

# Software Development Effort Estimation: Demystifying and Improving Expert Estimation

Magne Jørgensen, Stein Grimstad

## Abstract

**Research background:** Inaccurate estimation of software development effort is one of the most important reasons of IT-project failures. While too low effort estimates may lead to project management problems, delayed deliveries, budget overruns and low software quality, too high effort estimates may lead to lost business opportunities and inefficient use of resources. These IT-project problems motivated the BEST (Better Estimation of Software Tasks) project at Simula Research Laboratory to conduct research with the goal of improving effort estimation methods.

**Scientific challenges:** The project's main focus is to improve judgment-based effort estimation (expert estimation), which is the estimation approach most frequently used by the software industry. We argue that a better understanding of the mental steps involved in expert estimation is necessary to achieve robust improvement of software development estimation processes. A great challenge when studying expert judgment is to understand the unconscious steps involved. To study these steps, we make use of multidisciplinary competencies, especially psychology and software engineering, and financial resources enabling studies in realistic software development effort estimation contexts.

**Obtained and expected results:** In this section, we present the main results of the project from its beginnings in 2001 until today and describe how the project has benefited from being an integrated part of Simula Research Laboratory, especially through the

opportunity to conduct large-scale, controlled experiments in field settings. The section includes practical results on how to improve estimation methods, scientific results leading to a better understanding of the mental processes involved in judgment-based effort estimation, and our innovative research methods in the field of empirical software engineering. Results of the BEST project are currently in use by several software companies, as well as by researchers in forecasting and psychology, and they are also included in project management and software engineering textbooks. The BEST project plans to continue its multidisciplinary effort with the goal of constructing and evaluating better models of the mental steps involved in judgment-based effort estimation. An improved model of these steps will enhance our ability to accurately estimate software development effort and to predict when different types of estimation methods can be expected to deliver accurate effort estimates. This will in turn lead to a reduction of the number of IT-project failures and make better use of scarce financial and human resources.

## **1. Motivation**

### **1.1 Industrial Relevance**

The main determinant of many types of software-related investments is the amount of development effort required. The ability of software clients to make investment decisions based on cost estimates is consequently strongly tied to the software providers' ability to estimate the effort accurately. Similarly, the ability of project managers to plan a project and ensure efficient development frequently depends on accurate effort estimates. The importance of accurate effort estimates is illustrated by the findings of a 2007 survey of more than 1,000 IT professionals.<sup>1</sup> The survey reports that two out of the three-most-important causes of IT project failure were related to poor resource estimation, that is, inaccurate effort estimates.

The survey response is understandable. A review of estimation accuracy studies [1] reports that software projects expend, on average, 30%-40% more effort than is estimated. Software projects experience severe delivery and management problems when plans are based on overoptimistic effort estimates. The negative effects of overoptimism are accentuated by 1) software bidding rounds in which those companies that provide the overoptimistic effort estimates are more likely to be selected (the 'winner's curse' effect, see [2]), and 2) strong overconfidence in the accuracy of the estimates—for example, 90% confidence intervals (minimum-maximum intervals) of effort only include the actual effort 60%-70% of the time [3]. Not only can effort underestimates lead to problems with software projects (both for the provider and the client), but effort overestimates are also problematic. Overestimates may, for example, lead a provider to lose bidding rounds and clients not to start a project that would have been a good investment. Solving the problems related to inaccurate effort estimates would clearly improve the use of scarce financial and human software development resources.

---

<sup>1</sup> See: <http://certification.comptia.org/project>

While we cannot expect zero estimation error, due to the high complexity and inherent uncertainty of many software projects, there are reasons to believe that the estimation accuracy improvement potential is high. Reasons for believing this include the current strong bias towards overoptimism (cost overruns are far more typical than cost underruns), the high degree of inconsistency in the estimates (effort estimates for the same work can vary greatly, even when repeatedly estimated by the same developers), the limited collection and use of historical data (poor track record of learning from experience), and the lack of evidence-based selection of estimation methods. We discuss our results related to these reasons in more detail in the results section of this section.

## **1.2 Previous Research**

Software engineering researchers have been addressing the problems of inaccurate effort estimation in software development projects since at least the 1960s; for example, see Farr [4]. Nearly all of that research has focused on the construction of formal software effort estimation models. Model types that have been developed and evaluated include those founded on regression analysis, case-based reasoning, classification and regression trees, simulation, neural networks, Bayesian statistics, lexical analyses of requirement specifications, genetic programming, linear programming, economic production models, soft computing, fuzzy logic modelling, statistical bootstrapping, and combinations of one or more of these models. In a review, we identified 184 journal papers that introduced new variants of formal models for software development effort estimation [5]. This corresponds to a percentage as high as 60% of the total number of journal papers on effort estimation. Several formal estimation models have been included in commercially promoted tools. A survey by Moores and Edwards [6] found, for example, that 61% of the IT managers in the UK had heard about at least one of these tools for software

development effort estimation. The use of formal estimation models is also typically promoted by software process improvement frameworks and in educational readings on software engineering.

Yet in spite of the extensive research into estimation models, the high degree of availability of commercial estimation tools that implement the models, the awareness of these estimation tools, and the promotion of model-based estimation, software engineers typically use judgment-based methods<sup>2</sup> ('expert estimation') to estimate effort [7, 8]. At Simula Research Laboratory, a recent survey by Nils Christian Haugen (work in progress) suggests that the use of formal estimation models has declined, rather than increased, over the last five years.

### 1.3 Why Improve Judgment-based Effort Estimation?

Our research is based on the assumption that research on judgment-based effort estimation processes (sometimes in combination with formal estimation models) would be an efficient use of research resources when used to improve effort estimation accuracy. Reasons in support of this assumption include:

1. Even the use of unstructured and unsupported judgment-based effort estimation seems, on average, to lead to more accurate effort estimates than the use of sophisticated formal models, as reported in our systematic review of empirical studies comparing expert and model estimation accuracy [9]. The introduction of more structure, such as the use of structured group processes and an experience-based estimation checklist, consequently has the potential to substantially improve judgment-based effort estimates (see, for example, [10], for general

---

<sup>2</sup> We find it meaningful to categorize judgment- and model-based effort estimates on the basis of the type of mental process applied in the 'quantification steps' of the estimation work, that is, the steps in which an understanding of the software development estimation problem is translated into quantitative measures of the required effort, such as the number of work-hours most likely needed. If the quantification steps are based on tacit, intuition-based processes, the process is categorized as *judgment-based*. If the quantification steps are mechanical, the process is categorized as *model-based*. There will be a range of different estimation processes belonging to each of the categories, that is, neither expert judgment- nor model-based effort estimation should be considered simply as 'one method'. Software development estimation models typically take expert judgments as input, that is, both estimation approaches depend, to some extent, on subjective input.

forecasting evidence on the benefits of several types of expert judgment process structures and support). In [9], we report that the following two characteristics of the software development context may point to inherent, essential advantages of judgment-based effort estimation compared with model-based effort estimation:

- a. Software development domain experts typically possess highly context-specific information of importance that is not part of the model. This situation has been found to favour judgment-based estimation over model-based estimation [11, 12].
  - b. Essential software development relationships, such as between effort and size, seem to be unstable [13]. As pointed out in [14] judgment-based forecasts worked better in unstable situations, while the models performed better during periods of stability. Software development is characterized by unstable, dynamic conditions with frequent changes of problems to be solved, new teams, new clients, and new tools.
2. In comparison with that conducted on formal effort estimation models, there has so far been very little research on improving judgmental effort estimation processes. As an illustration, we found that only 15% of the journal papers on software development effort estimation analyzed judgment-based effort estimation [5]. Among these papers, we found none that aimed at an *improvement* of judgment-based effort estimation.
  3. Overall, the software industry seems to be more willing to accept and use judgment-based estimation methods. We have, for example, observed repeatedly that software projects that officially apply a formal estimation model, in reality use the model as a disguise for judgment-based estimation [15]. This low acceptance of model output in some contexts is observed in several disciplines and should not be surprising. In [16], for example, analysis (corresponds to model-based estimates) is described to have the characteristics of ‘low confidence in outcome, high confidence in method’, while intuition (corresponds to judgment-based estimates) is described to have the characteristics of high confidence in

outcome, low confidence in method (see also [17] for similar findings). An important reason for the rejection of output from formal software development effort estimation models, in addition to their low accuracy, may be that the experts' highly specific knowledge, for example, specific knowledge about the particular developers who are supposed to do the work, frequently cannot be included properly as model input. Understandably, it is not easy to trust an estimation method that is unable to make use of clearly relevant information and that does not 'feel right'. However, it is hardly possible to unite highly specific knowledge with the general relationships that are required by formal models.

4. It is difficult to avoid relying on software professionals' judgment when estimating effort, even when using estimation models. Improvement of the judgmental processes is consequently of importance in, perhaps, all real-life estimation situations.

A discussion between the first author of this paper and Barry Boehm (perhaps the most well-known estimation model builder) on the advantages and disadvantages of model- and judgment-based effort estimation can be found in [18].

#### **1.4 The Objectives of the BEST Project**

The Better Estimation of Software Tasks (BEST) project, launched at Simula Research Laboratory in 2001, built on a 2000 project started at the University of Oslo. The BEST project had and still has the following three main, interconnected objectives related to judgment-based effort estimation:

**O1: Better Understanding (Theory-building):** The main goal here has been to develop an evidence-based, 'constructive' model (theory) of judgment-based effort estimation. A constructive model in this context should be interpreted as one that is useful for the proposal (construction) of improved judgment-based effort estimation processes and principles. Such a constructive model should, for example, be able to support the design and selection of estimation

methods and predict when an estimation situation is likely to lead to inaccurate effort estimates. Moreover, our model of judgment-based effort estimation should also lead to an improved understanding of quantitative judgment in general and make a positive impact on research in other domains with similar types of judgments.

**O2: Better Practices (Improved estimation guidelines and processes):** The main goal here has been to develop methods and principles that have the potential to generate substantially more accurate effort estimates than currently occurs for software projects. This may include the construction of new methods for effort estimation, methods that combine the advantages of models and judgments, and methods for the improved selection of estimation methods. The methods and principles should, preferably, be based on robust theory on human judgment.

**O3: Substantial Industry Impact (Transfer of results):** The main goal here has been to disseminate improved, validated in-the-field, estimation guidelines and processes to the software industry. The methods and principles should be in a format that makes them adaptable to a variety of industrial contexts.

## **2. Research Results**

This section contains brief descriptions of what we consider our main research contributions on judgment-based effort estimation. The results are organized by the following topics: Improved knowledge about the state-of-practice (Section 2.1), Improved understanding of the mental steps involved (Section 2.2), Improved estimation processes (Section 2.3), and Improved research methodology (Section 2.4). The results presented in Sections 2.1 and 2.4 provide an important basis for achieving the three objectives (O1, O2, and O3) of the BEST project. Section 2.2 presents results with a focus on O1; and Section 2.3, results with a focus on O2. The results related to O3 are presented in Section 3. The actual studies and additional details about the results and proposed estimation methods can be found in the referenced studies.

We have not included our results on formal estimation models in this paper. Several of these results, however, particularly the uncertainty assessment model proposed in [19] and the regression-towards-mean adjustment of analogy-based models proposed in [20], have been subject to subsequent study by other researchers and/or inclusion in commercially available estimation tools, that is, they have had an impact on both research and practice. Neither have we included our contributions on Evidence-Based Software Engineering [21-23].

### **2.1 Improved Knowledge about State-of-Practice**

A deep understanding about state-of-practice is necessary to focus the research on essential topics and to comprehend the potential for improvement. Our main contributions documenting the state-of-practice are related to:

1. Surveys of estimation accuracy, estimation processes, and reasons of estimation error.
2. Disclosure of the low level of trustworthiness and the survey-method problems of the most frequently cited report documenting the state-of-practice in estimation work, that is, the Standish Group's CHAOS report.

3. The first documentation of the high level of inconsistency in software professionals' effort estimates.
4. The first documentation of the high level of overconfidence in the accuracy of software professionals' effort estimates.
5. The first in-depth study comparing top-down and bottom-up estimation processes.
6. Studies on group-based and mechanical combination of effort estimation.
7. The first review comparing the estimation accuracy of models and expert judgment.
8. Developing the currently most comprehensive database and review on estimation research.
9. The first field experiment on the (lack of) learning from estimation 'lessons learned' sessions.

Through our own surveys on estimation practices [24] and reviews of other surveys [1], we have documented a 30%-40% estimation error, on average, with no indication of improvement over the years. We also have documented the infrequent use of formal estimation models and the typical reliance on bottom-up, judgment-based effort estimation [25]. We report reasons for estimation error and indicators of high/low estimation error in [26, 27]. Two main contributions from our studies on the reasons for estimation errors are: 1) Current estimation error analysis practices are hampered by too strong a focus on direct reasons, inclusion of limited perspectives, and the application of simplistic cause-effect models, and 2) The best indicator of overoptimistic effort estimates is the previous level of estimator's overoptimism, but even that indicator is not optimal, that is, it seems to be difficult to predict when an effort estimate is overoptimistic. We survey the practice of estimation error analysis in [28] and find that the current practice presents severe shortcomings, for example, there is a lack of precise estimation terminology.

By far, the most frequently cited survey concerning effort estimation accuracy is one referred to in governmental reports as well as research papers. It was conducted by the Standish Group in 1994 and is known as the CHAOS report (<http://www.standishgroup.com/>). It reported an average

cost overrun of 189%. This cost overrun number has been used for several purposes, including excusing poor estimation work. In [29], however, we show that the results reported by the Standish Group are not trustworthy. We argue that the Standish Group's results, which deviate from all other studies on estimation error, are likely to have been caused by a strongly biased sample of projects. More recent reports from the Standish Group, on various topics, seem to exhibit many of the same survey method problems. This illustrates the need to increase awareness of proper survey methodology among researchers and software practitioners.

It is reasonable to expect that the effort estimates of the same project given to the same software professional, with but a few weeks in between, would not be identical; nevertheless, the size of the difference in the estimates (the level of inconsistency) we found turned out to be amazingly large [30]. The median difference between the estimates of the same tasks by the same software professional was as high as 50%. An implication of our study is that low estimation accuracy is, to a substantial extent, caused by a lack of consistency in the effort estimates. These results demonstrate the low robustness of some judgment-based estimation processes, and that increased consistency should be considered as an important method of improving estimation accuracy. The high degree of inconsistency also explains, to some extent, why it is difficult to predict overoptimism, that is, the results reported earlier herein.

Our studies on overconfidence in the accuracy of software development effort estimates are the first in field settings [3]. They document not only the high level of overconfidence—for example, that the figure '90% confident' to include the actual effort in a minimum-maximum interval corresponds to 'hit rates' of 60%-70%—but also demonstrates serious problems with the current guidelines for uncertainty assessment, as prescribed in common textbooks on project management. As an illustration of the problem with current approaches based on effort prediction intervals, we found only minor differences in effort prediction intervals when requesting 99%, 90%, 75%, or 50% confidence to include the actual effort, that is, that the participants provided

almost the same minimum and maximum effort values when they were asked to be 50% certain, compared with 99% certain to include the actual effort. Results from our studies on overconfidence, particularly our results providing new insight into the reasons for the overconfidence, have been included in the paper ‘Judgmental forecasting: A review of progress over the last 25 years’, *International Journal of Forecasting* [31].

Expert estimation processes are frequently described as either top-down or bottom-up, that is, a process in which either the effort estimate is derived from properties of the project as a whole (top-down) or derived from a decomposition of the project into activities (bottom-up). Our video-recording of estimation teams’ discussions when applying these processes resulted in a documentation of the estimation processes that they used along with insight as to when the processes led to the most accurate estimate [32]. The main observation was that the top-down strategy offered the best choice when the estimators brought to bear experience gleaned from a similar project. This new project, however, had to be quite similar for the analogy to be useful. If the software developers could identify projects that were only somewhat similar, then the bottom-up process produced more accurate results. The video-recordings and our results are now subject to further analysis of the interaction processes in estimation teams, as part of a PhD thesis in education and learning.

We have completed several studies on combination-based effort estimation [33-36]. These studies document the benefits of common as well as more innovative combination strategies. Of particular interest are the results on the use of the estimation method commonly applied in agile projects, such as ‘planning poker’. While many studies in other domains observe an increase in risk-willingness in teams (for example, see [37], we did not find this effect in group-based effort estimation. Instead, we found indications of the opposite, that is, that group dynamic effects led to increasingly higher effort estimates. Our study on different strategies of combination of effort prediction intervals [38] is the first in a software engineering context. The results suggest that a

discussion-based combination of prediction intervals should be used instead of mechanical combinations of individual prediction intervals. The overall result of our studies on combination of estimates from different sources, either mechanically or as structured group-discussions, finds that combination of estimates typically improves the average estimation accuracy compared with effort estimation based on one source only. This corresponds to results from other domains.

Our observation (see the review in [9] that formal estimation models did not produce more accurate effort estimates than those based on expert judgment is surprising in light of related research in other domains. A meta-analysis of comparisons of models and judgments is provided by Grove et al. [39] who found that mechanical predictions of human behaviours are equal or superior to clinical prediction methods for a wide range of circumstances. Dawes, Faust, and Meehl [40] emphasize the following two factors that underlie the observed superiority of statistical models: 1) Models are consistent (the same input always leads to the same conclusion), while experts are inconsistent; and 2) Models ensure the contribution that variables make to a conclusion is based on the variables' actual predictive power and relationship to the phenomenon of interest. There may, however, be essential differences between most of the previously studied domains and software development that can explain the lack of advantages to using effort estimation models in software development contexts. As described earlier, there are at least two characteristics of the software development context that may point to inherent, essential advantages of judgment-based effort estimation, that is, that software professionals typically possess highly context-specific information that is not part of the model, and that essential software development relations seem to be unstable. These two advantages may compensate for the possible disadvantages of judgment-based estimates, for example, a higher degree of inconsistency and improper weighting of variables.

As part of building a publicly available database of all software development effort estimation research ([www.simula.no/BESTweb](http://www.simula.no/BESTweb)), we categorized the properties of previous

estimation research [5]. The main contributions of that work are to support researchers with topics for future research and show where relevant research can be found. To date, more than 100 PhD students and researchers have used this database. The feedback we have received indicates that it has saved many of them considerable work, and that the quality of their research has been enhanced through the information provided by our database.

Learning from experience through lessons-learned sessions is a frequently proposed strategy to improve estimation accuracy. We evaluated how effective such lessons learned were and discovered a surprising result: They had no effect on estimation accuracy and effort uncertainty assessment realism [41]. Based on qualitative analysis of the data, we propose that the problem is rooted in the difficulty in assessing how much one has learned from experience. If, for example, a software developer learns from experience, but is overoptimistic about *how much* he or she has learned, the effort estimates will continue to be overoptimistic. We also found evidence suggesting that estimation learning on behalf of other software developers is easier than learning from one's own estimation experience. We plan to work on better estimation learning processes as a follow-up to these, somewhat surprising, results.

## **2.2 Improved Understanding of Judgment-based Effort Estimation**

Robust improvement of effort estimation practices is facilitated by an improved understanding of the mental processes involved in judgment-based effort estimation. Without a comprehensive understanding of why and when different types of judgment-based effort estimation methods can be expected to be accurate, the design and selection of estimation method will prove problematic. Our contributions towards a constructive model of expert judgment include:

1. The first studies documenting the importance of unconscious processes in judgment-based software development effort estimation.

2. The first studies on how much, why, and when irrelevant and misleading information impacts the unconscious processes involved in judgment-based software development effort estimates.
3. The first studies on how the format of requesting effort estimates impacts the level of optimism.
4. The first studies on how software professionals select and change estimation strategy.
5. The first attempt to synthesize existing results from various disciplines into a constructive model of judgment-based effort estimation.

Through video-recording of an estimation team's discussions, use of think-aloud protocols, interviews with software professionals, and studies in which we demonstrate that software professionals do not notice when and how much they are impacted by irrelevant information, we document that essential parts of the estimation processes are based on unconscious processes [32, 42]. An important implication of this finding is that we cannot simply ask software professionals or use think-aloud protocols to examine the real mental processes, but instead will need to rely on controlled experiments.

Another consequence of the unconscious nature of essential steps in judgment-based effort estimation is that the estimates are easily impacted by irrelevant and misleading information. We have conducted more than 30 laboratory and in-the-field experiments documenting this, such as studies on how the estimates and bids are affected by early and underestimates based on limited information [43, 44], clients' unrealistic price expectations or insufficient budgets [45, 46], future, promising opportunities [46], the use of loaded words, such as describing the same task as a major or minor extension [46], and inclusion of text not relevant for the use of effort [46]. Since the software professionals are unaware to what extent they are impacted by misleading or irrelevant information, it is very difficult, if not impossible, to adjust for the impact even when one knows that the information is misleading or irrelevant (for example, see [47]). Our most recent

study on this topic [48] is, as far as we know, the first that compares the effects of misleading and irrelevant information in field settings with corresponding effects in laboratory studies, regardless of domain. This type of field study is, in our opinion, essential to examine the importance—as opposed to merely the existence—of a phenomenon. We find in our field studies, for example, that the impact from some types of irrelevant information, such as the impact from the length of the specification, is present mainly in laboratory settings, while the impact from other types of irrelevant effort information, such as the clients’ unrealistic cost expectations, is lower but still important in field settings.

In a recent study, we examined the importance of the format of the estimation request [49]. In it, we report four experiments suggesting that the request format: ‘How much of Y can you complete spending X work-hours?’, which is a format sometimes used in agile projects and by clients, under certain circumstances leads to more optimistic effort estimates compared with the traditional request: ‘How much effort will it take to complete Y?’ An explanation of this finding, may prove relevant when trying to understand the mental processes involved in judgement-based effort estimates.

It is unlikely that software professionals use only one strategy when estimating effort. Instead, they may rely on a toolbox of strategies and an assessment of the fit of individual strategies to the current situation. Recently, we reported that there is a large individual variation in selection in situations for which there is no information clearly in favour of one particular strategy [50]. We also found that the following three strategy selection factors play important roles in situations for which relevant, historical data are available: individual strategy preference (prior beliefs), estimation surprise (large estimation errors), and aggregated accuracy information of a strategy in the current context. In the study, we found that estimation surprise was frequently required to make the developers change their estimation strategy, even when one strategy was clearly better than another. We also found that the strategy applied toward unrelated tasks

immediately before the estimation work occurred did impact the choice of estimation strategy (a so-called priming effect). For example, the completion of unrelated closest-analogy tasks immediately before led to more use of the closest-analogy strategy in the estimation work.

An important vision of the BEST project has been to synthesize existing results from various disciplines into a constructive model of judgment-based effort estimation and add our own results wherever we find research gaps or a need for more evidence. A constructive model in this case means a model that will be useful for predicting and improving expert judgment-based effort estimates. While we are still some steps away from reaching this goal, we have made a couple of preliminary models [42, 51] and identified research results from other domains, particularly the ‘selective accessibility’ theory presented by [52] and the ‘construal level’ theory presented by [53].

### **2.3 Improved Judgment-based Effort Estimation Processes**

Our research on judgment-based effort estimation has led to several documented improvements of the effort estimation (including bidding and uncertainty assessment) processes. The main process improvements are, in our opinion:

1. Improved estimation terminology.
2. An improved process for analyzing the estimation error.
3. A simple, yet realism-increasing, evidence-based method for the assessment of effort estimation uncertainty.
4. Evidence-based guidelines and principles of judgment-based effort estimation on preparation of estimation information, when to use models, how to combine, how to assess the uncertainty of an effort estimate, how to avoid common estimation biases, and so forth.
5. Guidelines on how to avoid bidding based on overoptimistic estimates.

Starting with our work [54] and further elaborated in [28], we show that the current use of estimation terminology in research papers, software engineering textbooks, and industry practice is confusing and does not enable the types of estimation error analyses we would like to complete. As an illustration, the term ‘effort estimate’ is used, variously, to denote the most likely use of effort, the median effort, the planned effort, or the effort used to price the project. To add to the confusion, this inconsistent use of the term typically appears with no definition of the intended meaning [28, 43]. This lack of precision in the terminology is unfortunate both because a measure of estimation error will be difficult to interpret and there easily could be an unfortunate mix of different concerns in the estimation process. To reduce these estimation terminology problems, the concern when estimating effort should be an assessment of realistic use of *work effort only* (not for example planning or pricing) and the meaning of an estimate should be clearly defined. The estimated effort should then be used as *input* to planned effort (whereby the main concern should be project control and efficiency) and price (whereby the main concern should be profit on short or long term) (see [55]). Not separating realism, planning, and pricing concerns implies that ‘wishful thinking’ may dominate on the cost of realism [56, 57]. Related to the clarification of what is meant by an effort estimate, we propose the use of pX-estimates, where X is the likelihood not to exceed the pX value. For example, the intended meaning of a p50-estimate is that this type of estimate will be exceeded about 50% of the time. This type of probability-based estimation terminology has been proposed earlier, see, for example, [58]. One problem arising with previous proposals, however, has been that they provided no practical way of producing the probability-based estimates. We propose a simple, yet accurate process for this based on a combination of estimation of most likely effort and use of the distribution of estimation error of similar projects. The process has been evaluated with success in several field settings [59, 60]. An important property of the pX-based terminology is that it enables a practical separation of concerns, for example, that the p70-estimate could be the planned effort and the p80-estimate the basis of the budget or price to client.

In several papers, for example, [43, 61, 62], we discuss problems with current processes and measures of estimation error evaluation. We show, among others, how effort estimates impact the actual use of effort and, for this reason, make it difficult to evaluate the estimates' accuracy. There are inherent problems in evaluating estimation error that resist easy solutions, but there also exists the potential for better estimation error measurement and analysis processes. In [62], we summarize our own and other relevant work, and propose a process for improved estimation error measurement and analysis. This method is, we argue, applicable for both researchers and practitioners. We have empirically evaluated the process and found that it improved the usefulness and meaningfulness of the error analysis, and the estimation process improvement work.

Project management and software engineering textbooks instruct that projects provide minimum and maximum effort intervals which include the actual effort with, for example, a likelihood of 98%. The problem, hardly ever mentioned in the project management literature, is that the method leads to much too narrow an interval and, as a consequence, insufficient contingency buffers for the projects and poor project plans. In several studies, we show a process built on the same idea that we use to improve the uncertainty assessment of estimation models as proposed in [19], can be used to improve the uncertainty assessment of judgment-based effort estimates [60, 63, 64]. The simple idea behind the process is to use the error distribution of previous projects as the main input for the prediction intervals of future projects. Our work on this realism-improving principle has attracted attention in both the psychology, and the management and forecasting literature [65-72] and is also accepted as a new forecasting principle at the prime forecasting community publishing their principles at [www.forecastingprinciples.com](http://www.forecastingprinciples.com).

Most of the results of our studies have direct consequences on judgment-based estimation processes. We are in the process of summarizing these results as estimation principles in a textbook, but have also published parts of these results in practitioners' magazines [59] and

presented them at several industrial conferences (see ‘Presentations’ under publications on [www.simula.no](http://www.simula.no)). These include guidelines on the removal of situational and human biases [46], better selection of estimators [73], the use of looking-back strategies to increase realism [63], a combination of estimates from different sources [38], the use of checklists to increase consistency [55], and better provision of training opportunities [25].

Both the software developing organizations and the clients frequently suffer from bids based on overoptimistic effort estimates, for example, financial losses, chaotic projects, low product quality, and loss of market opportunities [44, 74]. Based on available evidence we identify several of the process elements leading to overoptimistic bids; examples are: 1) Invitation of many bidders; 2) Focus on price in the selection of providers; 3) Lack of resources to thoroughly evaluate provider competence; 4) Information about price expectations included in the estimation material; 5) Description of the project as ‘small’; and/or 6) Bidding material with information about future opportunities. Particularly interesting, since it provides input to the understanding of the mental process of judgment-based effort estimation, is that the following bidding process led to 40% lower bids compared with a one-step bidding process: 1) Request of a specification containing more requirements than were actually needed; and 2) Request for bid updates based on a reduced set of requirements [44, 74]. Both studies took place in controlled experiments in field settings and have, we believe, a high external validity. We summarize our results as guidelines for software practitioners and clients in [75].

## **2.4 Improvement of Research Methods**

Our typical research process, following the selection of a research topic, has been as follows: 1) Exploratory studies in the field, for example, surveys documenting the overoptimism, overconfidence bias; 2) Proposal of explanations (theories) on the reasons for these effects, based on previous research from various domains; 3) Laboratory experiments with a focus on validity of

the explanations in software engineering contexts; 4) Field experiments on the robustness and relevance of the explanations; 5) Proposal of changes in estimation processes; 6) Laboratory and field experiments on the effect of the proposed changes in estimation process. Our recommendations on when to use different research methods are, to some extent, described in our paper [76]. The opportunities provided at Simula Research Laboratory, namely, the freedom in the use of large-scale resources, coupled with the opportunity to conduct realistic experiments and work in an environment that stimulates innovation in empirical software engineering methods, have been essential for the described improvements of research methods.

The challenges met when studying judgment-based effort estimation have led us to explore and advance several research methods previously not applied in software engineering research contexts. The main research method innovations, from a software engineering research point of view, are:

1. The first randomized, controlled software engineering experiments in the field.
2. The first use of multicountry populations (outsourcing companies) and studies of regional differences, in controlled software engineering experiments.
3. Extensive use of laboratory experiments to transfer research results to software practitioners and to provide input for our own research at seminars for software practitioners.

In many domains, findings from research based on randomized, controlled experiments is considered to be the most reliable form of scientific evidence [77]. As an illustration, all new medicines and surgical procedures must undergo such trials before being approved. As the name suggests, randomized controlled experiments involve a random allocation of an intervention, for example, a change of estimation practice, to subjects and mechanisms to ensure that the treatment is the cause of the effect. While laboratory studies are useful for many purposes, that is, to document the *existence* of an effect in a controlled environment, controlled field studies are needed to study the *size* of an effect in realistic environments. As documented in, for example,

[78], large estimation effects observed in laboratory settings are in some cases much lower, and not of importance, in real-life settings. Our use of randomized controlled experiments in field settings typically includes the following elements:

- *Design of the study.* The design includes the formulation of a precise and relevant research question. In our case, this research question is typically derived from proposed models of judgment-based effort estimation or from the need to test changed or new estimation methods. The type of data that needs to be collected is determined from the need to test hypotheses related to the research question and to understand the findings.
- *Natural settings.* It is frequently essential that the participating software developers behave as similarly to their normal work behaviour as possible in our studies. This is typically made possible by paying software consultants customary fees for the estimation work. When this is the case, it is possible to create a situation in which the companies are not participating in an experiment, but rather are completing routine paid work for a client.
- *Participant selection.* There are several biases that might ensue as a result of sending invitations to a large set of companies and using only those that offer a positive response as participants in our studies. For example, there might be a bias towards more low-skilled than high-skilled companies responding on the type of estimation tasks we are requesting. To ensure that the estimation skill is acceptable and representative for the type of developer skill we want to study, we typically request curriculum vitas from the software professionals in charge of the estimation work and allow only those who possess an acceptable level of expertise for the purpose of the study to participate. A multicountry population of companies and software developers is sometimes selected to enable more robust, less cultural-dependent results.
- *Randomized treatment.* The estimation material and instructions about estimation processes to be followed are typically presented in more than one version. This enables the testing of

previously defined research questions, for example, a research question related to how a change in the presentation of information or a changed estimation process affect the effort estimates. Typically, a study has a control group representing the default estimation process and one or more treatment groups. The control group will typically receive nonmanipulated estimation material and no estimation instruction or training. The allocation of software professionals and companies to a control group or a treatment group will be random.

- *The estimation context.* To ensure an optimal level of realism, the companies should be asked to both estimate and develop the software work. However, this would make this kind of study extremely costly, and we therefore typically either 1) ask only for the estimation work, or 2) let the estimation work be part of a bidding round in which only one (or a few) of the companies are asked to develop the software. We have experience with both alternatives and find that they represent two different estimation contexts, both of which are realistic and likely to lead to valuable research results.
- *The estimation task.* The software projects to be estimated should be selected to fit the purpose of the study. If, for example, purpose is not related to specification size, then there is no need to specify a very large software system.
- *Monitoring and data collection.* We have experienced that written communication (mainly e-mail) is an efficient way to monitor and document the communication during the estimation work. When communicating by e-mail, one researcher can handle the communication that is necessary when acting as a client of at least 20 software development projects in parallel, provided that he/she has good support from administrative personnel for contractual and financial purposes. It has been essential for the quality of the results that we have had highly qualified personnel to manage and evaluate the deliveries of the software companies.

We are, as far as we know, the first to use outsourcing companies from a variety of countries in both field and laboratory experiments. This includes companies from India, Pakistan, Nepal, Vietnam, Russia, Ukraine, Poland, Romania, Slovakia, Bulgaria, Belarus, Moldova, and Serbia. The use of outsourcing companies has, at least, three advantages: 1) It reduces the costs compared with the use of, for example, Norwegian software companies; 2) The results become more robust compared with single-culture studies; and 3) It enables the study of regional (or even cultural) differences. As an illustration, in an ongoing study, we seem to find that information deemed as intentionally misleading had a much larger impact on software developers in India and Pakistan compared with software developers in Eastern Europe. This suggests that there may be a cultural component related to the perception of irrelevant information that needs to be better understood and/or more fully addressed.

Our laboratory experiments are typically conducted at seminars for software practitioners. A typical laboratory experiment is as follows:

- Formulation of a research question that can be addressed by designing a study in which the software professionals solve small estimation tasks. Preferably, the research topic should be related to the topic at the seminar.
- Design of the experiment. The experiment typically involves a control group and one or more treatment groups.
- Preparation of estimation tasks meaningful to complete in, for example, 5-15 minutes time. This may restrict the estimation tasks to be based on the development of very small programs or the provision of rough estimates on larger tasks.
- Random division of the seminar participants into groups, whereby participants in different groups get different treatments.

- Motivation of the participants to conduct professional estimation work through the information that: a) This is input for our research, b) They may learn from it, and c) They will receive the main results during the seminar.
- Completion of the tasks by the seminar participants.
- Continuation of the seminar, while another researcher inputs the results in a spreadsheet with analysis support.
- Information regarding the outcome of the experiment's main results provided to the participants.

We have, almost without exception, received positive responses from the participants on this kind of participation in seminars.

### **3. Transfer to and Impact on the Software Industry**

The BEST-project members have emphasized the transfer of relevant research results to the software industry in and outside Norway. Examples of how we have transferred the results are:

- Inclusion of our research results in the most recent and most popular estimation textbooks and courses on software engineering, for example, the textbook *Agile Estimating and Planning*, by Mike Cohn; the textbook *Software Estimation*, by Steve McConnell; and several international courses leading to the *Scrum Master* (agile development) title.
- Frequent publication of research summary results in practitioners' magazines, such as *IEEE Software*, for example, guidelines on judgment-based effort estimation [59], how to avoid impact on estimates from misleading and irrelevant information [46], how to avoid making or accepting bids based on overoptimistic effort estimates [75], a debate with Barry Boehm (the most prominent estimation model builder and researcher) on the strengths and weaknesses of model-based and judgment-based effort estimation [18], and regular presentation of our own

and other's research results relevant to the software industry in a regular (every 6th week) column in Computerworld Norway ([www.idg.no/computerworld/](http://www.idg.no/computerworld/)).

- Presentation of results in 10-20 practitioners' seminars and conferences per year (the presentation can be downloaded from 'Presentations' under publications at [www.simula.no](http://www.simula.no)). This includes the organization of our own annual estimation research seminar with key decision makers in the Norwegian public and private sector.
- Spread of our research results on web-resources. This includes writing an article on software development effort estimation on Wikipedia ([en.wikipedia.org/wiki/Software\\_development\\_effort\\_estimation](http://en.wikipedia.org/wiki/Software_development_effort_estimation)), establishing a user group on software cost estimation at [www.forecastingprinciples.com](http://www.forecastingprinciples.com), offering easy download of our research results with practical implications for the software industry at [www.simula.no/research/engineering/projects/best/bibliography](http://www.simula.no/research/engineering/projects/best/bibliography), and offering free access, to the world, the most comprehensive database of studies on effort estimation at [www.simula.no/BESTweb](http://www.simula.no/BESTweb).
- Different types of collaborations with Norwegian IT-companies. This includes advisory work, company-internal seminars, and more formal collaborations in which a PhD student is supported with grants from an industry partner to implement estimation processes based on our research and to conduct valuable estimation research inside the organization.
- Building of two software companies based on the estimation research.

The total effect of our strong emphasis on the transfer of research results to and impact on the software industry is difficult to assess. However, based on numerous e-mails from, and personal communication with, software developers worldwide, we believe that many companies have changed their approach to estimation work based on our research results. The spread of certain research results is likely to be slow, and we expect that the main impact of our process change proposals is yet to come. In particular, we expect that our results on improved uncertainty

assessment processes will be: implemented in more and more organizations, included as textbooks are updated, and incorporated into project management and effort estimation efforts throughout the world.

#### **4. Conclusions and Future Work**

The BEST-project proposes several estimation methods, principles, and guidelines with documented positive effects on estimation accuracy. The project has not yet been able to achieve its goal of developing a comprehensive model of the mental process involved in judgment-based effort estimation. It has, however, achieved a sound basis for the achievement of this goal and for the use of such a model to improve the effort estimation accuracy of the software industry. The project has the required multidisciplinary competencies (with researchers from psychology, education, software engineering, economics, and forecasting), it has acquired the knowledge base through extensive previous work on the topic, and it is infused with the professional background necessary to understand the practical implications of the research findings in software development contexts (namely, many of the project members are, or recently were, software practitioners). The project plans to continue its parallel focus on building a model of the mental steps involved in judgment-based effort estimation and transferring the resulting insight into practical estimation processes. The project also plans to initiate several commercial innovations based on recent and future research findings and has launched a new software company for this purpose.

**Acknowledgments:** Current and former members of the BEST-project: Kristin Børte, Stein Grimstad (project leader 2008-2009), Tanja Miljana Gruschke, Nils Christian Haugen, Magne Jørgensen (project leader 2001-2008), Alf Børre Kanten, Kjetil Johan Moløkken-Østvold and Morten Onshuus. Others who have made research contributions to the project: Scott Armstrong,

Bente Anda, Tore Dybå, Aiko Fallas Yamashita, Bjørn Faugli, Torleif Halkjelsvik, Jo Hannay, Ulf Indahl, Geir Kirkebøen, Barbara Kitchenham, Martin Shepperd, Karl Halvor Teigen and Dag Sjøberg.

## References

1. Moløkken, K. and M. Jørgensen. *A review of software surveys on software effort estimation*. Proceedings of *International Symposium on Empirical Software Engineering*. 2003. Rome, Italy: Simula Res. Lab. Lysaker Norway: p. 223-230.
2. Jørgensen, M. and S. Grimstad. *Over-optimism in software development projects: "the winner's curse"*. Proceedings of *IEEE CONIELECOMP*. 2005. Puebla, Mexico: IEEE Computer Society: p. 280-285.
3. Jørgensen, M., K.H. Teigen, and K. Moløkken, *Better sure than safe? Over-confidence in judgement based software development effort prediction intervals*. *Journal of Systems and Software*, 2004. **70**(1-2): p. 79-93.
4. Farr, L., *Factors that affect the cost of computer programming v.1*. 1964, united states air force Electronic Systems Division: L.G. Hanscom Field, Bedford, Massachusetts.
5. Jørgensen, M. and M. Shepperd, *A systematic review of software cost estimation studies*. *IEEE Transactions on software engineering*, 2007. **33**(1): p. 33-53.
6. Moores, T.T. and J.S. Edwards, *Could large UK corporations and computing companies use software cost estimating tools? - a survey*. *European Journal of Information Systems*, 1992. **1**(5): p. 311-319.
7. Heemstra, F.J. and R.J. Kusters, *Function point analysis: Evaluation of a software cost estimation model*. *European Journal of Information Systems*, 1991. **1**(4): p. 223-237.
8. Hihn, J. and H. Habib-Agahi. *Cost estimation of software intensive projects: A survey of current practices*. Proceedings of *International Conference on Software Engineering*. 1991. Austin, TX , USA: IEEE Comput. Soc. Press, Los Alamitos, CA, USA: p. 276-287.
9. Jørgensen, M., *Estimation of Software Development Work Effort: Evidence on Expert Judgment and Formal Models*. *International Journal of Forecasting*, 2007. **23**(3): p. 449-462.
10. Armstrong, J.S., ed. *Principles of forecasting: A handbook for researchers and practitioners*. *International Series in Operations Research & Management Science*. 2001, Kluwer Academic Publishers: Boston. XII, 849 s. : ill.
11. Goodwin, P., *Improving the Voluntary Integration of Statistical Forecasts and Judgment*. *International Journal of Forecasting*, 2000. **16**(1): p. 85-99.
12. Webby, R.G. and M.J. O'Connor, *Judgemental and Statistical Time Series Forecasting: A Review of the Literature*. *International Journal of Forecasting*, 1996. **12**(1): p. 91-118.
13. Dolado, J.J., *On the problem of the software cost function*. *Information and Software Technology*, 2001. **43**(1): p. 61-72.
14. Sanders, D.E. and L.P. Ritzman, *On knowing when to switch from quantitative to judgemental forecasts*. *International Journal of Forecasting*, 1991. **11**(6): p. 27 - 37.

15. Jørgensen, M. and T.M. Gruschke. *Industrial Use of Formal Software Cost Estimation Models: Expert Estimation in Disguise?* Proceedings of *Empirical Assessment of Software Engineering (EASE)*. 2005. Keele, England: Keele University: p. 1-7.
16. Hammond, K.R., et al., *Direct comparison of the efficacy of intuitive and analytical cognition in expert judgment*. IEEE Transactions on Systems, Man, and Cybernetics, 1987. **17**(5): p. 753-770.
17. Epstein, S., *Cognitive-experiential self-theory: An integrative theory of personality*. The relational self: Theoretical convergences in psychoanalysis and social psychology, ed. R.C. Curtis. 1991, New York: Guilford Press.
18. Jørgensen, M. and B. Boehm, *Software Development Effort Estimation: Formal Models or Expert Judgment?* IEEE Software, 2009(Jan/Feb).
19. Jørgensen, M. and D.I.K. Sjøberg, *An effort prediction interval approach based on the empirical distribution of previous estimation accuracy*. Information and Software Technology, 2003. **45**(3): p. 123-136.
20. Jørgensen, M., U. Indahl, and D.I.K. Sjøberg, *Software effort estimation by analogy and "regression toward the mean"*. Journal of Systems and Software, 2003. **68**(3): p. 253-262.
21. Kitchenham, B., T. Dybå, and M. Jørgensen. *Evidence-based Software Engineering*. Proceedings of *International Conference on Software Engineering (ICSE'04)*. 2004. Edinburgh: IEEE Computer Society: p. 273-281.
22. Dybå, T., B. Kitchenham, and M. Jørgensen, *Evidence-based Software Engineering for Practitioners*. IEEE Software, 2005. **22**(1): p. 58-65.
23. Jørgensen, M., B. Kitchenham, and T. Dybå. *Teaching Evidence-Based Software Engineering to University Students*. Proceedings of *11th IEEE International Software Metrics Symposium*. 2005. Como: IEEE Computer Society: p. 19-22.
24. Moløkken-Østvold, K. and M. Jørgensen, *A Comparison of Software Project Overruns – Flexible vs. Sequential Development Models*. IEEE Transactions on Software Engineering, 2005. **31**(9): p. 754-766.
25. Jørgensen, M., *A review of studies on expert estimation of software development effort*. Journal of Systems and Software, 2004. **70**(1-2): p. 37-60.
26. Jørgensen, M., *Regression Models of Software Development Effort Estimation Accuracy and Bias*. Empirical Software Engineering, 2004. **9**(4): p. 297-314.
27. Jørgensen, M. and K. Moløkken-Østvold, *Reasons for software effort estimation error: impact of respondent role, information collection approach, and data analysis method*. IEEE Transactions on Software Engineering, 2004. **30**(12): p. 993-1007.
28. Grimstad, S., M. Jørgensen, and K. Moløkken-Østvold, *Software Effort Estimation Terminology: The Tower of Babel*. Information and Software Technology, 2006. **48**(4): p. 302-310.
29. Jørgensen, M. and K. Moløkken-Østvold, *How Large Are Software Cost Overruns? Critical Comments on the Standish Group's CHAOS Reports*. Information and Software Technology, 2006. **48**(4): p. 297-301.
30. Grimstad, S. and M. Jørgensen, *Inconsistency in Expert Judgment-based Estimates of Software Development Effort*. Journal of Systems and Software, 2007. **80**(11): p. 1770-1777.
31. Lawrence, M., et al., *Judgmental forecasting: A review of progress over the last 25 years*. International Journal of Forecasting, 2006. **22**(3): p. 493-518.
32. Jørgensen, M., *Top-down and bottom-up expert estimation of software development effort*. Information and Software Technology, 2004. **46**(1): p. 3-16.
33. Moløkken, K. and M. Jørgensen, *Expert Estimation of Web-Development Projects: Are Software Professionals in Technical Roles More Optimistic Than Those in Non-Technical Roles?* Empirical Software Engineering, 2005. **10**(1): p. 7-30.

34. Moløkken-Østvold, K. and M. Jørgensen, *Group Processes in Software Effort Estimation*. Empirical Software Engineering, 2004. **9**(4): p. 315-334.
35. Moløkken-Østvold, K., N.C. Haugen, and H.C. Benestad, *Using Planning Poker for Combining Expert Estimates in Software Projects* To appear in Journal of Systems and Software, 2008.
36. Jørgensen, M. and K. Moløkken-Østvold. *Combination of software development effort prediction intervals: Why, when and how?* Proceedings of *IEEE Conference on Software Engineering and Knowledge Engineering*. 2002. Ischia, Italy: p. 425-428.
37. Castore, C.H. and J.C. Roberts, *Subjective estimates of own relative riskiness and risk taking following a group discussion*. Organizational Behaviour and Human Performance, 1972. **7**(1): p. 107-120.
38. Jørgensen, M. and K. Moløkken. *Combination of software development effort prediction intervals: Why, when and how?* Proceedings of *Fourteenth IEEE Conference on Software Engineering and Knowledge Engineering (SEKE'02)*. 2002. Ischia, Italy: p. pp. 425-428.
39. Grove, W.M., et al., *Clinical versus Mechanical Prediction: A Meta-Analysis*. Psychological assessment, 2000. **12**(1): p. 18-30.
40. Dawes, R.M., D. Faust, and P.E. Meehl, *Clinical versus actuarial judgment*. Science, 1989. **243**: p. 1668-1674.
41. Jørgensen, M. and T.M. Gruschke, *The Impact of Lessons-Learned Sessions on Effort Estimation and Uncertainty Assessments*. To appear in IEEE Transactions on Software Engineering, 2008.
42. Jørgensen, M. *The "Magic Step" of Judgment-Based Software Effort Estimation*. Proceedings of *International Conference on Cognitive Economics*. 2005. Sofia, Bulgaria: New Bulgarian University: p. 105–114.
43. Jørgensen, M. and D.I.K. Sjøberg, *Impact of effort estimates on software project work*. Information and Software Technology, 2001. **43**(15): p. 939-948.
44. Jørgensen, M. and G. Carelius, *An empirical study of software project bidding*. IEEE Transactions on Software Engineering, 2004. **30**(12): p. 953-969.
45. Jørgensen, M. and D.I.K. Sjøberg, *The impact of customer expectation on software development effort estimates*. International Journal of Project Management, 2004. **22**: p. 317-325.
46. Jørgensen, M. and S. Grimstad, *Avoiding Irrelevant and Misleading Information When Estimating Development Effort*. IEEE Software, 2008. **May/June**: p. 78-83.
47. Jørgensen, M. and S. Grimstad. *Judgment-Updating among Software Professionals*. Proceedings of *The 2nd international conference on software knowledge information management and applications (SKIMA)*. 2008. Kathmandu, Nepal: p. 62-67.
48. Jørgensen, M. and S. Grimstad, *The Impact of Irrelevant and Misleading Information on Software Development Effort Estimates: A Randomized Controlled Field Experiment*. submitted to IEEE Transactions on Software Engineering, 2008.
49. Jørgensen, M. and T. Halkjelsvik, *The Effects of Request Formats on Judgment-based Effort Estimation*. submitted to Journal of Systems and software, 2008.
50. Jørgensen, M., *Selection of Effort Estimation Strategies*. submitted to International Journal of Forecasting, 2008.
51. Jørgensen, M. *A Preliminary Model of Judgment-based Project Software Effort Predictions*. Proceedings of *IRNOP VIII*. 2006. Xi'an: Publishing House of Electronic Industry: p. 661-668.
52. Mussweiler, T., *Comparison Processes in Social Judgment: Mechanisms and Consequences*. Psychological Review, 2003. **110**(3): p. 472-489.
53. Trope, Y. and A. Liberman, *Social hypothesis testing: Cognitive and motivational factors.*, in *Social psychology: Handbook of basic principles*, E.T. Higgins and A.W. Kurganski, Editors. 1996, Guilford Press: New York. p. 239-270.

54. Jørgensen, M., *How much does a vacation cost? or What is a software cost estimate?* Software Engineering Notes, 2003. **28**(6): p. 5-5.
55. Jørgensen, M. and K. Moløkken-Østvold. *A Preliminary Checklist for Software Cost Management*. Proceedings of *IEEE International Conference on Quality Software*. 2003. Dallas, USA: IEEE Computer Society: p. 134-140.
56. Edwards, J.S. and T.T. Moores, *A conflict between the use of estimating and planning tools in the management of information systems*. European Journal of Information Systems, 1994. **3**(2): p. 139-147.
57. Goodwin, P., *Enhancing judgmental sales forecasting: The role of laboratory research*, in *Forecasting with judgment*, G. Wright and P. Goodwin, Editors. 1998, John Wiley & Sons: New York. p. 91-112.
58. DeMarco, T., *Controlling software projects*. 1982, New York: Yourdon Press.
59. Jørgensen, M., *Practical Guidelines for Expert-Judgment-Based Software Effort Estimation*. IEEE Software, 2005. **22**(3): p. 57-63.
60. Jørgensen, M., *Realism in assessment of effort estimation uncertainty: it matters how you ask*. Software Engineering, IEEE Transactions on, 2004. **30**(4): p. 209-217.
61. Jørgensen, M. *A Critique of How We Measure and Interpret the Accuracy of Software Development Effort Estimation*. Proceedings of *1st International Workshop on Software Productivity Analysis and Cost Estimation*. 2007. Nagoya: Information Processing Society of Japan: p. 15-22.
62. Grimstad, S. and M. Jørgensen. *A Framework for the Analysis of Software Cost Estimation Accuracy*. Proceedings of *ISESE*. 2006. Rio de Janeiro ACM Press: p. 58-65.
63. Jørgensen, M. and K. Moløkken-Østvold. *Eliminating Over-Confidence in Software Development Effort Estimates*. Proceedings of *Conference on Product Focused Software Process Improvement*. 2004. Japan: Springer-Verlag: p. 174-184.
64. Jørgensen, M. and K.H. Teigen. *Uncertainty Intervals versus Interval Uncertainty: An Alternative Method for Eliciting Effort Prediction Intervals in Software Development Projects*. Proceedings of *International Conference on Project Management (ProMAC)*. 2002. Singapore: p. 343-352.
65. HalvorTeigen, K., *More than X is a lot: Pragmatic implicatures of one-sided uncertainty intervals*. Social Cognition, 2008. **26**(4): p. 379-400.
66. Moore, D.A. and P.J. Healy, *The trouble with overconfidence*. Psychological Review, 2008. **115**(2): p. 502-517.
67. de Venter, G. and D. Michayluk, *An insight into overconfidence in the forecasting abilities of financial advisors*. Australian Journal of Management, 2008. **32**(3): p. 545-557.
68. Overskeid, G., *They should have thought about the consequences: The crisis of cognitivism and a second chance for behavior analysis*. Psychological Record, 2008. **58**(1): p. 131-151.
69. Budescu, D.V. and N. Du, *Coherence and consistency of investors' probability judgments*. Management Science, 2007. **53**(11): p. 1731-1744.
70. Teigen, K.H., A.M. Halberg, and K.I. Fostervold, *More than, less than, or minimum, maximum: How upper and lower bounds determine subjective interval estimates*. Journal of Behavioral Decision Making, 2007. **20**(2): p. 179-201.
71. Teigen, K.H., A.M. Halberg, and K.I. Fostervold, *Single-limit interval estimates as reference points*. Applied Cognitive Psychology, 2007. **21**(3): p. 383-406.
72. Mason, S.J., et al., *Conditional exceedance probabilities*. Monthly Weather Review, 2007. **135**(2): p. 363-372.
73. Jørgensen, M., B. Faugli, and T.M. Gruschke, *Characteristics of Software Engineers with Optimistic Predictions*. Journal of Systems and Software, 2007. **80**(9): p. 1472-1482.

74. Jørgensen, M., *The Effects of the Format of Software Project Bidding Processes*. International Journal of Project Management, 2006. **24**(6): p. 522-528.
75. Jørgensen, M., *How to Avoid Selecting Providers with Bids Based on Over-Optimistic Cost Estimates*. To appear in IEEE Software, 2009. **May/June**.
76. Hannay, J. and M. Jørgensen, *The Role of Deliberate Artificial Design Elements in Software Engineering Experiments*. to appear in IEEE Transactions on Software Engineering, 2008.
77. Lachin, J.M., J.P. Matts, and L.J. Wei, *Randomization in Clinical Trials: Conclusions and Recommendations*. Controlled Clinical Trials, 1988. **9**(4): p. 365-374.
78. Jørgensen, M. and S. Grimstad, *The Impact of Irrelevant and Misleading Information on Software Development Effort Estimates: A Randomized Controlled Field Experiment*. Submitted to a journal, 2008.