The Use of Precision of Software Development Effort Estimates to Communicate Uncertainty

Magne Jørgensen[1]

[1] Simula Research Laboratory, Fornebu, Norway
magnej@simula.no

Abstract. The precision of estimates may be applied to communicate the uncertainty of required software development effort. The effort estimates 1000 and 975 work-hours, for example, communicate different levels of expected estimation accuracy. Through observational and experimental studies we found that software professionals (i) sometimes, but not in the majority of the examined projects, used estimate precision to convey effort uncertainty, (ii) tended to interpret overly precise, inaccurate effort estimates as indicating low developer competence and low trustworthiness of the estimates, while too narrow effort prediction intervals had the opposite effect. This difference remained even when the actual effort was known to be outside the narrow effort prediction interval. We identified several challenges related to the use of the precision of single value estimates to communicate effort uncertainty and recommend that software professionals use effort prediction intervals, and not the preciseness of single value estimates, to communicate effort uncertainty.

Keywords: Software cost estimation; estimation uncertainty; estimate precision; human judgment

1 Introduction

There is a substantial uncertainty in required effort of software development projects, which is reflected in a high average estimation error. A review of seven surveys on software projects gave that the median effort overrun was 21% [1]. Not only is the average error high, the proportion of projects with very high estimation error is substantial. A study of 3.650 software projects found that 12.8% of the projects had a effort overrun of more than 105.9% [2].

The sometimes very high effort uncertainty makes it essential to realistically assess and properly communicate the level of uncertainty of the produced effort estimates. The level of uncertainty may, amongst others, affect the pricing, cost-benefit analyses, prioritization, budgeting and planning of software projects [3-6], i.e., most of the different types of usages of effort estimates involves judgment of the uncertainty of the estimate. A client may, as an illustration, find a project worthwhile to invest in only if the effort uncertainty, and consequently the risk of high cost overrun, is low. Similarly, a software development company may choose not to bid for a project or to add high cost contingency when the effort uncertainty is high.

More recently introduced software development approaches, such as incremental or continuous delivery or other elements of agile development, may give the perception that there is no more need for effort estimates, which would invalidate the need for communicating the uncertainty of estimates as well. The "no estimates"-movement (*zuill.us/WoodyZuill/beyond-estimates/)* seems to argue in this direction. This movement provides good arguments in favour of that there are several situations where we do not need effort estimates and where effort estimates can be harmful (see our own study on negative effect of too low effort estimates on software quality in [7]). Few, if any, of those within the movement do, however, claim that we *never* need to or should estimate, see for example *ronjeffries.com/xprog/articles/the-noestimates-movement/*. The "no estimates"-movement may also be accused of not being clear about of what is meant by an estimate. It seems, for example, that estimates of production capacity, e.g. answers to questions like "How much can be delivered in the next increment?" and "Is it likely that we will be able to deliver a certain amount of features within the client's budget?" are frequently not counted as estimates. Clearly, answers on these questions are based on estimates of effort and would benefit from assessing and communicating the estimation uncertainty. Even when applying a continuous delivery model there will frequently be a need for the client and/or provider to know the approximate need for effort before starting the development of new features. Also of interest is that the most successful agile projects, but not the less successful agile projects, seem to carry out as much up-front estimation and planning as the non-agile projects [8]. Finally, hardly any client, given that they have the choice not to build the software, is likely to start a software project without any knowledge about how much it will cost to meet their business needs. The value of such cost (and benefit) estimates would be limited if there was no communication of the uncertainty of them.

One possible means of communicating the estimation uncertainty in software projects is through the precision of the estimate, e.g., through the number of significant digits or trailing zeros of an effort estimate. For example, an estimate with low precision, such as 1000 work-hours (three trailing zeros), is likely to indicate a much higher effort uncertainty than an estimate with higher precision, such as 1075 work-hours (no trailing zeros). This paper examines the following topics: i) To what degree the estimation precision is actually used to communicate estimation uncertainty, ii) Challenges by using the estimation precision to communicate uncertainty, and iii) How the estimation precision affects the assessments of the competence of the estimator, the trustworthiness of the estimate and, after the actual effort is known, the perceived accuracy of the estimate. If the actual effort is 1200 work-hours we may, for example, assess an estimate of 1000 work-hours to be more accurate, given the low precision which implied a large estimation interval, than the much more precise and clearly incorrect estimate of 1075 work-hours.

Research from many contexts [9-12], including software engineering contexts [13-15], reports a strong tendency towards over-confidence in the accuracy of one's estimates. As an illustration, in [16] the authors found that most software developers assessed it to be likely (at least 60% likely) that the actual effort would be within +/-10% of the estimated effort. In reality, however, only 15% of the project efforts fell within this narrow effort interval.

Research on answers given to so-called "almanac questions", such as the question "*How long is the Nile?*" suggests that more precise answers, e.g., that the Nile is 6180 km compared to that the less precise answer that it is about 7000 km, are interpreted as indicating increased confidence in the answer and more competence in the topic. The research also finds that more precise answers have larger influence on other people's decisions [17].

The only study on the use and effect of single value precision in a software development estimation context is, as far as we know, the study reported in [18]. That study failed to replicate previous result that more precise suggestions (estimation anchors) were more influential for subsequent predictions. An effort suggestion (estimation anchor) of 998 work-hours did not have more impact on the subsequent effort estimate than an effort suggestion of 1000 work-hours. A potential explanation for this difference in results is that an overly precise suggestion, such as 998 work-hours for a complex project, may be seen as indicating incompetence, rather than higher competence in a software development context. There may consequently be domain and context differences in the interpretation and judgmental effect of more precise predictions.

This paper aims at extending previous research by:

- Examining the use of the estimation precision to indicate software development effort estimation uncertainty (Sections 2 and 3).
- Examining how the estimation precision is interpreted as indicator of estimator competence and estimate trustworthiness, and how estimation accuracy of overly precise effort estimates is evaluated (Section 4).

We discuss the findings and conclude in Section 5.

2 Measuring Single Value Effort Precision

The measurement of the precision of a single value effort estimate, in connection with the potential use of it to communicate estimate uncertainty, is not straightforward. It may be easy to agree on that precision and estimate uncertainty is, or at least should be, connected. An estimate of 1000 work-hours is for example expected to be less accurate, i.e., more uncertain, than an estimate of 830 work-hours. It is however not easy to formalize this connection in an effort estimation context. What is, for example, a reasonable interpretation of the uncertainty of an estimate of 830 work-hours? Should it be interpreted to communicate that the expected actual effort is in the interval [825, 835], in the interval [820, 840], or some other interval? How should a belief in that the actual effort is likely to be in the interval [810, 850] be communicated? An estimate of 800 work-hours indicates less precision than intended and an estimate of 830 work-hours may indicate too high precision. Also, the implied effort intervals have no obvious connected confidence level. We may argue that the estimator believes that it is likely that the actual effort will be inside the interval, but it is not clear whether that corresponds to, for example, 70%, 80% or 90% confidence. In addition, the number of significant digits does not say anything about trailing value of 5, which may also be used to indicate uncertainty.

The use of the Fibonacci numbers in agile estimation of software development effort, see for example [19], may be argued to assume that the intended precision of user story-based estimates, typically in story points, is about +/-30%, assuming non-overlapping uncertainty intervals. This is so, because the ratio of two succeeding Fibonacci numbers is always approximately 1.6. The estimated numbers themselves, however, do not necessarily communicate that level of uncertainty.

There may also be precision measurement challenges connected with the choice of units. Estimating values connected with work effort or calendar time means that the value 10 is not always the natural unit. Indicating that a task takes about one day of work may, for example, result in the seemingly very precise estimate of 8 work-hours.

From the above discussion, see also [20] for elaboration on several of the above challenges, we see that the precision of a single point estimate may not be an optimal method of communicating clearly about effort estimation uncertainty.

For the purpose of our studies, trying to address some of the above measurement challenges, we apply the following measures:

- *NumbTrailZero*: Number of trailing zeros. If there is at least one trailing zero, this indicates (but do not guarantee, since a trailing zero may be a significant digit) that some rounding has been taken place, or that the estimate has a high granularity.
- *RelPrec*: The relative precision (uncertainty) of an estimate measured as a function of trailing zeros and the estimate. We include the factor w to enable different precision interpretation and assume that all estimates are integer number. The need for the w-factor is motivated by the lack of commonly accepted precision interpretation in effort estimation. The interpretation of the meaning of, for example, 8000 work-hours could be 8000 +/- 500, 8000 +/- 1000 or something else.

$$RelPrec(w) = w \cdot 10^{NumbTrailZero} / Estimate \; . \qquad \textbf{(1)}$$

Example: Assume that we observe the estimates 3000 work-hours and 3020 work-hours, where the first estimate has three and the second has one trailing zero. We then have that the relative precision, for the interpretation w = 1.0, is $10^3/3000$ = 33% (corresponding to the interval 3000 +/- 1000 work-hours, assuming symmetric intervals) for the first estimate and 0.3% (3020 +/-10 work-hours) for the second. Changing the w-value to be 2.0 gives a relative precision of 67% (3000 +/- 2000 work-hours) and 0.7% (3020 +/-20 work-hours). If the actual effort, for example, turns out to be 3655 work-hours we have that the implied effort interval, by assuming w=1, of the estimate 3000 +/-1000 work-hours includes the actual effort (estimation error of 22% and *RelPrec* of 33%), while the assumed implied interval estimate 3020 +/-10 work-hours does not (estimation error of 21% and *RelPrec* of 0.7%). Notice that the number precision interpretation commonly applied in physics would be the one with w=0.5 and is related to the reliability of measurement rather then expected accuracy of estimates. We are unaware of other studies applying the *RelPrec* measure for studies on numerical precision. This does not mean that the proposed measure in any way is innovative. If there had been a normative interpretation of how trailing zeros in software development effort estimates should be connected with effort

estimation uncertainty, we would not need the *w*-factor and the measure would be trivial.

If the trailing zeros of estimates are successfully applied to indicate estimation uncertainty, with the interpretation indicated by the *w*-factor, we would expect that the average *RelPrec* is similar to the average absolute value of the relative error. In particular, if the average error tends to be much higher than *RelPrec*, then the effort estimate is likely to have been too precise to reflect the effort uncertainty. The lack of standardized precision interpretations in software effort estimates means, however, that we will examine different *w*-factors and also analyse which precision interpretation (which *w*-factor) that would give a good correspondence between precision and accuracy.

3 Effort Uncertainty Indicated by Estimate Precision

Table 1 summarizes our examination of the effort estimates of three different software project datasets. The datasets were selected based on availability and we do not make any claim that they will be representative for the software development industry as a whole. The purpose is mainly to indicate the existence and degree of systematic of use of precision of cost or effort estimates to indicate estimation uncertainty in software projects.

An important reason for high precision in software professionals' effort estimates is, based on our experience from several organizations, that the estimate of the total project effort is based on adding the effort of many activities, i.e., based on bottom-up based estimation processes. Consequently, adding one activity of about 2 work-hours with one with about 100 gives the sum of 102 work-hours, which is not likely reflect the total uncertainty given the presumably high uncertainty of the activity estimated to take 100 work-hours.

Trailing zeros may not always be a result of estimate rounding based on uncertainty considerations. They may also be a result of request for early, rough, perhaps analogy-based, estimates. We can, however, exclude this reason for trailing zeros in the examined data sets, since all estimates were based on bottom-up estimation processes where effort of many activities were added to find the total effort. Trailing zeros can, of course, also be significant digits.

Our analysis of the three data sets may be summarized to say that some software projects used estimate precision to communicate the effort uncertainty, but that most software projects communicated estimates with a precision much higher than warranted by the actual effort usage uncertainty. The variance in the *w*-factors that would lead to good correspondence between precision and uncertainty suggest that, even when trailing zeros are used, that there are various interpretations of what they mean in terms of estimate precision.

Table 1. Use of estimate precision in three data sets

Data set description	*Precision indicated by trailing zeros*
Data set 1: Seven very large (larger than 440 mill NOK, corresponding to larger than about 50 mill Euro) Norwegian software development projects with governmental client. Data set found at www.nrk.no/fordypning/tvilsomt-om-kvalitetssikring-virker-1.11936733. Median estimation error of 21%.	Only one project estimate (which were in mill. NOK) had trailing zeros. The other estimates were presented to the nearest million NOK. The only project estimate using trailing zeros to indicate relative precision (two trailing zeros, *RelPrec* = 29%) had corresponding relative accuracy (31%) for *w*=2, but were seemingly over-precise, i.e., not reflecting the cost estimation inaccuracy of the software projects, for precision interpretations based on lower *w*-factors. For all other projects the estimates were too precise to reflect the observed estimation uncertainty.
Data set 2: Fifty-two medium large and small software development projects (average estimated effort of 850 work-hours) carried out by a Norwegian software provider for various clients. Median estimation error of 14% (sixteen of the projects were not completed at the time of data collection). We asked the project leaders, at time of estimation, to assess the expected accuracy of the estimates, with the alternatives +/-10%, +/-25%, +/-50% and more than +/-50% error of estimates. Data set is available upon request to the author.	Twenty-one (40%) of the project estimates had trailing zeros. The project estimates that used trailing zeros to indicate relative precision (median *RelPrec* of 18% for w=1) had corresponding relative median estimation accuracy (20%). Seventeen of the twenty-one projects with trailing zeros of the estimates provided informed about the expected precision of the estimates. The relative precision of these estimates, with *w*=1, corresponded well with the projects' expected precision. The +/-10% estimation error category (n=11) had a median relative precision of 6%, the +/-25% category (n=5) had a median relative precision of 17%, and the +/-50% category (n=1) had a relative precision of 50%. This suggests that the project managers in these projects actually used the precision of the estimate to communicate the expected accuracy of the estimates.
Data set 3: Thirty Medium large projects (average estimated effort of 2300 work-hours) carried out in-house in a large Norwegian company. Median estimation error of 11%. Data set is available upon request to the author.	Twelve (40%) of the project estimates had trailing zeros. The project estimates using trailing zeros to indicate relative precision (median precision of 3%, for *w*=1) had not corresponding relative median estimation accuracy (9%). Only two of the project estimates used two trailing zeros or more. These two projects were the only ones with correspondence between estimate precision and accuracy.

4 Competence, Trustworthiness and Accuracy

The precision of an effort estimate may not only be read as an indication of the estimation uncertainty. It may also be read as indication of competence of the source (the estimator) and/or the trustworthiness of the estimate. Our hypothesis was that estimate precision, both in terms of less trailing zeros and narrower intervals, would be interpreted as increased competence of the source and increased trustworthiness of the estimate.

To test this we invited 236 project managers and software developers from six different software providers located in Poland, Romania and Ukraine. The mean length of experience was 6 years, varying from 0.5 to 35 years of experience. Twenty-four per cent were project managers. Eight per cent were female. The participants were asked to describe how they would interpret varying degrees and types of precision of effort estimates.

First, we gave the participants the following scenario and questions:

> Four different developers (Developer A, B, C and D) are estimating the same project and spend about the same time to read and understand the requirement specification.
>
> Developer A says: "*I think I will need 1020 work-hours to complete the project*"
> Developer B says: "*I think I will need 1000 work-hours to complete the project*"
> Developer C says: "*I think I will need between 900 and 1100 work-hours to complete the project*"
> Developer D says: "*I think I will need between 500 and 1500 work-hours to complete the project*"
>
> Assume that this is the only information you have about the four developers.
>
> Question 1: Which of the developers would you think is the most and the least competent?
> Question 2: Which of the estimate would you trust the most and the least?

The rationale behind the chosen estimates and estimation intervals were as follows:

- The precision of 1000 and 500-1500 work-hour estimates may be said to be similar in precision, assuming the precision interpretation corresponding to w=0.5. Both represent a situation where the estimation uncertainty is high.
- The estimate of 1020 work-hours represents a situation with very precise estimate. Given that the other developers, especially B and D, communicate high estimation uncertainty, the participants have to choose between believing that the high precision is a consequence of high competence, i.e., low estimation uncertainty, or over-precision. If they believe that the estimate is likely to be over-precise, i.e., that the estimate precision does not reflect the actual effort usage uncertainty, they have to assess to what degree this is likely to be an indicator of low competence.
- The interval 900-1100 work-hours is a situation with very precise (narrow interval) estimate. Similarly to the situation with the 1020 work-hours estimate,

the participants have to decide to what extent this indicates high competence and, if not, whether an overly narrow effort interval indicates low competence or not.

We suspected that the software professionals' responses on Question 1 and 2 would be strongly correlated, in spite of Question 1 requesting judgment of the developer and Question 2 requesting judgment of the estimate. If you, for example, do not think a developer is competent, you will hardly trust his/her estimate. The inclusion of both questions was mainly to see if they respondents tended to think that a developer with precise estimates would be the most competent, and at the same time not trusting his/her estimates. This may happen if, for example, the respondents thought that over-confident software developers are the most competent, but their estimates are usually less trustworthy.

There are several threats to the *external validity* of the results of this experiment. The above situation with four estimates on the same projects is not a typical situation for software professionals. Similarly, software professionals are not typically asked to evaluate competence and trustworthiness of the estimates based on characteristics of the estimates alone. Usually, they will know more about the context and the software professionals providing the estimates. These threats means that we should be careful about using the results to make claims about of what will happen in more realistic situations, e.g., situations where only one estimate is received and where the receiver knows much more about the context and the estimator. It is however our belief that we, in spite of these limitations, may get a first indication of how the precision of the estimate affects how software professionals think about those who provide the estimates and the estimate itself in situations with little context information. To what extent more context information in more realistic situations will over-ride the observed effects we will not be able to make claims about.

There are also threats to the *internal validity* of the results of this experiment. The software professionals were only allowed to select one developer as the most/least competent or having the most/least trustworthy estimate. This means that if, for example, one developer were judgment to be just slightly better by most respondents, this developer will turn up as the significantly most competent in our study. This is a limitation of just asking for the ranking and not allowing selecting more than one alternative. The direction of the results (the ranking) should, however, not be affected by this design choice.

When the participants had completed the first two questions, they were randomly divided into two groups and given a scenario where they were informed about the actual effort. For one group the actual effort was described to be 800 work-hours and for the other 1200 work-hours. We wanted to know how the estimate precision affected the participants' assessment of the accuracy of the estimate. For this purpose we asked them the following question:

Question 3: Assume that all four developers completed this project and that all of them used close to 800 (Group 1)/1200 (Group 2) work-hours to complete the project. Which of the developers do you think had the most and the least accurate estimate?

We had no prior expectation about which of the four estimates the participants would chose as the most and the least accurate. The actual effort values of 800 and 1200 work-hours are both outside the narrow interval of 900 – 1100 work-hours, i.e., Developer C may be said to be wrong about his/her estimate. The precise estimate of 1020 work-hours may also be said to be wrong, since it deviates quite much from both 800 and 1200 work-hours. The estimate of 1000 work-hours may be interpreted as less wrong if, for example, it is interpreted (w=2) as 1000 +/- 200. The interval 500-1500 work-hours is clearly correct, but may be viewed as not informative enough to qualify for an accurate estimate.

The responses from the two first questions are displayed in Table 2 and for the last question in Table 3.

Table 2. Competence and trustworthiness

Estimate	*Most competent*	*Least competent*	*Most trustworthy*	*Least trustworthy*
Developer A (1020)	6%	31%	7%	49%
Developer B (1000)	11%	13%	7%	14%
Developer C (900 – 1100)	74%	1%	70%	1%
Developer D (500 – 1500)	9%	55%	16%	36%

Table 2 shows that the narrow interval of Developer C was assessed to indicate the highest competence of the source and the highest trustworthiness of the estimate by most participants. The least competent and trustworthy were assessed to be the wide interval by Developer D and, interestingly, the very precise estimate of Developer A. The results, consequently, suggest that precision in the format of narrow effort intervals makes an estimator look more competent and the estimate more trustworthy, while high precision of single value estimates has the opposite effect. We discuss limitations and possible explanations of this observation in Section 5.

We found no large differences in responses dependent on role (project manager or developer), gender, length of experience, or company.

Table 3. Accuracy (answers of the project managers in brackets)

Estimate	*Most accurate when Act = 800*	*Least accurate when Act = 800*	*Most accurate when Act = 1200*	*Least accurate when Act = 1200*
Developer A (1020)	1% (0%)	41% (45%)	9% (12%)	24% (32%)
Developer B (1000)	13% (13%)	7% (10%)	21% (16%)	18% (16%)
Developer C (900 – 1100)	69% (68%)	0% (0%)	55% (60%)	6% (8%)
Developer D (500 – 1500)	17% (19%)	52% (45%)	15% (12%)	52% (44%)

Table 3 reports the same tendency found in Table 2. There is again a positive evaluation of the narrow interval of 900-1100 work-hours. This was the case in spite of that the narrower effort interval did not include the actual effort, while the wider effort interval of 500-1500 work-hours did. The result corresponds with earlier research that reports that people tend to emphasize and reward informativeness rather

than correctness [13, 21, 22]. While 500-1500 work-hours is a "correct" interval, it may be felt to give less useful information than the "incorrect" interval of 900-1100 work-hours.

The situation for the precise (1020 work-hours) and the less precise (1000 work-hours) single value estimate was, similar to the situation in Table 2, the opposite. Here the more precise estimate was considered by more participants to be the least accurate. This was the case even for the group with actual effort of 1200 work-hours, i.e., even in the situation when the precise estimate was closer to the actual effort than the less precise.

A limitation of the single value estimate comparison is that an estimate 1020 work-hours may be said to be clearly over-precise. The interval 900-1100 work-hours may also be said to be over-precise, but less over-precise than 1020 work-hours.

Another limitation is that our results may have been different if we had a pairwise comparison of estimates. The difference in responses for the 1020 and the 1000 work-hour estimate may, for example, be different if not the two other, effort interval alternatives were present.

In spite of the described limitations and threats, we believe that the observation that over-precise intervals but not over-precise single estimate are rewarded in terms of competence and trustworthiness points at challenges when it comes to communicating estimation uncertainty through estimate precision.

5 Discussion and Conclusion

Previous research, such as the study reported in [23] and the studies referred to in the introduction of this paper, tend to find that people with more precise estimates are, on average, more confident and more accurate. Previous research also suggests that people in many situations tend to convey the expected accuracy of their estimates by use of trailing zeros, i.e., by estimate precision. It is therefore not unreasonable to expect that less trailing zeros of an effort estimate and more narrow effort prediction intervals are interpreted as coming from a more competent and accurate source.

Our results only partly correspond with those of the previous research. We did, in accordance with previous research, find that more precise (more narrow) prediction intervals were interpreted as coming from a more competent source and to be more trustworthy. We did, however, not find the same effects for more precise single value effort estimates. Neither did we find that most software projects, but instead only a minority, tried to convey the expected uncertainty through the precision of single valued effort estimates.

The difference between our and the previous results may have been caused by differences in context, e.g., that software professionals know based on previous experience that people with overly precise single value effort estimates tend to be less competent than those with estimates with more trailing zeros or effort intervals. Then, however, we should perhaps expect to find the same effect for the more narrow effort prediction interval. In an earlier study, see [16], we found that software professionals with the high confidence and narrow effort prediction intervals did *not* give the most accurate effort estimates.

The differences between our and previous results may also be a result of the particular numbers and effort intervals selected for the scenario used in our study. A more precise effort interval than 900-1100 work-hours, e.g., the interval 950-1050, or a less precise single value estimate than 1050 work-hours, e.g., the value 1100, may have led to different results. More research is needed to get more insight into how different contexts affect the precision-related interpretations. (We conducted a follow-up study on with 83 software developers where we observed that very narrow intervals, such as 1100-1020 work-hours were considered to come from low confidence source. This study confirmed the finding that intervals are perceived to come from more confidence sources than single estimates with similar precision. The interval 1000-1200 work-hours was for example considered to come from a much more confidence source than the estimate 1100 work-hours.)

Finally, the difference in results may also be a result of that the participants assumed different underlying estimation processes in the interval and the single value estimation context. The estimate 1050 work-hours may be interpreted as coming from a process with addition of most likely effort values without any uncertainty analysis. The effort interval 900-1100 work-hours may, on the other hand, be interpreted as coming from a more thorough, and consequently more competent and trustworthy, process where both minimum, most likely and maximum efforts have been considered.

The perhaps most surprising result is that the participants, including the project managers, even after knowing that the effort interval of 900-1100 work-hours was overly narrow, i.e., did not include the actual effort of 800 or 1200 work-hours, still thought that this was the most accurate interval. This suggests that effort prediction intervals are not interpreted as wrong or inaccurate when the actual effort is close, although not inside the interval. One possible reason for this result is that trailing zeros of the minimum and maximum values communicate that that these boundary values also have uncertainty. Our effort interval, where the boundary values have two trailing zeros, may consequently have been interpreted, for example, as the interval from 800+/-100 to 1100 +/-100 or as "between around 900 to around 1100 work-hours". Another potential reason is that, in spite of not being able to include the actual effort, they expected that the prediction interval 900-1100 work-hours to be the most useful, i.e., a result in correspondence with the informativeness findings reported earlier. Estimation accuracy measurement by software professionals is then not a mechanical calculation of deviations, but also affected by the usefulness of the estimate.

The findings of our study and their *practical implications* (recommendations) include the following:

- The observed challenges related to the measurement and interpretation of different levels of precision of single value effort estimates suggest that uncertainty communication by use of estimate precision is, at best, unclear. Which level of effort usage uncertainty does, for example, an estimate of 1100 work-hours intend to communicate? Is it 1050-1150 work-hours (w=0.5), is it 1000-1200 (w=1) work-hours, or is it something else? In addition, how do an estimator convey that he/she expects the effort to be between 1300 and 1700 work-hours by the use of the precision of a single estimate? How should a group of software professionals proceed to agree on a proper w-value (interpretation of trailing

zeros) and communicate this interpretation to other parties? While we, for the above reasons, would not recommend single value estimate precision to communicate effort uncertainty, we believe that the common practice of high precision (none or few trailing zeros) of highly uncertain effort estimates should be avoided to avoid communicating a higher certainty that is warranted. This is in particular the case when no other means, such as effort prediction intervals, are used to communicate the uncertainty.
    - **Recommendation 1**: Do not use or rely on the precision of single effort estimates to communicate estimation uncertainty.
    - **Recommendation 2**: Avoid high precision estimates in situations where the estimation accuracy is low. This also applies for the values used as minimum and maximum effort values in prediction intervals.
- The uncertainty communicated by use of effort prediction intervals also seems to have interpretation challenges, but perhaps less than the challenges connected with single value estimates. To enable reasonable clear communication of expected uncertainty of effort estimates, we therefore believe that effort prediction intervals, preferably including confidence levels, is to be recommended.
    - **Recommendation 3**: Use prediction intervals to derive and communicate estimation uncertainty, i.e., use minimum-maximum effort intervals with connected confidence levels. Examples of how to do this can be found in [24, 25].
- Our results on the assessment of competence and trustworthiness suggest that software professionals, when providing effort estimates, will be more positively evaluated by those receiving the estimates when communicating narrow effort intervals rather than highly precise single value estimates.
    - **Recommendation 4**: Be aware your tendency to interpret narrow effort intervals (but not highly precise numbers) as indicating high competence and trustworthiness, even when it is demonstrated to be over-confident. Use other means that the effort prediction interval itself, such as the estimation process and the relevant experience of the developer [16], to evaluate estimator competence and estimate trustworthiness.

References

1. Halkjelsvik, T. and M. Jørgensen, *From origami to software development: A review of studies on judgment-based predictions of performance time*. Psychological Bulletin, 2012. **138**(2): p. 238-271.
2. Budzier, A. and B. Flyvbjerg, *Making-sense of the impact and importance of outliers in project management through the use of power laws*. Proceedings of IRNOP (International Research Network on Organizing by Projects), At Oslo, 2013. **11**.
3. Little, T., *Schedule estimation and uncertainty surrounding the cone of uncertainty*. Software, IEEE, 2006. **23**(3): p. 48-54.

4. Jørgensen, M., *Evidence-based guidelines for assessment of software development cost uncertainty*. Software Engineering, IEEE Transactions on, 2005. **31**(11): p. 942-954.
5. Moses, J., *Measuring Effort Estimation Uncertainty to Improve Client Confidence*. Software Quality Journal, 2002. **10**: p. 135-148.
6. Kitchenham, B. and S. Linkman, *Estimates, uncertainty, and risk*. IEEE Software, 1997. **14**(3): p. 69-74.
7. 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.
8. Serrador, P. and J.K. Pinto, *Does Agile work?—A quantitative analysis of agile project success*. International Journal of Project Management, 2015.
9. McKenzie, C.R.M., M. Liersch, and I. Yaniv, *Overconfidence in interval estimates: What does expertise buy you?* Organizational Behavior and Human Decision Processes, 2008. **107**: p. 179-191.
10. Cesarini, D., Ö. Sandewall, and M. Johannsesson, *Confidence interval estimation tasks and the economics of overconfidence*. Journal of ecnomic behavior & organization, 2006. **61**: p. 453-470.
11. Winman, A., P. Hanson, and P. Jusling, *Subjective probability intervals: how to reduce overconfidence by interval evaluation*. Journal of experimental psychology: learning, memory and cognition, 2004. **30**(6): p. 1167-1175.
12. Klayman, J., et al., *Overconfidence: It depends on how, what and whom you ask*. Organizational Behaviour and Human Decision Processes, 1999. **79**(3): p. 216-247.
13. 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.
14. Jørgensen, M. and K.H. Teigen. *Uncertainty Intervals versus Interval Uncertainty: An Alternative Method for Eliciting Effort Prediction Intervals in Software Development Projects*. in *International Conference on Project Management (ProMAC)*. 2002. Singapore.
15. Teigen, K.H. and M. Jørgensen, *When 90% confidence intervals are 50% certain: On the credibility of credible intervals*. Applied Cognitive Psychology, 2005. **19**(4): p. 455-475.
16. Jørgensen, M., B. Faugli, and T. Gruschke, *Characteristics of software engineers with optimistic predictions*. Journal of Systems and Software, 2007. **80**(9): p. 1472-1482.
17. Jerez-Fernandez, A., A.N. Angulo, and D.M. Oppenheimer, *Show Me the Numbers Precision as a Cue to Others' Confidence*. Psychological science, 2014. **25**(2): p. 633-635.
18. Løhre, E. and M. Jørgensen, *Numerical anchors and their strong effects on software development effort estimates*. submitted for publication, 2014.
19. Cohn, M., *Agile estimation*. 2006: Prentice Hall.
20. Ferson, S., et al., *Natural language of uncertainty: numeric hedge words*. International Journal of Approximate Reasoning, 2014.

21. Yaniv, I. and D.P. Foster, *Graininess of judgment under uncertainty: An accuracy-informativeness trade-off*. Journal of Experimental Psychology: General, 1995. **124**(4): p. 424.
22. Yaniv, I. and D.P. Foster, *Precision and accuracy of judgmental estimation*. Journal of Behavioral Decision Making, 1997. **10**(1): p. 21-32.
23. Welsh, M.B., D.J. Navarro, and S.H. Begg. *Number preference, precision and implicity confidence*. in *Annual Meeting of the Cognitive Science Society (33rd: 2011: Boston, USA) CogSci 2011*. 2011.
24. Briand, L.C., K. El Emam, and F. Bomarius. *COBRA: A hybrid method for software cost estimation, benchmarking, and risk assessment*. in *International Conference on Software Engineering*. 1998. Kyoto, Japan: IEEE Comput. Soc., Los Alamitos, USA.
25. Jørgensen, M., *Realism in assessment of effort estimation uncertainty: It matters how you ask*. IEEE Transactions on Software Engineering, 2004. **30**(4): p. 209-217.