The Ignorance of Confidence Levels in Minimum-Maximum Software Development Effort Intervals

Magne Jørgensen

***Abstract*—Software professionals are frequently asked to provide minimum-maximum effort intervals for a given confidence level. They may for example be asked to provide minimum-maximum intervals where it is 90% likely to include the actual use of effort. If their response is the interval from 800 to 1200 work-hours this should correspond to that it is 90% likely that the actual effort will be more than 800 and less than 1200 work-hours. This effort interval information is, amongst others, used in the planning and budgeting of software projects. In this paper we show that software professionals tend to ignore the confidence levels connected with the minimum-maximum effort intervals. As a consequence, the meaning of minimum-maximum effort interval is unclear and the use of such intervals questionable. The experiment used to document the ignorance of the confidence level is based on requesting one group of software developers to be 98% confident, and another group to be 80% confident when providing their effort intervals. In spite of a difference in confidence levels that should generate quite difference effort intervals, the actual intervals were almost the same. This finding challenge commonly recommended effort uncertainty assessment practices, e.g., those implemented in the PERT method, which are based on the assumption that software professionals are able to provide minimum-maximum effort that reflect the stated confidence levels. The finding does also challenge the explanation typically given for too narrow confidence intervals, i.e., that people are over-confident. Instead, we propose that a more likely explanation is that people ignore the confidence level when setting the minimum and maximum values.**

***Index Terms*—Confidence intervals, effort estimates, expert judgment, project management.**

I. Introduction

How much effort does a software project require? This question may be answered by a single point estimate, e.g., that the project most likely requires 1000 work-hours to be completed. A single point estimate does however not provide much information about the *distribution* of possible effort usages. It does, for example, not tell you what you should use as your budget to be, for example, 80% confident not to exceed it. For the purpose of assessing the distribution of possible use of effort, it is common to use three-point-estimates, i.e., to estimate the minimum (most optimistic), most likely and maximum (most pessimistic) effort. A project may for example be estimated to most likely require 1000 work-hours and with 90% confidence between 800 and 1200 work-hours. A 90% confidence effort interval, given that it is correctly set, means that it is 90% probable (9 out of 10 times) that the actual effort will be between the minimum and the maximum effort.

There is little doubt that the statistical theory behind the use of confidence intervals (three point estimates with connected confidence levels) to assess the uncertainty of effort estimates is valid. This has led project planning methods, such as the PERT method [1], to use three point estimates and typically request 98% or 90% confidence intervals of effort usage as input to the planning process The theoretical validity does however not help much if software professionals are not able to know which minimum-maximum interval that reflects a certain level of confidence, i.e., when the input to the confidence intervals is of low quality. Previous studies about experts' abilities to assess how confident they are, unless trained and provided with proper methods, give reasons to be skeptic about the quality of these assessments [2]-[9]. A general finding is that people tend to be over-confident, e.g., provide too narrow intervals for given confidence levels. As an illustration, in [3], we report that the 90% (or "almost sure") minimum-maximum effort intervals of software projects typically include the actual effort only 60-70% of the time. Interestingly, although methods like PERT require confidence intervals as input, they provide no support on how to provide such intervals.

In this paper we examine to what extent software professionals ignore the confidence levels when deciding on minimum and maximum effort. If the minimum-maximum effort intervals are the same regardless of instructed to provide 98% or 80% confidence intervals, we know that the stated level of confidence of the minimum-maximum interval does not give much information about the probability of including the actual effort in the interval. Our study is, as far as we know, the first study on this topic that applies software professionals as subjects.

II. The Study

A. Design

We hired 62 Ukrainian software developers with, on average, 8 years experience in estimating software development effort to estimate a rather simple software project, see specification in Appendix I. The software developers were paid ordinary per hour fees for this task and were instructed to estimate this project similarly to ordinary estimation work. The estimation work should include a brief design of the solution, a description of the technology to be

Manuscript received January 15, 2014; revised April 1, 2014.

Magne Jørgensen is with University of Oslo and researcher at Simula Research Laboratory, Norway (e-mail: magnej@simula.no).

DOI: 10.7763/LNSE.2014.V2.144

used, a list of work packages/activities needed to complete the project, the effort estimates of each work package/activity, and the total estimate of the most likely effort. In addition, we asked them to provide a confidence interval for the total effort.

The developers were randomly divided into two groups: Group98% and Group80%. These groups received identical instructions, with one exception. Those in Group98% should provide a 98%, while those in Group80% should provide an 80% confidence interval of the use of effort. More specifically, the confidence interval instructions and response formats were:

Group98%: How accurate do you think your effort estimate is?
I am 98% confident that I will use between _____ (minimum) and _____ (maximum) work-hours.
NB: A 98% confidence means that you think that the actual effort will be within the minimum-maximum interval in about 98 out of 100 times in similar situations.

Group80%: How accurate do you think your effort estimate is?
I am 80% confident that I will use between _____ (minimum) and _____ (maximum) work-hours.
NB: An 80% confidence means that you think that the actual effort will be within the minimum-maximum interval in about 80 out of 100 times in similar situations.

As can be seen, the confidence level is explained in terms of frequencies and not as probabilities. The motivation for this is that several studies, e.g., [10], [11], suggest that a change from probabilities to frequencies improves the realism of this type of assessments.

Our hypothesis, tested in Section II.B, is that the minimum-maximum effort intervals of those in Group98% would not be significantly wider than those in Group80%, i.e., that the software developers tend to ignore or not sufficiently adhere to the confidence level instruction. We measure the width of an interval, relative to the estimate of most likely effort, as:

RWidth = (Maximum effort – Minimum effort)/Most likely effort

The statistical power of the study is not very high for small and medium large differences in RWidth. Assuming a medium large difference (Cohen's $d = 0.5$) and a significance level of 0.05, the statistical power is only 0.49. Fortunately, we may argue that the difference (the effect size) should be larger given what, normatively speaking, is likely to be the real difference in RWidth for a 98% and an 80% confidence interval.

The evidence for claiming that a normatively correct confidence interval should lead to large effect sizes is taken from an analysis of the estimation error distribution of two software project/task data sets. The first data set includes 42 small/medium large in-house software development projects and the second 443 smaller tasks (user stories) within the same project. The data sets will be sent to interested readers on request. Both data sets imply that the RWidth of a 98% confidence interval should, on average, be about 2.5 times higher than that of an 80% confidence interval. Together with the observation that the standard deviation of RWidth is typically less than 1.0 this means that the expected effect size is at least 2.5 and that the statistical power of the test is close to 100%. In other words, it is very likely that we will find a significant difference in RWidth if the difference in the effort intervals of those in the two groups is close to the normatively correct difference.

B. Results

First, we controlled that the random allocation of developers to the two groups had resulted in similar competence in each group. Table I suggests that this is the case, as both the median length of experience and the median estimates of most likely effort were similar in both groups. Notice that we use the median values in the presentation of the results. The use of the mean value has the risk that a few very high values have a very strong impact on the reported results.

TABLE I: GROUP CHARACTERISTICS

Group	Length of experience (median)	Most likely effort (median)
Group98%	6 years	160 work-hours
Group80%	7 years	144 work-hours

The main result of our study is presented in Table II. Table II shows that there was only a small increase in RWidth from Group80% to Group98%. The difference in RWidth is not statistically significant (two-sided t-test gives p=0.4, Kruskal-Wallis non-parametric test on median values gives p=0.3). The standard deviation of RWidth is 0.26 for both groups, i.e., the standard deviation assumption made in the calculation of the statistical power seems to be valid. The boxplots in Fig. 1 show the distribution of the RWidth values and confirms that there are only small differences in uncertainty assessments between the two groups.

TABLE II: MINIMUM, MAXIMUM AND RWIDTH

Group	Minimum effort (median)	Maximum effort (median)	RWidth (median)
Group98%	110 work-hours	210 work-hours	0.44
Group80%	112 work-hours	176 work-hours	0.40

C. Discussion

The finding in Table II suggests that the software professionals were not much, if at all, affected by the instruction that they should be 98% or 80% confident when deciding on the minimum and maximum effort. The minimum and maximum effort was consequently based on other, unknown to us, reflections. This not only suggest that the software professionals had no method that could tell them when they were 98% or 80% confident, but also that the meaning of minimum-maximum effort intervals is very unclear. If not the level of confidence decides the width of the minimum and maximum use of effort, then what does decided it? Even worse, if the criteria for deciding on the minimum-maximum efforts are not explicit, they are likely to differ from individual to individual and from context to context. Adding confidence intervals from different sources, which in our experience is frequently done in software projects, becomes highly questionable.

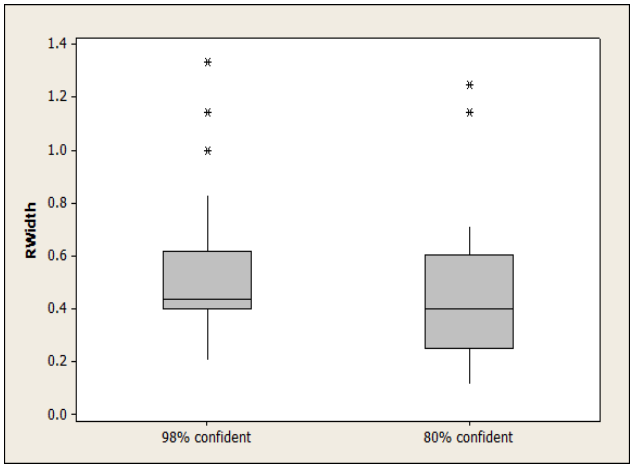

Fig. 1. Distribution of RWidth values.

It is, of course, not possible to generalize from our study of a few software developers to all developers within the software industry. In spite of this, our study adds to the previously mentioned reasons (see Section I) to doubt that the effort intervals used by many software projects can be interpreted as confidence intervals, e.g., that we can use the statistical theory (as is done by the method PERT) related to confidence intervals to analyze the total effort uncertainty of software projects. Our results fit well with the observation that the confidence intervals are typically too narrow, see Section I. The main difference between our work and the previous work on over-confidence when providing confidence intervals is that our explanation of the phenomenon is not over-confidence, but rather ignorance of confidence levels.

It may be objected that the context of this study is artificial and that the participants may have had performed better in more realistic context and with better training in the use of confidence intervals. Although not possible to exclude, we doubt that this has been the case. The developers were experienced software developers, paid ordinary fees for their estimation work and asked to treat this as ordinary estimation work. The effect on training on the width on effort estimation intervals seems also to be limited, see [12].

The main reason for the ignorance of the confidence level, we believe, is that the software professionals have no method to support them for the quite complex task of knowing when something is 98% and when 80% likely.

III. Conclusion

It does not help that a method is theoretically sound if people are not able to provide meaningful input to it. In this paper we demonstrate that the confidence level, typically instructed to be 90% or 98% in software projects, of including the actual effort in a minimum (most optimistic) and maximum (most pessimistic) effort interval is ignored by software developers. If this, as we believe is the case, is typical in most software project contexts, we have that: i) Software projects tend to add minimum-maximum intervals with unknown, and not constant, meaning, ii) The calculations used in methods like PERT cannot be trusted due to violation of an essential assumption, i.e., that the effort intervals reflect the stated confidence interval, iii) The common explanation of too narrow effort intervals, i.e., over-confidence, should be replaced (or at least extended) with an explanation based on ignorance of confidence level.

Appendix

Instructions and Requirement Specification for the Estimation Task

Instructions: Assume that you are asked to develop a shoe sale support system (as specified below) for a client. You will be doing all the development and testing work yourself and you can select the development platform and languages you know best. Assume also that the necessary technical infrastructure for the implementation of the system is available.

Specification: A jogging shoe shop owner needs a software system to support their customers when they look for jogging shoes. The software system should ask a potential jogging shoe buyer about his/her weight in kg, the surface on which the shoes are supposed to be used, whether the shoes are training or competition shoes, etc. Based on the responses on these questions, the system should give a recommendation. There will be about 10 questions to the client and about 50 "rules". All questions and rules are already specified by the shoe shop owner. The rules are of the type: "The shoes xx1 and xx2 are not suited for people over 80 kg" and "The shoes yy1 and yy2 are to be used on hard surfaces." There are, currently, about 100 different types of jogging shoes. Every year there will be new jogging shoes at the market, and new knowledge about jogging. There should consequently be easy to create new and update existing questions and rules.

The system will be used by many inexperienced users and should be robust with respect to incorrect input and give responses to the users that are easy to understand.

References

[1] J. J. Moder, C. R. Phillips, and E. W. Davis, *Project Management with CPM, PERT and Precedence Diagramming*, Van Nostradn Reinhold, 1983.

[2] T. Connolly and D. Dean, "Decomposed versus holistic estimates of effort required for software writing tasks," *Management Science*, vol. 43, pp. 1029-1045, July 1997.

[3] M. Jørgensen, 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*, vol. 70, pp. 79-93, Feb. 2004.

[4] M. Alpert and H. Raiffa, "A progress report on the training of probability assessors," in *Judgment under Uncertainty: Heuristics and Biases*, D. Kahneman, P. Slovic, and A. Tversky, Eds., Cambridge: Cambridge University Press, 1982, pp. 294-305.

[5] D. Kahnemann, P. Slovic, and A. Tversky, *Judgement under Uncertainty: Heuristics and Biases*, Cambridge University Press, 1982.

[6] A. Tversky and D. Kahneman, "Judgment under uncertainty: Heuristics and biases," *Science*, vol. 185, pp. 1124-1131, Sep.1974.

[7] I. Yaniv and D. P. Foster, "Precision and accuracy of judgmental estimation," *Journal of Behavioral Decision Making*, vol. 10, pp. 21-32, 1997.

[8] S. Lichtenstein and B. Fischhoff, "Do those who know more also know more about how much they know?" *Organizational Behaviour and Human Decision Processes.*, vol. 20, pp. 159-183, Dec. 1977.

[9] H. R. Arkes, "Overconfidence in judgmental forecasting," in *Principles of Forecasting: A Handbook for Researchers and Practitioners*, J. S. Armstrong, Ed., Boston: Kluwer Academic Publishers, 2001, pp. 495-515.

[10] S. A. Sloman, D. Over, L. Slovak, and J. M. Stibel, "Frequency illusions and other fallacies," *Organizational Behavior and Human Decision Processes*, vol. 91, pp. 296-309, 2003.

[11] G. Gigerenzer, "Why the distinction between single-event probabilities and frequencies is important for psychology (and vice versa)," in *Subjective Probabilities*, G. Wright and P. Ayton, Eds., Chichester: Wiley, 1994.

[12] M. Jørgensen and T. M. Gruschke, "The impact of lessons-learned sessions on effort estimation and uncertainty assessments," *IEEE Transactions on Software Engineering*, vol. 35, pp. 368-383, May-Jun. 2009.

Magne Jørgensen works as a researcher at Simula Research Laboratory and a professor at the University of Oslo. Previously, he worked with software development, estimation, and process improvement in the telecom and insurance industry. He is one of the founders of evidence-based software engineering and teaches this to students and software professionals. His current main research interest is effort estimation, bidding processes, outsourcing, and software development skill assessments.