decision Markets

3.3.1 Delphi

The Delphi technique was developed by the RAND corporation in 1963 [22]. Delphi’s goal is to “obtain the most reliable consensus of opinion of a group of experts [22].”

The Delphi method is an example of an attempt to structure the interaction among the group members. The reason for doing this is to preempt non desirable effects of group processes caused by the social status and politics within a group. In short the Delphi method limits the group members’ interaction with each other to getting to know the mean estimate for the group. It then asks them to estimate again until the deviation in the estimates has reached an acceptable level.

The Delphi technique has four key features[2]:

- Anonymity
- Iteration
- Controlled feedback
- Statistical aggregation

These features are chosen to preempt the undesired effects that may be the result of unstructured groups.

Through the anonymity feature one hopes to deal with the group members desire to conform with a dogmatic group member, the group majority or social and political expectations [2].

The iteration feature enables the group members to learn from each other and also gives them an opportunity to change their opinion with out the fear of losing face or credibility [2]. Through controlled feedback and statistical aggregation one tries to prevent that the most vocal, political or social influential group members get to much influence on the final result reached by the group. Merely providing a framework should also deal with some of the coordination issues [15].


1. A coordinator presents each expert with a specification and a form upon which to record estimates.
2. The experts fill out forms anonymously. They may ask questions of the coordinator, but should not discuss the situation with each other.
3. The coordinator prepares a summary of the experts’ responses on a form requesting another iteration of the experts’ estimate, and the rationale behind the estimate.
4. The experts fill out forms, again anonymously, and the process is iterated for as many rounds as appropriate.

No group discussion is to take place during the entire process.

3.3.2 Wideband Delphi

As a result of research done by Farquhar showing different problems with the Delphi technique [23]. Boehm and Farquhar updated the Delphi technique. The main problem discovered by Farquhar was lack of communication between the group members. Boehm and Farquhar named their updated version Wideband Delphi underlining the widening of the communication channels [4]. The Wideband Delphi technique is very similar to the Nominal Group Technique, also know as the estimate-talk-estimate technique [24]. Due to its similarities that techniques is not presented or discussed in this paper.

Boehm gives the following steps for the Wideband Delphi Technique [4].

1. A coordinator presents each participant with a specification and an estimation form.
2. The coordinator calls a group meeting in which the experts discuss estimation issues with the coordinator and each other.
3. Experts fill out form anonymously
4. The coordinator prepares and distributes a summary of the estimates on an iteration form.
5. The coordinator calls a group meeting, specifically focusing on having the experts
discuss points where their estimates varied widely
6. The experts fill out forms, again anonymously, and steps 4 to 6 are iterated for as many rounds as appropriate.

The big difference between Wideband Delphi and the original Delphi technique is that the estimators meet face to face to explain their estimates.

3.3.3 Planning Poker

Planning Poker is an updated version of Wideband Delphi, which has been adapted for an agile software development process. Planning Poker was first described by Greening [25] and is part of Mike Cohn’s book on Agile Estimation and Planning [26]. Planning Poker is designed to solve two problems that Greening saw with estimation. He felt that estimation was taking too much time and he was concerned about the fact that not the whole development team participated in the estimation process [25]. Planning Poker should be seen more as an alternative to the frequently used unstructured group technique than to the similar Wideband Delphi.

As with Wideband Delphi, one hopes that Planning Poker will provide a common understanding of the tasks and assumptions, in addition to estimates. Haugen emphasizes that Planning Poker makes sure that all the estimators’ opinions are heard [10]. As with Wideband Delphi, Planning Poker is an attempt to combine the positive effects of having a strict structure to the group process including protecting the anonymity of the group members, with the benefits from an unstructured group process with face-to-face interaction. Haugen labels Planning Poker a “semi-structured estimation process [10].”

Greening gives the following steps for a Planning Poker process [25]:
1. The customer (or the project manager or responsible developer) reads a user story.
2. There is a discussion clarifying the user story
3. Each team member (programmer) writes their estimates on a note card without discussing with the others
4. All turn over their cards
5. If there is an agreement, no discussion is needed. Just record the estimate and move on.
6. If there is disagreement, let the group discuss their estimates and try to reach consensus.

3.3.4 Decision Markets

Hanson defines Decision Markets as “Decision Markets are (markets) designed primarily for the purpose of using the information in market values to make decisions [27].”

A decision market is set up like a stock market, where traders are invited to invest money in the alternative, stock, that they think will be the eventual outcome. A trader holding a stock that becomes the actual outcome receives a fixed amount of money. Through the dynamics of a market this result in higher stock prices for the alternatives that the most people think will be the outcome, creating a likelihood distribution for the different outcomes.

According to Surowiecki a market is wise because it aggregates the opinions of traders, that are diverse, independent of each other and carry with them local knowledge [16].

Inspired by Surowiecki, Berndt, Jones et al. advocate for the use of Decision Markets to do software effort estimation [28]. They underline that through letting all project stakeholders take part in the decision market, one ensures diversity in the input to the estimation process, and also aggregates the knowledge from all the project stakeholders. According to Berndt, Jones et al. another positive feature with Decision Markets is that the different traders could apply the estimation technique they wish, enabling a combination of different estimation techniques.

A decision market is, as Delphi, a way of aggregating different opinions without face-to-face meetings. Like Delphi a Decision Market would like to preempt the social and political problems caused by the use of groups, while at the same time utilize the knowledge increase you get when using a group.

The main difference is the way in which the knowledge and opinions of the group members are aggregated.

3.4 Comparison

In [29] Moløkken-Østvold and Haugen categorize different group techniques based on structure, anonymity, interaction and overhead. Below this table has been extended to also include Decision Markets.
Table 1: Group technique overview

<table>
<thead>
<tr>
<th>Technique</th>
<th>Structure</th>
<th>Anonymity</th>
<th>Interaction</th>
<th>Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unstructured group</td>
<td>None</td>
<td>No</td>
<td>Yes</td>
<td>Limited</td>
</tr>
<tr>
<td>Delphi</td>
<td>Heavy</td>
<td>Yes</td>
<td>No</td>
<td>Major</td>
</tr>
<tr>
<td>Wideband Delphi</td>
<td>Moderate</td>
<td>Limited</td>
<td>Limited</td>
<td>Moderate</td>
</tr>
<tr>
<td>Planning Poker</td>
<td>Light</td>
<td>No</td>
<td>Yes</td>
<td>Limited</td>
</tr>
<tr>
<td>Decision Markets</td>
<td>Heavy</td>
<td>Yes</td>
<td>No</td>
<td>Major</td>
</tr>
</tbody>
</table>

The table shows that Delphi and Decision Markets have many of the same properties. Wideband Delphi and Planning Poker are also quite similar. The processes’ take on anonymity seems to be defining for the rest of the properties. To have anonymity no face-to-face interaction can take place. In order to achieve that one needs more structure in the process, which in turn leads to more overhead.

4 Discussion

In this section the different group processes’ ability to utilize the potential gains of group problem solving and their ability to preempt potential undesired effects will be discussed.

4.1 Unstructured group

The effects of utilizing an unstructured group process have been thoroughly discussed in the introduction, and also through out the presentation of the other techniques.

An unstructured group enables the utilization of these positive group effects [15]:

- The division of labor
- The cancelling out of biases
- The motivational factors caused by the creation of a group identity

However, as no preemption strategy is applied all the possible undesired group effects discussed in this paper have a higher likelihood for occurring than for the other group techniques.

The most evident advantage with an unstructured approach is that is leads to very little overhead with regards to facilitating the opinion sharing and aggregation. As mentioned earlier different studies have shown that to use an unstructured group approach outperforms individuals when doing software effort estimation [12].

The other group processes’ features are discussed and studied with the unstructured group processes as a basis.

4.2 Delphi

In reviewing research on the Delphi technique, Rowe and Wright conclude that one should use Delphi when no statistical models or data do exist [2]. “Delphi groups are substantially more accurate than individual experts and traditional groups and somewhat more accurate than statistical groups [2].”

Rowe and Wright conclude that if we have to use humans to do the estimation, which is almost always the case in software development projects [13], let them do it using the Delphi technique [11].

Rowe and Wright [2] identified the consensus reaching ability of the Delphi technique, but they are skeptical to the nature of the consensus. However, they question if the reason that the groups reaches a consensus is a result of the team members actually changing their minds, or if it is just a result of them wishing to conform to the group.

That one can increase a group’s productivity by proper preparation and facilitation is interesting in a Delphi context as the Delphi technique provides some facilitation by giving clear instructions to the group members [15].

Some undesired effects of groups could be increased by using the Delphi-technique. E.g. getting to know the mean estimate in a Delphi process could cause the individuals to further adjust their estimates in a more extreme direction.

Most studies on the Delphi-technique is done in other settings than software effort estimation. The suitability of the Delphi-technique to derive software effort estimates is poorly understood.

4.3 Wideband Delphi

Stellman and Greene claim that, in addition to getting estimates for the different tasks, the use of the Wideband Delphi technique gives the development team a more detailed Work-Break Down Structure and a list of assumptions [30]. A main reason for inaccurate estimates is that the estimator does not know enough about the task he is estimating. When a group discusses tasks and assumptions they increase their individual and common understanding of the problem they should solve. The increase in understanding makes it easier to make accurate estimates, and perhaps more importantly to actually solve the problem one was supposed to solve. As a group the estimators are more likely to remember task that initially has been overseen. Overseen tasks has been found to be a major contribution to software effort overruns [31, 32].

According to Wiegers, using Wideband Delphi also results in a bigger commitment to own estimates [33]. In a setting where the estimators are more committed to their estimates and feel that they are
held accountable, the estimates tend to become more accurate [13]. By allowing more group interaction and face-to-face communication one reduces the anonymity, risking jeopardizing the main goal of the Delphi technique: The preemption of undesired group effects due to social and political concerns. On the other hand the increase in interaction may cause an increase in group identity and information sharing, leading to increased productivity.

There has been limited or no research on Wideband Delphi’s ability to produce more accurate software effort estimates.

**4.4 Planning Poker**

Planning Poker is very similar to Wideband Delphi, and therefore most of the effects discussed under 4.3 do also apply to Planning Poker.

In a recent study Haugen compared Planning Poker to an unstructured group approach, and found that Planning Poker made a larger share of the group involved in the estimation process [10]. Haugen also observed that the estimation team took a liking for the process. They thought that it was fun, and that it was more effective. However, no objective measure of this effect was done. The study also found that Planning Poker groups provided more accurate estimates for familiar tasks, while as for unfamiliar tasks the Planning Poker groups performed worse.

The polarization effect seems to be stronger for the Planning Poker groups [10]. As Planning Poker causes more group discussion it is also more likely to be affected by the undesired group effects as polarization and groupthink. Planning Poker performed better when the team had experience with similar tasks. This may not be surprising as one of the goals with Planning Poker is to better facilitate information sharing among the team members, making it more likely to perform better when there is more experience and knowledge to be shared.

Another recent study by Moløkken-Østvold and Haugen on group processes and Planning Poker found that estimates derived by Planning Poker were less optimistic than the average of individual estimates [29]. However, they found that compared to the individual estimates, Planning Poker estimates were less accurate. The tasks in the planning poker group were significantly larger. Moløkken-Østvold and Haugen argue that a reason for that might be that through Planning Poker, and the voicing of more opinions that it enables, the developers got a more comprehensive view of what the task encompasses, leading them to doing a more thorough job when carrying out the tasks [29]. A tendency for to find complying with scope and functionality more important than to comply with schedule and cost estimates was found in a recent study [34].

**4.5 Decision Markets**

The most famous decision market is the Iowa Electronic Markets (IEM) who has been set up to predict outcomes of US presidential primaries and general elections [35]. The IEM has a good record of both predicting the presidential candidates and the president, and also the percentage distribution between the candidates. The accuracy has been very impressive, outperforming most major opinion polls [35].

According to Berg and Rietz Decision Markets have these proven positive effects [35]:

1. *The markets give continuously updated dynamic forecasts.*
2. *Through the price formation process, the markets aggregate information across traders, solving what would otherwise be complex (at best) aggregation problems.*
3. *The evidence suggests that such markets give unbiased, relatively accurate forecasts well in advance of outcomes.*
4. *These forecasts can outperform existing alternatives.*
5. *The evidence suggests that market dynamics can overcome biases that individual traders may have, effectively eliminating them from forecasts.*

In his book “The Wisdom of Crowds”, James Surowiecki argues for the information, knowledge and decision potential that lies in groups of people [16]. However, to make a group function properly one needs it to be diverse, the members’ opinions and inputs must be independent of each other, and it has to be decentralized. Surowiecki has a great belief in the combination of different people’s opinions, but sees the challenge in aggregating these opinions in a manner that does not cause many of the undesired group effects. Surowiecki sees Decision Markets as the best way to get knowledge from as many as possible, and to aggregate them in a decentralized way that upholds the needed diversity and independence.

No research on the use of Decision Markets to do software effort estimation has been found. However, a recent paper by Berndt, Jones et al. describes an ongoing study where Decision Markets are applied to do software effort estimation [28]. As Decision Markets have a good track record for forecasting other outcomes, this study is welcomed.
4.6 Comparison

The table below show a presentation of the different group processes’ ability to preempt undesired effects of doing problem solving as a group. The ability to preempt these effects must not be seen as a way in which one can classify the processes. It is important to note that all the group processes encompasses a trade-off between preempting undesired group effects, increasing the probability and ability to take advantage of potential positive group effects and generated overhead.

Table 2: The group techniques preemption abilities

<table>
<thead>
<tr>
<th></th>
<th>Preempts</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coordination</td>
</tr>
<tr>
<td>Unstructured Group</td>
<td>No</td>
</tr>
<tr>
<td>Delphi</td>
<td>Yes</td>
</tr>
<tr>
<td>Wideband Delphi</td>
<td>Yes</td>
</tr>
<tr>
<td>Planning Poker</td>
<td>Yes</td>
</tr>
<tr>
<td>Decision Markets</td>
<td>Yes</td>
</tr>
</tbody>
</table>

As mentioned under Section 3.4, how important anonymity is seen to be, has a significant impact on the group process. The focus on anonymity is a result of a belief in the need for independence between the different group members, in order to fully utilize the group’s potential [16]. Through anonymity one hopes that the different group members will voice their own opinion without being influenced by political, social or other considerations. In order to have anonymity one can not have face-to-face discussions. By not having face-to-face discussion one avoids, or at least reduces, the chance for social and political conflicts, polarization and groupthink. On the other hand avoiding face-to-face discussions creates more overhead and a need for more structure in the process.

An unstructured group process does not do anything to try to protect the anonymity and independence of the group members, resulting in no preemption of any of the undesired group effects.

Wideband Delphi and Planning Poker represent a combination between protecting independence and allowing face-to-face communication. In different ways the two techniques make sure that the group members’ initial estimates are independent of each other. Wideband Delphi does it through making the initial estimates anonymous, while in Planning Poker it is done through the simultaneous showing of the estimates. However, after the initial estimates are revealed they allow face-to-face discussions. The hope is that the potential positive effects of face-to-face discussions will be worth the increase in the chances for political and social conflict, polarization and groupthink.

Both Delphi and Decision Markets have a stronger emphasis on the need for independence between the group members in order for them to produce a good outcome. That is why the two techniques in different ways sets up structure and generates overhead to protect the anonymity of the group members. By not allowing any face-to-face interaction one preempts the possibility for groupthink and political and social conflicts. As both techniques have a way in which the average or aggregated opinions are reported back to the group members, the possibility for polarization is not preempted as good, as the other undesired effects.

Delphi, Wideband Delphi, Planning Poker and Decision Markets are all build on the assumption that to do software effort estimation as a group is a good idea. In different ways they try to make sure that the knowledge and information held by the group members are shared, so that they together can reach a good estimate.

In addition to reaching a more accurate estimate, having the software developers share their views could have other positive effects for the software development project.

The developers could gain a better understanding of the tasks and what they encompass [29]. It may also have the result that the tasks become more complex, as more views and interpretation of the task is voiced. Knowing developers tendency to prioritize scope and functionality over cost and schedule compliance, this might lead to an undesired increase in scope [34].

5 Conclusion and further studies

It is clear that to utilize groups for software effort estimation has both potential benefits and disadvantages. The different group techniques try, in different ways, to maximize the positive effects and to reduce the negative effects as much as possible.

Little research is done on the use of these different estimate combination techniques for software effort estimation. There are strong arguments for all the techniques presented in this paper, and there is an obvious need for more research on the different techniques abilities to produce accurate software effort estimates. Through empirical studies one could find which techniques to use, and when and how to use them. It is likely that the different techniques will
be better fits for some settings than for others. Research is needed to find out which techniques to apply in which setting.

References


