The "Magic Step" of Judgment-Based Software Effort Estimation

Magne Jørgensen
Simula Research Laboratory
Oslo, Norway
magnej@simula.no

Abstract*: Estimates of effort are important input to software development project planning and bidding. In the worst case, one single over-optimistic effort estimate can ruin a software development company. Currently, most software development effort estimates are based on expert judgments. In this paper we claim that the estimation process step from understanding the problem to quantifying the required effort to solve the problem is particularly poorly understood and difficult to analyze. Our claim is supported by data from two studies of software professionals' effort estimates. The software professionals seem to be unable to describe this quantification step in detail. A search for models that describe the mental processes involved has not, so far, led to identification of models or theories useful for the purpose of improving judgment-based software development effort estimation.*

1. Introduction

Planning, bidding and budgeting software development projects take as input the estimate of the effort required to complete the project. Surveys of software projects suggest that these estimates are quite inaccurate and strongly biased towards over-optimism. On average, there is an effort overrun of about 30% [1]. Yet, in spite of more than 40 years of research on formal software effort estimation models, nearly all software project estimates are expert judgment-based [2]. In fact, even on the few occasions when formal effort estimations are used, there are reasons to believe that many of the formal estimation processes would be better described as expert estimation disguised as formal estimation model [3], i.e., formal models are applied as means to rationalize the intuition-based software effort estimates.

It is possible, perhaps likely, that it will never be possible to develop effort estimation models that replace expert judgment. The best improvement strategy may, therefore, be to improve the judgment-based effort estimates. One important step is to better understand the processes involved in judgment-based estimation. To date, there has been scant research on this topic. We scanned software effort estimation articles in about 100 journals, and found only one paper that made an attempt to model the expert judgment process [4]. This paper describes the expert judgment process mainly as analogy-based: "*... most judgements are based on the results of referring to historical costs data, and then adjusting up or down accordingly in order to predict the cost of a new project.*" Unfortunately, the paper provides only a shallow description of the processes involved in expert judgment-based estimation.

In other domains, there may be a similar lack of knowledge about the mental processes involved in judgment-based quantification. According to Brown and Siegler psychological research on real-world quantitative expert estimation *“has not culminated in any theory of estimation, not even in a coherent framework for thinking about the process”* [5].

This paper illustrates the problems we encountered when trying to understand the experts' estimation processes. In particular, we focus on the problems of understanding the step from understanding software development work to quantification of the required effort to complete the work. Currently, we believe this step is the least understood and most difficult to understand. We will term it the "magic step" of software cost estimation in this paper.

Section 2 describes a study in which we attempt to understand the processes through the analysis of discussions of estimation teams. The rationale for this study was the conjecture that software professionals might, to some extent, make their estimation strategies explicit as a means to support the validity of their estimates when estimating a software project in co-operation with other estimators. Section 3 describes a study in which we instructed the estimators to describe their estimation processes immediately after they had estimated. Section 4 discusses the results.

2 A study of Estimation Team Discussions

In this study, we analyze the estimation process through video recording, transcription and coding of estimation teams' discussions. This analysis is based on the same data as in our study that compares top-down and bottom-up estimation processes [6], but additional analyses are included in this paper.

2.1 Design

Excerpt from the study design, as described in [6]:

"*The study was conducted within a large international IT consultancy company. The Norwegian branch of that company has more than 1000 employees. Most of the employees are project managers and software developers. A variety of types of applications are developed and maintained, including financial and telecom applications. The company was paid for the participation of seven estimation teams. The estimation teams were selected to be as close to real estimation teams as possible. Each estimation team included one experienced project manager and one or two software developers with relevant background. A senior manager of the company conducted the selection and ensured that each team had sufficient development and estimation competence for realistic estimation. In total, seven project managers and ten software developers participated in the study. Two software development projects (projects A and B) were selected as the basis for the estimation tasks.*

.....

When estimating the effort of the projects, the participants had the following information available:

- *The requirement specifications of the projects.*

- *The company's on-line database of completed projects.*
- *Other available people and documents in the company.*

The information available to the estimation teams was similar to the information they normally had. The main differences from a completely realistic estimation process were, we believed, the limited time available (four hours to estimate two projects), the presence of estimation instructions, the presence of an "experiment leader" answering questions related to the instructions, the lack of contact with the customer, and the video recording of the estimation discussions."

Our instructions were that half of the project estimations should use a top-down estimation process (estimation of the total work "directly", e.g., by comparing similarity to other software projects) and the other half a bottom-up estimation process (decomposition of total work into activities and estimation of each activity separately). Each team estimated the same two projects. A more comprehensive description of the design can be found in [6].

In the previous analysis of the estimation discussions we focused on the relation between the characteristics of the estimation process and estimation accuracy [6]. The analysis in this paper builds upon that analysis, but focuses instead on the step from "understanding the work" to "judging the number of work-hours required for completion of the work".

2.2 Coding of the Teams' Estimation Discussions

We transcribed the estimation discussions word-by-word, including important visual information as comments to the discussion. The transcription included about 180 pages of text. The coding of the discussion was based on repeated reading. We separated the discussion elements into nine categories:

1. **Estimation process discussions (Category: Process)**.
2. **Specification and project understanding (Category: Understand)**.
3. **Search for similar projects (Category: SearchProj)**.
4. **Search for similar project parts (Category: SearchPart)**.
5. **Estimate analysis phase (Category: EstimAna)**.
6. **Estimate design phase (Category: EstimDes)**.
7. **Estimate programming phase (Category: EstimProg)**.
8. **Estimate "other" activities (Category: EstimOther)**.
9. **Estimation of total effort (Category: EstimTot)**.

Formatted: Bullets and Numbering

We marked each text sequence with respect to these categories. To increase the robustness of this categorisation process, another researcher conducted an independent coding of the transcribed text, applying the same categories. The differences between the analyses were examined and the reasons for the differences discussed. Based on these examinations and discussions the final coding was developed. A more comprehensive description of the categories and the coding process can be found in [6].

In this paper we will focus on categories 5-9. Typically, 30-40% of the estimation discussions, measured as amount of transcribed text, were of these types, as can be derived from Table 1. There were seven top-down estimation discussions and seven bottom-up estimation discussions.

Table 1: Discussion Category Distributions

Discussion category	**Top-Down Estimation**	**Bottom-Up Estimation**
1) Process	9%	7%
2) Understand	31%	49%
3) SearchProj	19%	0%
4) SearchPart	6%	1%
5) EstimAna	0%	2%
6) EstimDes	0%	1%
7) EstimProg	8%	16%
8) EstimOther	0%	4%
9) EstimTot	27%	20%

2.3 Analysis of the "Magic Step"

We examined manually all parts of the discussion coded as 5-9, i.e., the categories where there were elements of quantification of work-effort. For each of these parts we analyzed the discussion immediately before an estimate was proposed or adjusted and, if any, the discussion leading to adjustments of the initial estimate. We classified the discussion as belonging to one of the following informal discussion types:

1. **"Out of the air"-numbers without any explicitly described process, no "negotiation". (Category: No Description)**
2. **"Out of the air"-numbers without any explicitly described process, including "negotiation". (Category: No Description + Negotiation)**
3. **Simple rules. The simple rules were mainly classification-based. (Category: Simple Rule)**
4. **Comparison with similar projects. (Category: Analogy)**

Formatted: Bullets and Numbering

Below we include one example of a discussion of each type.

Category: "No Description"
Developer: *My "gut feeling" says about 800 work-hours.*

Category: "No Description + Negotiation"
[The project leader and the developer have just added the estimates for each programming activity and got the sum of 180 work-hours.]

Developer: *I feel that 180 work-hours for programming is low, very low actually....*

Project leader: *Mmmmm. What if we add, say, 40 work-hours on analysis? Then we have 220 work-hours. Then we have 250 work-hours for the implementation.*

Developer: *I think that is more correct.*

Project leader: *Then we should increase the test effort to 200 work-hours. Is that OK? If you say that programming effort is minimum 200 work-hours?*

Category: "Simple Rule"

Developer: *This is tailor-made functionality. Agree? I have this simple rule in my head: Simple user screens is 6 work-hours, medium is 15 and difficult is 60, and if it's extreme you have to add much more.*

Category: "Analogy"

Developer: *The use of* [name of the development tool] *is not very efficient. Even simple reports take time. I remember I estimated two work-days per report [in another project]. We don't know exactly what they want. As it stands here, it seems to be easy.*

[They finally decide that each report requires 10 work-hours each.]

The purpose of this classification of discussion types is to examine how frequently the estimators give, on their own initiative or based on a request from the other estimation team member(s), detailed insight into how the estimates are derived. We suspected that most of the discussions would be of types 1 and 2. When type 3 or 4 was present we expected that steps important for understanding the estimation process would be lacking. There were many estimates involved to reach the estimate of the total effort of one single project. We estimate the total number of estimates in the 14 team discussions to be more than 500.

Table 2 includes the approximate distribution of the discussion types.

Table 2: Refinements of Discussion of Category 5-9

Discussion Type	**Approximate Proportion**
No Description	40%
No Description + Negotiation	40%
Simple Rule	5%
Analogy	5%

As can be seen in Table 2, about 80% of the discussions were of the type where the process was not described at all. A typical example of discussion involving many numbers and no description of the estimation process (Category: No Description + Negotiation) is the following:

Project leader: *How much [effort] would you think is required? ... I have an idea, but would like you to start. If I say what I think, you will only think the same* [laughing]

Developer: *You think so?* [pause]

Developer: *I think it is a simple job ... but ... work effort ... it is perhaps 3-4 work-days*.

Project leader: *I estimated 40 work-hours. There may be synergy-effects here. Did you mean 3-4 work-days on each job or in total*?

Developer: *On each job.*

Project leader: *Then we agree.*

Developer: *I just want to look at the other two if they are very similar. Import of the product register must be more complex, because it has to control the product register.*

Project leader: *Yes, that job is more complex.*

Developer: *Then I would think about 60 work-hours.*

Project leader: *Yes, I think so too.*

As can be seen, there are not much description of the background for the numbers, although there is an awareness that anchors may strongly affect the outcome of the process. The estimation may be described as a negotiation that ends when they find a value that both parties feel they can accept. We have no reason to doubt the development skill of the estimators, i.e., their knowledge about how to complete the task. For that reason, we were somewhat surprised that the negotiations for an increase or decrease in an estimate was always based on "feelings" and what the other party was "willing" to accept, and not on evaluation of rational argumentation.

The estimation discussions involving simple rules were all based on expert-judgment categorization of small software development elements, e.g., user screen, and the work connected to them. In no case was there any discussion on how this categorization was performed. Similarly, the strategy for finding analogies, comparing them with the current project and adjusting for differences, was not described.

We were not able to conclude much about the actual expert estimation quantification processes from this analysis of team discussion among software professionals. An important observation is, nevertheless, that estimation was performed without any argument being produced for particular effort estimates that made it possible to review their quality. While the decomposition work and, partly, the search for analogies could be reviewed to some extent, the quantification step was more "magical".

3 A Study of the Estimators' Own Description of How They Estimate

In this study we instructed software professionals to describe their estimation processes. Results from that study, with a focus different from this paper, have previously been described in [7]. In total, we received descriptions of the processes used for estimation in 52 different projects. The Chief Project Manager of the company was in charge of the data collection. He asked the estimators to complete one questionnaire before starting the software project and another one upon completion of the project.

Similarly to the study discussed in Section 2, we were mainly looking for descriptions of the step from understanding the problem to quantification of the effort. We read all the 52 descriptions repeatedly, with a focus on the quantification steps, and classified them in accordance with a slightly modified version of the categories described in Section 2 (see Table 3).

Table 3: Quantification Strategy Distribution

Quantification Strategy	**Approximate Proportion**
None Described	60%
Simple Rule	30%

Analogy	10%

Perhaps the most striking observation from our analysis of the descriptions was, again, the lack of detail regarding the quantification step. While the processes preceding quantification were, to some extent, rational and reviewable, none of the estimation process descriptions enabled us to understand the quantification step in any depth. Most estimates were based on a decomposition of the work and estimation of the effort required for the development each of the work units (bottom-up estimation). These work unit estimates were typically described as pure expert estimates, and no clues were given as to how they were derived. In some cases, there were references to simple classification-based estimation rules-of-thumb or to similar projects. Similarly to the study discussed in Section 2, however, nobody described how this classification or similarity assessment was made.

4 Discussion and Future Work

The studies described in Sections 2 and 3 suggest that software professionals do not describe the quantification step of software cost estimation in estimation team discussions or when asked to formulate how they estimated the effort. We believe that the main reason for this is a lack of ability, not the lack of willingness to explain the steps.

Evidence from a previous study where we used a think-aloud protocol on simple estimation tasks [8] and experience from our work as estimator advisor strengthen our belief that there is a lack of ability to describe the quantification step. We have, for example, tried several times to make software development estimators explain how they quantified the required effort of a task. The typical response is a detailed description of the steps preceding the quantification and none of the actual quantification. In addition, we observe that the software estimators typically feel uncomfortable when they are requested to present an argued rationale for the estimate. Perhaps they feel that they should be able to describe this step better (after all, they are rational software engineers) but do not know how to do so.

A lack of ability to describe this is, perhaps, not surprising. Perform the following thought experiment for yourself. Assume that you are going to estimate the number of work-hours you need to produce a scientific paper on one of your research topics. You have to design a study, conduct the study, analyze the outcome and write the paper. I am confident that you would, somehow, be able to estimate the required effort, but would you be able to describe the quantification step in detail? Perhaps not. How, for example, did you decide that the effort is more likely to be 250 than 350 work-hours, or why did you select a particular analogy as starting point?

There are several problems inherent in the estimation of the effort required both to produce scientific papers and develop software. There is, for example, typically a lack of complete specification of the outcome and there will frequently be unexpected problems, i.e., events and problems of which you are unaware at the time of estimation. On the one hand, you have a flexibility in delivery and process that may make your estimate to some extent self-fulfilling. (We have reported a study on the substantial impact of the software effort estimate on the project work in [9]). On the other hand, you know that there will probably be unforeseen problems. This type of estimation task is clearly different from, for example, forecasting the weather or estimating the distance between two cities. It would be no surprise to us if this kind of estimation problem involves quite different mental processes compared to less "dynamic" estimation tasks with more complete information and a correct answer.

Our results from a recent study on project bidding suggest that the specification of the properties of the software to be estimated explains only a small part of the variation of estimates [10], and that there may be a substantial "random" (unexplained) element in the determination of the software effort estimates. In that study, 35 companies delivered bids for the same software project. The highest bid was almost 30 times as high as the lowest bids. Even when adjusting for differences in development efficiency and skill levels, this suggests a large unexplained variance in estimates of effort. Interestingly, we found that the companies with the least relevant experience typically had the lowest bids, which suggests that in order to understand the quantification step of software effort estimation we need to understand how the amount of previous relevant experience affects the process.

There are many studies on topics related to understanding of the quantification steps, for example, the following:

- Studies comparing high-level quantification strategies, e.g., how estimation performance is related to level of decomposition [11].
- Studies on how single factors affect quantitative estimates, e.g., how anchors affect software cost estimates [12].
- Studies on how simple estimation strategies, which people would be able to follow without calculation tools, perform. Examples of this type of estimation study are studies on the "heuristics" described in [13].

Unfortunately, we have not been able to identify comprehensive models of how people actually perform the step from understanding the task to quantification of the effort required to complete it in situations similar to software development effort estimation. Our lack of identification of such models may, however, be a result of lack of skill in searching the extensive literature on human judgment. We will strengthen our effort to find models that can be applied as a starting point for a model of expert judgment-based software cost estimates and evaluate them. A good model for this purpose is, we believe, a necessary precondition for systematic improvement of expert judgment-based estimation processes.

References

1. Moløkken-Østvold, K. and M. Jørgensen. *A review of software surveys on software effort estimation*. in *International Symposium on Empirical Software Engineering*. 2003. Rome, Italy: Simula Res. Lab. Lysaker Norway: p. 223-230.
2. 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.
3. Jørgensen, M. and T. Gruschke. *Industrial use of formal software cost estimation models: expert estimation in disguise?* in *EASE*. 2005. Keele: p. 1-7.

4. Rush, C. and R. Roy, *Expert judgement in cost estimating: modelling the reasoning process.* Concurrent Engineering: Research and Applications, 2001. **9**(4): p. 271 - 284.
5. Brown, N.R. and R.S. Siegler, *The role of availability in the estimation of national populations.* Memory and Cognition, 1993. **20**: p. 406-412.
6. Jørgensen, M., *Top-down and bottom-up expert estimation of software development effort.* Information and Software Technology, 2004. **46**(1): p. 3-16.
7. 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.
8. Bratthall, L., E. Arisholm, and M. Jorgensen. *Program understanding behavior during estimation of enhancement effort on small Java programs*. in *Product Focused Software Process Improvement*. 2001. Kaiserslautern, Germany: ABB Corporate Res. Billingstad Norway: p. 356-370.
9. 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.
10. Jørgensen, M. and G. Carelius, *An empirical study of software project bidding.* IEEE Transactions on Software Engineering, 2004. **30**(12): p. 953-969.
11. Armstrong, J.S., W.B. Denniston Jr., and M.M. Gordon, *The use of the decomposition principle in making judgments.* Organizational Behaviour and Human Decision Processes., 1975. **14**(2): p. 257-263.
12. 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.
13. Gigerenzer, G. and P.M. Todd, *Simple heuristics that make us smart*. 1999, New York: Oxford University Press. 432.