Impact of Effort Estimates on Software Project Work

Magne Jørgensen[i], **Dag I. K. Sjøberg**[ii]
{magnej, dagsj}@ifi.uio.no
i) Department of Informatics, University of Oslo, Norway
ii) Simula Research Laboratory, Norway

Abstract

This paper presents results from two case studies and two experiments on how effort estimates impact software project work. The studies indicate that a meaningful interpretation of effort estimation accuracy requires knowledge about the size and nature of the impact of the effort estimates on the software work. For example, we found that projects with high priority on costs and incomplete requirements specifications were prone to adjust the work to fit the estimate when the estimates were too optimistic, while too optimistic estimates led to effort overruns for projects with high priority on quality and well specified requirements.

Two hypotheses were derived from the case studies and tested experimentally. The experiments indicate that: 1) Effort estimates can be strongly impacted by "anchor" values, e.g., early indications on the required effort. This impact is present even when the estimators are told that the anchor values are irrelevant as estimation information. 2) Too optimistic effort estimates lead to less use of effort and more errors compared with more realistic effort estimates on programming tasks.

Keywords: effort estimation; estimation accuracy; project management;

1 Introduction

Work expands so as to fill the time available for its completion, The Parkinson's principle [1].

Frequently, the required effort to develop a software product has to be estimated before a break down of the project in activities and a corresponding plan of the project is available [2]. This is, for example, necessary to give the customer or the developer organisation an early indication of the project costs. In this paper we denote the effort estimates developed at this stage *pre-planning estimates*. Pre-planning estimates may be based on very limited information about the product requirements and the development activities. The next estimation of effort may be conducted when more information about the product and the project activities are available. We denote these estimates *detailed planning estimates*.

In this paper we study how the pre-planning effort estimates impact the detailed planning effort estimates and how the detailed planning effort estimates impact the process and outcome of software development projects. Although the *existence* of impacts of the effort estimates on the project work is of no surprise to experienced project leaders, we believe that they are usually unaware of their size and nature. For example, Sengupta and Abdel-Hamid [3] report in a study of software effort estimation that the estimators did not show much understanding of the cause-effect complexity of the software project environment. We have been unable to find much discussion on how an effort estimate may impact the project work in the research literature or text books on software effort estimation. The few references we have found, [3-5], seem to have had little impact on subsequent software estimation research and processes.

To better understand the size and nature of the impacts of the effort estimates, we conducted two case studies and two experiments. An overview of the case studies and experiments is presented in Section 1.1. Section 1.2 provides definitions of important terms used in the description of the results from the studies. The results from the case studies are reported in Sections 2 and 3, and from the two experiments in Sections 4 and 5. The results are compared with results from related studies in Section 6. Section 7 discusses threats to the validity of the studies. Our work has the goal of improving the software estimation and management processes. The conclusions of this initial work and description of further work to achieve this goal are described in Section 8.

1.1 Overview of the Studies

Table 1 gives an overview of the four studies we have conducted. More information about the studies, e.g., the case study questionnaires, the experiment material, and the data from the experiments, can be downloaded from *www.simula.no.*

Described in:	**Type of study:**	**Main data sources and subjects:**	**Main focus:**
Section 2	Case study of an e-business development organisation.	11 interviews. Project documentation. Software professionals.	The impact of pre-planning estimates on the detailed planning estimate. The impact of too low/high effort estimates on project work and product quality.
Section 3	Case study of a telecom development organisation.	6 interviews. 8 experience reports. Software professionals.	The impact of too low/high effort estimates on project work and product quality in relation to project priorities.
Section 4	Experiment. Two groups with different treatments, i.e., a between subject research design.	38 + 20 computer science students (two similar experiments on different groups of students).	The impact of the pre-planning estimates on the detailed planning estimate.
Section 5	Experiment. Each subject exposed to different treatments, i.e., a within subject research design.	10 computer science students.	The impact of too low effort estimates on effort usage and program quality.

Table 1 Overview of the studies

From the two case studies we derived the following two hypotheses:

Hypothesis 1: Pre-planning effort estimates have a major impact on the detailed planning effort estimates. This impact is present even when the estimators are told that the pre-planning effort estimates are not based on historical data or expert knowledge.

Hypothesis 2: Too low effort estimates lead to less effort and more errors compared with more realistic effort estimates (for small programming tasks).

The role of the experiments is to test two of the indications from the case studies in a more controlled environment.

1.2 Definitions – Theory and Practice

When discussing effort estimates and estimation accuracy in this paper it is important to be aware of the differences between estimated effort, planned effort and price-to-win. Assume the possible effort usage of a software development activity can be described as a probability distribution. This means that each possible effort interval has a connected probability, e.g., that the actual effort is with a probability of 50% within the interval 1000 to 1500 work-hours. According to, for example, Moder *et al*. [6] the effort *estimate* is the median value of that probability distribution. If the possible effort values follow a beta-distribution with minimum effort *a*, most likely effort *m* (the mode), and maximum effort *b*, then the median effort, i.e., the estimate, is approximately:

$$Estimate = \frac{a + 4 \cdot m + b}{6}$$

The definition of the estimate as the median value of a probability distribution means that there is (theoretically) a 50% probability of not exceeding the estimated effort. A project leader may want the probability of keeping the *planned* effort to be higher than 50%. Consequently, the project leader will add effort to the estimate to get the planned effort. The amount of effort added should correspond to the project risks and the desired probability of not exceeding the planned effort [7].

The *price-to-win* is what the customer accepts to pay for the software development work. The price-to-win should be *based on* the effort probability distribution, but is not an estimate itself. A development project may choose to accept a high probability of using more effort than the effort corresponding to the price-to-win, e.g., to get a reference customer.

The interpretations of estimate, plan and price-to-win were, unfortunately, blurred in the two development organisations we studied. Typically, no difference was made between the estimated and planned effort, and the planned effort was sometimes based on price-to-win instead of the estimate.

Estimation accuracy is frequently defined as the Magnitude of Relative Error (MRE) [8]:

$$MRE = \frac{|\,Actual\ effort - Estimated\ effort\,|}{Actual\ effort}$$

To compare the estimation accuracy of different development projects, the following criteria should be met:

- There is a common interpretation of 'estimate', e.g., the effort estimates are interpreted as the median values of distributions of possible effort values.
- The estimates have no impact on the software work, i.e., no 'self-fulfilling' estimates.
- The specified products correspond to the delivered products, i.e., no change in the basic assumptions of the estimate.

In our experience, very few software projects meet all these criteria. For this reason, when we describe the values that the organisations presented as their projects' estimation accuracy values, we will indicate how this accuracy was achieved, i.e., how much the criteria were violated.

2 Study of an e-business Development Organisation

We studied the Norwegian branch of a large international software company that delivers e-business solutions on internet platforms. The Norwegian branch was established in 1998 and includes about 130 employees. Most projects involve four separate disciplines: project management, graphical design, human computer interaction (HCI) development and technical software development. Most projects are small (less than 6 person-months) and there is a large variety in the maturity of the customers; some of them are very informal regarding requirements specifications. The organisation has defined and described work processes that the software projects are supposed to follow. There was no data available that indicated the estimation accuracy of the earlier projects, but there had been some very large cost overruns recently.

We collected information through interviews with two project leaders, two graphical designers, one HCI developer and six programmers. The interviewed subjects, selected by one of the managers, were believed to be some of the most experienced employees of the respective roles. The interviews focused on the actual estimation processes, the accuracy of the estimates, estimation training and improvement suggestions. The time used in each interview was 1– 1.5 hours. One of the authors participated in the estimation of one project.

The interviews indicated that a typical estimation and planning process consisted of the following steps:

- The customer requests an e-business solution. Typically, the customer has an incomplete specification, an idea of how much he/she is willing to spend and a limited knowledge about the possibilities and technical solutions.
- The project leader outlines several alternatives with costs not too far from what it is believed that the customer will accept. The costs of these alternatives (pre-planning estimates) are based on very rough expert estimates. A standard risk buffer is added.
- The customer eventually accepts one of the outlined alternatives. Then, the project is analysed in more detail, re-estimated (detailed planning estimate) and planned in accordance with the level of functionality and costs accepted by the customer.

The pre-planning effort estimate was frequently based on little knowledge about the final product and calculated before the project members were allocated. There was no clear distinction between price-to-win and realistic estimates in this early phase. According to the interviewed project leaders, this was a reasonable consequence of the very incomplete specifications. The project leaders believed that the pre-planning effort estimate may have some impact on the detailed planning effort estimate, but were unable to indicate how much.

The interviews with the graphical designers and the HCI developer indicated that if the customer or competitors cut the costs but tried to keep the functionality, the proportion of effort used on graphical design and HCI was reduced more than the proportion of effort on programming related activities. In spite of this, the effort estimates of graphical design and HCI were assessed to be more "accurate" than the programming effort estimates. The graphical designers and HCI developer explained that the actual effort was very much determined by the available effort, i.e., the estimation accuracy was not really an indicator of the quality of the original effort estimate. A statement from a graphical designer was:

We get no real requirements specification. The effort is determined by how important graphical design is for the customer. A graphical design may take 5 hours or 5 months.

The interviewed HCI -developer had a similar statement:

The HCI solution may be delivered with very low effort, but then the risk of low quality gets higher.

Not only the graphical design and HCI activities were flexible. A software developer stated that:

If we find in the detailed requirements specification phase that the effort estimates have been too low, we simplify the software solution.

According to the project leaders, projects paid per work-hour seldom or never used less effort than estimated. One reason for this was, as stated by one of the project leaders:

Extra budget time is used to improve the delivered product.

The interviewed project leaders and developers were, to some extent, aware of the potential impact of the effort estimate on the project planning and management. However, when assessing their own estimation accuracy they did not adjust for these impacts. Consequently, they interpreted estimation accuracy as a function of both how good the initial effort estimates were and how well they managed to adjust the work to fit the estimate, e.g., simplifying the solution to avoid exceeding the estimates. The programmers assessed their task effort estimates to be, on average, about 30% lower than the actual effort. These estimates were based on expert guesses. None of the interviewed people had any training or education in estimation.

The organisation also carried out enhancement projects paid per work-hour, with emphasis on product quality and well specified requirements. The effort estimates in these projects had, according to the developers, only minor impact on the actual effort.

3 Study of a Telecom Development Organisation

We studied an organisation that is a department of a large telecom company. The studied organisation has more than 1000 employees. The organisation has over a long period implemented well-defined processes and has been assessed to be at CMM [9] level 2 and aspires to CMM-level 3. The organisation carries out few, but large projects, involving both hardware and software development. The organisation has recently implemented an experience database containing information about completed software development projects. An important purpose of this database is to improve software effort estimation. One of the authors of this paper was involved in this work and, in that context, we analysed final project reports summarising project estimates, facts and lessons learned of the eight last projects of the organisation. A final report, following an organisation specific template, was mandatory for all projects in the organisation for the purpose of learning from experience. To better understand the final reports and the estimation processes in the organisation, we interviewed six of the most senior project leaders. The analysed projects used between 6.000 and 55.000 work-hours. The organisation's own statistics on estimation accuracy showed that the projects varied from almost no deviation to almost 100% more work-hours used than estimated. On average, the deviation between actual and estimated effort (as estimated at the end of the analysis phase) was about 25%. However, a small deviation between estimated and actual effort did not necessarily mean that the original estimate had had a high quality. For example:

- Project A started, according to the project leader, without competent developers and changed project scope in the middle of the project period. This situation would indicate a high risk of inaccurate estimates. However, the actual effort was very close (4% higher) to the estimated effort. From the final report and interviews there is evidence of a significant cut of functionality. This cut was possible because the requirements specification was at a very high level. The quality of the outcome was, according to the project leader, much lower than planned. Nevertheless, the solution was accepted by the customers.
- Project B completed with less functionality than specified and too low quality (according to the project leader of a subsequent project using the software produced by B). The actual effort on this project was only 10% higher than the estimated effort. Project B had a priority on cost. If the quality should have been as expected by the users of the project outcome, then the project would have had to use significantly more effort.
- Project C had the second highest deviation between estimated and actual effort (about 30% higher actual effort). This project had, according to the project leader, the following priority: 1. Lead-time, 2. Quality, 3. Cost, i.e., cost had the lowest priority. The time available was, according to the project leader, very short and not in accordance with the wanted quality, nor the amount of functionality. Nevertheless, the project delivered only one month late and with a fault density within the quality requirements. Project C is an example of a software project optimising project behaviour on lead time on the cost of effort.
- Project D had the highest deviation of the studied projects with the actual effort twice as high as the estimated effort. This project had quality as priority one. The reasons for the deviation were, according to the project leader, much too low estimates on firmware components and some major change requests during the project execution. It seems that the high quality requirements on the product prevented a high impact of the estimated effort on the actual effort in this project. The project data showed no evidence of process or product shortcuts.

These four projects illustrate the difficulties of interpreting the organisation's own estimation accuracy values as indicators of the quality of the original estimate. Estimation accuracy, as the organisation measure it, is a mix of how well the project leader is able to manage the project to not exceed the estimate and the realism of the original estimate.

Typically, the studied projects followed the effort estimates on the requirements specification, design, administration and reviews. Programming and, in particular, testing were on the other hand frequently underestimated. This may indicate a situation where the project makes process shortcuts in the early phases, to keep the activity estimates, on the costs on rework in the programming and testing phases. How much a too low effort on requirements specification, design, administration and reviews increases the effort on programming rework and testing seems, therefore, to depend on the project priorities regarding quality.

4 Experiment on the Impact of Pre-planning Estimates on Detailed Planning Estimates

There were several indications from the case studies that the pre-planning effort estimate strongly impacted the detailed planning effort estimate. We got the impression, in particular in the first case study, that a rather arbitrary early effort estimate impacted the detailed planning estimates even when it was known that the early effort estimate was unrealistic. We decided to test experimentally the following hypothesis:

Hypothesis 1: Pre-planning effort estimates have a major impact on the detailed planning effort estimates. This impact is present even when the estimators are told that the pre-planning effort estimates are not based on historical data or expert knowledge.

The subjects were 38 under-graduate computer science students at the University of Oslo attending a course in software engineering. The course was given by the authors of this paper. The experiment was carried out in a lecture about software effort estimation at the beginning of the course. To make the estimates as realistic as possible, the students were supposed to estimate own work. The experiment was carried out as follows:

1. **Task**: The students should estimate how much effort they would use on the course in software engineering. The activities to be estimated were: Attending the lectures, attending the weekly exercises, work on the mandatory software development project and self study on course topics. The students could use the activity category "Other activities" if an activity did not fit any of the categories above.
2. **Pre-planning estimate**: Before estimating the course effort per activity the students were asked whether they thought they would use more than X work-hours on the course. Half of the students were presented with X=200 and the other half with X=800. The course gives four credits, which correspond to 40% of full time study load. While 200 work-hours was a realistic effort usage, 800 work-hours would mean that the students would use about 50 hours each week on the course, i.e., a much too high effort estimate. The students were asked not to use more than 5 minutes answering this question. They were informed (in bold letters) that the X-value was not based on historical data or what the course teachers believed was a reasonable amount of effort. The X-value was intended to be similar to a pre-planning software project effort estimate based on limited knowledge.
3. **Detailed estimate**: Then, the students were asked to provide the minimum effort, the most likely effort and the maximum effort of each course activity, i.e., a description of the distribution of possible effort usage. The students were instructed that there should be approximately 90% probable that the actual effort usage would be within the prediction interval [minimum effort, maximum effort]. The estimated effort was then calculated applying the estimation formula in Section 1.2. The total effort estimate is calculated as the sum of the estimate of each activity. We calculated the approximate 90% prediction interval for the total

effort as the sum of the prediction intervals of each activity, i.e., we calculated the sum of all maximum values and the sum of all minimum values. An exact calculation of the 90% prediction interval of the total effort requires a complex addition of probability distributions and assumptions about the dependencies between the distributions. This complex addition is unnecessary in this study since the summation error we make, e.g., due to dependencies between the maximum values of different activities, will be approximately the same for all participants.

4. **Analysis**: The mean estimated effort for the groups with X=800 and X=200 were compared.

The resulting total effort estimates are depicted in Figure 1. Table 2 gives the p-values of the one sided t-tests of differences in mean values between the X-groups for the total estimate and for each major sub-activity assuming different variance. The effort estimates of each group are approximately normally distributed, i.e., a t-test on difference in mean values is appropriate. Instead of a pre-determined p-value for rejecting no difference in mean values, we will present the actual p-values for each test in the analyses as suggested in, e.g., [10]. To test the size of effect, we use Cohen's size of effect measure (d) [11], which is defined as follows:

d = (mean value group 1 – mean value group 2) / pooled standard deviation amongst groups 1 and 2

Conventionally, a d-value of about 0.2 indicates a small size of effect, a d-value of about 0.5 indicates a medium size of effect and a d-value of about 0.8 or more indicates a large size of effect [11].

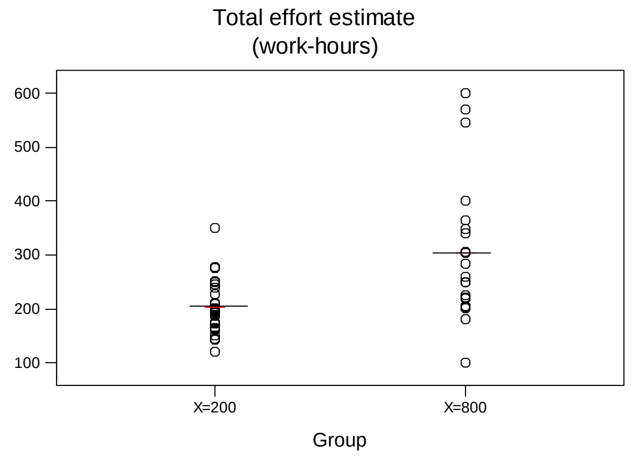

Figure 1 – Total effort estimates by group
(observations are indicated by circles, mean total effort estimates by lines)

Activity	**Group mean X=200**	**Group mean X=800**	**t-test**	**Size of effect (Cohen's d)**
Total	202	303	p=0.003	1.1
Lectures	37	46	p=0.011	0.77
Exercise	25	32	p=0.003	0.95
Project work	73	116	p=0.006	0.87
Self-study	56	93	p=0.006	0.88
Other activities	9	14	p=0.20	0.27

Table 2 – T-tests of differences in mean effort

To our surprise the differences in mean effort of *all* activities, except “Other activities”, were significant (p-values ≤ 0.011) and large ($d \geq 0.77$). This includes activities that we believed were rather predictable, such as attendance on lectures. This shows that the pre-planning estimate (the X-value) had a significant impact on subsequent effort estimate, that is, hypothesis 1 is supported by this experiment.

Another interesting impact of the different X-values was the difference in the relative width (RWidth) of the prediction interval. A high RWidth value indicates a higher degree of uncertainty of the estimate compared with a low RWidth. RWidth is used in several other studies on prediction intervals, e.g., [12], and is defined as:

RWidth = (maximum effort – minimum effort) / most likely effort

There was a significant difference (p=0.032, one-sided t-test, assuming different variance) in RWidth between the two X-groups, see Figure 2. The mean value for X=200 was 0.50 and the mean value for X=800 was 0.68. A potential reason for this difference is that 800 work-hours was further from a realistic use of effort than 200 work-hours. While the X=200 group, on average, estimated effort use very close to 200 work-hours, the X=800 estimated, on average, about 300 work-hours, i.e., far from 800 work-hours. The large difference between 800 and 300 work-hours may have impacted the uncertainty judgements of the students. The X-value may, for this reason, not only have impacted the effort estimate, but also the judged uncertainty of the estimate.

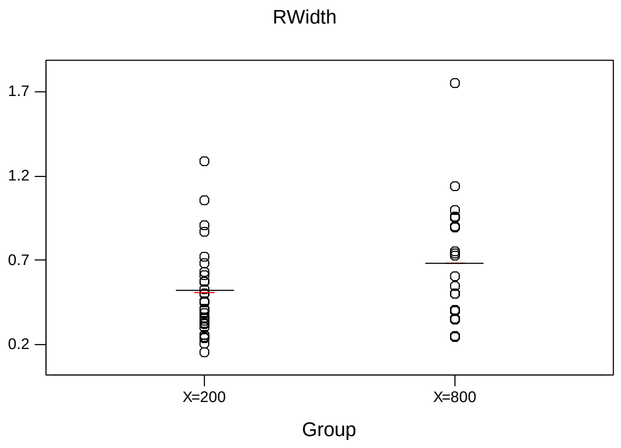

Figure 2 – RWidth by group
(observations are indicated by circles, mean RWidth by lines)

To test whether the estimators believed themselves that they were impacted by the X-value, we carried out a similar experiment at a software development course given by Hedmark College, Norway. This course was in work-load similar to the course in the previous experiment. Here, we added a third question:

In the first task it was stated that the value X hours was not based on historical data or what the course teachers believed was a reasonable amount of effort. Do you, nevertheless, believe that this value has impacted your estimates?

In this experiment we used the X-values: 50 and 500, i.e., a very low and a very high value. A likely effort usage on this course would be, we believed, 100–200 work-hours. The students were only asked to estimate the most likely use of effort, i.e., they were not asked to provide the minimum and maximum values of the effort estimates. The experiment gave results similar to the previous experiment. The mean value for X=50 was 117 work-hours; for X=500 it was 225 work-hours. A one-sided t-test of the difference, assuming different variance, gave p=0.06 and d=0.95. Less than half of the students (7 of 20) believed that they had been impacted by the X-value, i.e., they answered NO on the third question. This low number is, in our opinion, surprising since the wording in our question, 'do you, nevertheless, believe that this value has impacted your estimates?', may have been leading. In addition to the YES/NO-answer on question three, the estimators were asked to describe in which way they had been impacted, or why they believed they had not been impacted. These answers are hard to summarise, but indicated that even those who believed that they could have been impacted had only a vague idea of how much and why. A typical answer was:

Yes, the 500-value may have had an impact. I felt that my own estimates were too low.

This answer indicates, we believe, that the estimator felt a pressure to increase his estimate by the high X-value, in spite of the fact that the X-value was far too high to be realistic.

5 Experiment on the Impact of Too Low Effort Estimates

The case studies indicated that a too low effort estimate put a high pressure on the project leader and software developer to reduce the use of effort and that this pressure could, for example, lead to less emphasis on design and testing. We conducted an experiment to test the impact of this pressure from the effort estimate and made the following hypothesis:

Hypothesis 2: Too low effort estimates lead to less use of effort and more errors compared with more realistic effort estimates.

This hypothesis was tested using computer science students attending a graduate course on software process improvement at the University of Oslo. The experiment was conducted as follows:

1. Three small programming task specifications (T1–T3) were developed. T1–T3 were similar in size and complexity. All tasks involved the programming of a control-flow rich program. The programs should be written in a Pascal similar syntax, using pen and paper only.
2. The students solved the first task (T1), which was a calibration task, with no effort estimate. Based on the information from solving T1, we developed a simple estimation model predicting the effort on the second and third programming tasks (T2 and T3). In the solving of T1, 12 students participated.
3. The students were divided into two groups (G1 and G2) with similar characteristics regarding mean development effort *based on the performance of the first task*, i.e., we ensured that there was no large difference between the two groups.
4. In the following lecture, one week later, the students were told to solve the two remaining programming tasks (T2 and T3). Two of the students in G2 had not the opportunity to attend that lecture. For this reason, G1 and G2 had not the same size. G1 contained 6 participants (P1–P6); G2 only 4 (P7–P10). The students were given the task specifications and the estimated effort to solve each of the tasks. While one of the tasks had a realistic estimate based on the estimation model, the other task had a very optimistic estimate, see Table 3. The optimistic estimate was the estimate derived from the estimation model reduced with 40%. The purpose of the somewhat arbitrary 40% reduction was to provide an estimate that was sufficiently small to put a pressure on the programming without being too unrealistic. The students were not told that one of the estimates was very optimistic; they believed that both estimates were realistic.[1] We asked the students to behave as closely as possible to how they

[1] There are ethical concerns regarding this design. However, we assumed that the learning effects would be significant for both groups of students and the negative impacts low. We used several hours to discuss the design and results with the students after the experiment. There were no negative reactions on the fact that they had not been informed about the optimism of one of the estimates. Nevertheless, some of the students felt a higher performance pressure (got more stressed) than we expected. We expected a low pressure since the performance on these tasks was irrelevant for the students' course grading. One should be very careful in using such experiments where the participants are not informed

would have behaved if this were a regular task that they would have had to carry out as employees in a software development organisation. We did not tell them to try to keep the effort estimates given by the effort estimation model.

Group	**Task 2**	**Task 3**
G1	Realistic estimate	40% too low estimate
G2	40% too low estimate	Realistic estimate

Table 3 – Task and estimate assignments

5. The students decided themselves when the tasks were completed. On completion we registered the effort used and inspected the solutions to assess the quality in number of errors. We counted all types of logical errors, but not syntactical errors that a compiler would have discovered.

The few observations did not enable meaningful analyses of the distribution of the use of effort. Instead of using a parametric test on the effort data, we followed the recommendations in [13] and used the Wilcoxon signed ranks test. This test is useful when there are matched pairs, e.g., a between subject research design, and the relative magnitude of the difference in rank values is meaningful. To use the Wilcoxon signed ranks test, we ranked the participants effort use of each task and calculated the difference of a participant's effort rank on the task with realistic effort estimates with the rank on the task with too low effort estimates. We ranked the effort instead of using the actual effort values, because the two tasks differed somewhat in average use of effort. If the too low effort estimate had no impact on the use of effort, there should, on average, be no significant difference between the sum of the ranks of tasks with realistic estimates and task with too low estimates. The Wilcoxon signed ranks test on a difference in rank values gave p=0.09. Table 4 shows the differences in rank values. Four of the participants improved their rank (P3, P5, P7, P10), four had no change in rank (P1, P4, P8, P9) and two had worse rank (P2, P6) on the task with too low effort estimates.

Participant	Task with realistic estimate	Task with 40% too low estimate	Effort rank on task with realistic estimate	Effort rank on task with 40% too low estimate	Difference in ranks
P1	Task 2	Task 3	10	10	0
P2	Task 2	Task 3	7	8	-1
P3	Task 2	Task 3	7	4	3
P4	Task 2	Task 3	3	3	0
P5	Task 2	Task 3	9	5	4
P6	Task 2	Task 3	5	7	-2
P7	Task 3	Task 2	9	6	3
P8	Task 3	Task 2	2	2	0
P9	Task 3	Task 2	1	1	0
P10	Task 3	Task 2	5	3	2

Table 4 – Effort ranks

Applying the Wilcoxon signed ranks test on the error data gave p=0.13. Three participants increased the number of errors (P7, P8, P10), seven had no change (P1, P2, P3, P4, P5, P6, P9) and none made less errors on the task with too optimistic effort estimate. Five participants made no errors on neither tasks.

about the real purpose. In this case, we considered the potential harm to be low in comparison with the benefits. Through the participation in the experiment, the students gained valuable experience in how a too low estimate impacted their own work.

The low number of participants makes the interpretation of the p-values difficult. We interpret the results as a weak indication of a relationship between too low effort estimates and a pressure on the software work that, for small programming tasks, can lead to less use of effort on the cost of more errors. Clearly, larger, more realistic studies with professional software developers are needed to validate the size and significance of that relationship.

6 Related Studies

While a high number of software effort estimation models have been developed, see [14] for an overview, there are few studies on how effort estimates impact the project behaviour. Most of these studies are found in the system dynamics literature. A summary of the relevant system dynamic studies can be found in [5], p 29– 54. In particular, the studies described in [4, 5] are relevant for this study. In [5] a simulation model of the dynamics of effort and schedule estimation is developed and evaluated in a NASA project. The simulation model illustrates several impacts similar to those we describe, such as:

- Different estimates create different projects.
- The most accurate estimate is not necessarily the best estimate. A strategy to achieve very accurate estimates is to add a very large risk buffer and use the additional effort to "gold-plate" the product. This strategy may, on the other hand, lead to very low productivity.

Similar findings are reported in [15-17].

Our experiment described in Section 4 was inspired by the human judgement experiments on judgement anchoring, e.g., [18, 19]. These anchoring experiments show that people's judgements are very much impacted by the anchor (the initial value or starting point). An example of an experiment on anchoring is described in [19, p 14] where subjects were asked to estimate the percentage of African countries in the United Nations (UN). Before estimating, the subjects watched a "wheel of fortune" randomly stop at a number between 0 and 100. The subjects were first instructed to indicate whether that number was higher or lower than the percentage of African countries in UN, and then to estimate the percentage. The randomly generated number had an amazingly large impact on the estimated percentage. For example, the median estimates of the percentage of African countries in the UN were respectively 25 and 45 for groups that received 10 and 65 as starting points! Based on these findings the results of our experiment on the impact of the pre-planning estimate should be no large surprise.

Including the results from studies on human judgement and system dynamics in the software project literature, we may improve the understanding of why the impact of the effort estimates on the software work can be so large and maybe avoid some of the unwanted impacts.

7 Threats to Validity

7.1 The Case Studies

The results in the case studies strongly rely on statements in the interviews and the final reports. For this reason a major threat to the validity of our study is biased information from project leaders and software developers in the organisations. In addition, as researchers, we have biases when selecting and interpreting information. The discussion below focuses on the potential biases of the project leaders and developers, but most of this discussion is also relevant biases for us as researchers.

- **Biased perception**: The same events are perceived differently by different people. Studies show that what we perceive is heavily influenced by what we expect and wish to find [20, 21]. For example, a study [22] of how different football supporters perceived a football game concluded: *"It seems clear that the 'game' actually was many different games. ... It is* *inaccurate and misleading to say that different people have different 'attitudes' concerning the same 'thing'. For the 'thing' simply is not the same for different people whether the 'thing' is a football game, a presidential candidate, communism, or spinach.*" An example from our work is that the difference in how project leaders perceived the quality of a delivery depended very much on whether they were responsible for the delivery. We have tried to reduce the probability of perception bias through multiple data sources and supplementing experiments of important relationships, but cannot remove this source of error completely.

- **Biased selection**: Both the project leaders and we selected, to some extent, the project events and characteristics based on our beliefs of what we believed was important. These selections may be biased. For example, while the project leaders in the final reports, typically, discuss in-depth why they used *more* effort than estimated (creeping functionality, poor sub-contractors, etc.), the discussions about why they sometimes used *less* effort than estimated on some activities were very shallow. This lack of emphasis on activities where the project performed better than expected seems to be a pattern of most project final reports we have read.
- **Biased interpretation**: It is difficult to be objective and unbiased when interpreting events, statements, opinions and measured values. For example, if a project leader strongly believes that an incomplete requirements specification is the main reason for cost overruns, this will impact how project events are interpreted.
- **Hindsight bias**: When people know the actual outcome of a process, they tend to regard that outcome as having been fairly predictable all along – or at least more predictable than they would have judged before knowing the outcome [23]. Experience from software projects collected after completion can be strongly impacted by hindsight bias ("I-knew-it-from-the-beginning ..."). Unfortunately, it is not enough to inform people about hindsight bias and encourage them to avoid it [24]. There are several experiences potentially based on hindsight bias in our study. For example, one of the project leaders in the final reports describes that the project time constraints were impossible to meet and that this was known from the beginning of the projects. It is hard to know how clear this insight actually was at the beginning of the project, and how much that can be described as hindsight bias.

7.2 The Experiments

Our experiments were carried out in controlled situations on the cost of realism. We believe that the main limitations regarding realism of our experiments are:

- **Use of students**: Students have limited industrial experience and may behave differently from software professionals regarding effort estimation impacts. Arguments in favour of a not too large difference between students and software professionals in our experiments are:
  - In our interviews with the professional developers and project leaders we found that they had not been taught how to estimate and that they did not get proper estimation accuracy feedback. This means there may not be a very large difference regarding estimation skill between students and software professionals. As reported in [25], the estimation skill may not improve very much with increased experience when there is no proper learning environment.
  - The students in the experiment described in Section 4 should estimate their own work. In this situation the student is a domain expert, i.e., it is a situation similar to the one a programmer faces when estimating a programming task where he/she is a domain (programming) expert, but no expert in effort estimation.
  - Most of the participants in the experiment described in Section 5 had several years of experience from software organisations.
  - Results in [26] indicate no large differences between students and software professionals regarding assessment of lead-time impact.
- **Lack of realism**: Although we tried to make the experimental situations realistic, we cannot be sure that the experimental situation had no unwanted impacts on the estimation or development process. For example, the students may have ignored the instruction that the pre-planning effort estimate was not based on historical data or expert knowledge because it came

from their lecturer, not because pre-planning effort estimate functioned as an anchor value. The development tasks in the second experiment were much smaller than most real-life tasks. Therefore, the results from that experiment are mainly valid for small programming tasks. Large-scale experiments in more realistic environments are needed.
- **Low number of subjects**: Both experiments, but in particular the second experiment, would have benefited from more subjects. Replications of the experiments are recommended to evaluate the validity of the results. Section 1.1 provides a link to the experiment material necessary for replications.

8 Conclusions and Further Work

The studies in this paper show that:
- Pre-planning effort estimates can have a major impact on the detailed planning effort estimate. One of the experiments indicate that this impact is present even when the estimators are told that the early estimates are not based on historical data or expert knowledge. The awareness of the size of this impact seems low.
- Effort estimates can have a strong impact on the project work. The size and nature of this impact depend, among other variables, on project priorities and completeness of requirements specification. For example, we found that projects with high priority on costs and incomplete requirements specifications were prone to adjust the work to fit the estimate when the estimates were too optimistic, while too optimistic estimates led to effort overruns for projects with high priority on quality and well specified requirements.
- The impact of the estimate on the project work, combined with the lack of a common interpretation of 'estimate' in the studied organisations, imply that it is not trivial to compare the quality of effort estimates of different software projects. A meaningful interpretation of projects' estimation accuracy requires knowledge about the size and nature of the impacts of the effort estimates on software work.

We believe that our combination of case studies and experiments was useful. The case studies were strong on realism and weak on control of variables, while the experiments were weak on realism and strong on control of variables. In other words, the case studies and experiments complement each other.

We will in our further work focus on:
- replication of the experiments with software professionals,
- better approaches for the comparison of effort prediction accuracy and
- development of estimation processes that avoid the impact of early, very inaccurate effort estimate on the subsequent effort estimates.

Acknowledgements

We gratefully acknowledge the collaboration with our industrial partners enabled through the research project PROFIT (PROcess improvement For the IT industry) funded by The Research Council of Norway, and the valuable comments from the anonymous reviewers of the first version of this paper.

REFERENCES

[1] C. N. Parkinson, *Parkinsons's law or the pursuit of progress*. London: John Murray, 1947.
[2] I. Sommerville, *Software Engineering*, 6 ed., Essex, England: Addison Wesly, 1996.
[3] K. Sengupta and T. K. Abdel-Hamid, “The impact of unreliable information on the management of software projects: a dynamic decision perspective,” *IEEE Transaction on systems, man, and cybernetics*, vol. 26, pp. 177-189, 1996.

[4] T. K. Abdel-Hamid and S. E. Madnick, “Impact of schedule estimation on software project behavior,” *IEEE Software*, vol. 4, pp. 69-75, 1986.

[5] T. K. Abdel-Hamid and S. E. Madnick, *Software project dynamics. An integrated approach*. New Jersey: Prentice Hall, 1991.

[6] J. J. Moder, C. R. Phillips, and E. W. Davis, *Project management with CPM, PERT and precedence diagramming*. Wisconsin, U.S.A: Blitz Publishing Company, 1995.

[7] S. McConnel, *Software project survival guide*: Microsoft Press, 1998.

[8] S. D. Conte, H. E. Dunsmore, and V. Y. Shen, *Software engineering metrics and models*: Benjamin/Cummings Publishing Company Inc., 1986.

[9] M. C. Paulk, B. Curtis, M. B. Chrissis, and C. V. Weber, “Capability maturity model, Version 1.1,” *IEEE Software*, vol. 10, pp. 636-651, 1993.

[10] T. H. Wonnacott and R. J. Wonnacott, *Introductory statistics*, 5th ed: John Wiley & Sons, 1990.

[11] J. Cohen, *Statistical power analysis for the behavioral sciences (Rev. ed)*. New York: Academic Press, 1977.

[12] I. Yaniv and D. P. Foster, “Precision and accuracy of judgmental estimation,” *Journal of behavioral decision making*, vol. 10, pp. 21-32, 1997.

[13] S. Siegel and N. J. Castellan Jr, *Non-parametric Statistics for the Behavioral Sciences*, 2nd Edition ed: McGraw Hill College Div, 1988.

[14] L. C. Briand, K. El Emam, D. Surmann, I. Wieczorek, and K. D. Maxwell, “An assessment and comparison of common software cost estimation techniques,” ISERN 98-27, 1998.

[15] F. Calzolari, P. Tonella, and G. Antoniol, “Dynamic model for maintenance and testing effort,” presented at Int. conf. on software maintenance, 1998.

[16] M. R. Vidger and A. W. Kark, “Software Cost Estimation and Control,” National Research Council of Canada, Ottaws, Ontario, Canda NRC No 37 116, 1994.

[17] A. G. Rodrigues and T. M. Williams, “System Dynamics in Project Management: Assessing the Impacts of Client Behaviour on Project Performance,” presented at International System Dynamics Conference, Boston, MIT, 1996.

[18] P. Slovic and S. Lichtenstein, “Comparison of Bayesian and regression approaches to the study of information processing in judgment,” *Organizational Behavior and Human Performance*, vol. 6, pp. 649-744, 1971.

[19] D. Kahnemann, P. Slovic, and A. Tversky, *Judgement under uncertainty: Heuristics and biases*: Cambridge University Press, 1982.

[20] R. P. Vallone, L. Ross, and M. R. Lepper, “The hostile media phenomenon: Biased perception and perceptions of media bias in coverage of the Beirut massacre.,” *Journal of Personality and Social Psychology*, vol. 49, pp. 577-585, 1985.

[21] S. Plous, *The psychology of judgment and decision making*: McGraw-Hill, 1993.

[22] A. H. Hastorf and H. Cantril, “They saw a game: A case study,” *Journal of Abnormal and Social Psychology*, vol. 49, pp. 129-134, 1954.

[23] M. R. Leary, “Hindsight distortion and the 1980 presidential election.,” *Personality and Social Psychology*, vol. 8, pp. 257-263, 1982.

[24] B. Fischhoff, “Perceived informativeness of facts,” *Journal of Experimental Psychology: Human Perception and Performance*, vol. 3, pp. 349-358, 1977.

[25] M. Jørgensen, D. Sjøberg, and G. Kirkebøen, “The Prediction Ability of Experienced Software Maintainers,” presented at 4th European conference on software maintenance and reengineering, Zürich, Switzerland, 2000.

[26] M. Höst, B. Regnell, and C. Wohlin, “Using students as subjects - a comparative study of students and professionals in lead-time impact assessment,” *Empirical Software Engineering*, vol. 5, pp. 201-214, 2000.