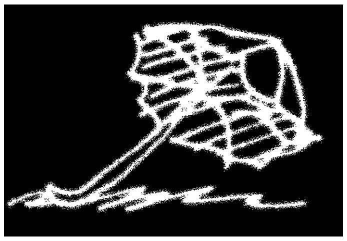


Estimates, Uncertainty, and Risk

BARBARA KITCHENHAM and STEPHEN LINKMAN, *University of Keele*

The authors examine four of the leading contributors to estimate uncertainty and assert that a broad-based approach to risk management spanning all of an organization's projects will be most effective in mitigating the penalties for estimate error.

Tom DeMarco recently made a significant throwaway statement when he said "the software cost estimation problem is solved."[1] He made this comment during a lecture at a European conference on software estimation. You could consider this a contentious statement, since upwards of 100 participants were there to consider the latest estimation and sizing methods. However, DeMarco's point is essentially correct. Almost all people involved in software estimation, whether they are consultants, tool vendors, researchers, or software managers, would agree that to get good estimates you need estimation methods and models based on your organization's performance, working practices, and software experience. DeMarco understood this as early as 1982.[2] However, a major problem remains: although software managers know what to do, they just don't do it.

In our view, the main reason for this paradox is that managers do not understand how to use estimates correctly. In particular, they usually do not handle properly the uncertainty and risks inherent in estimates. This article discusses the sources of uncertainty and risk, their implications for software organizations, and how risk and uncertainty can be managed. Specifically, we assert that uncertainty and risk cannot be managed effectively at the individual project level. These factors must be considered in an organizational context.

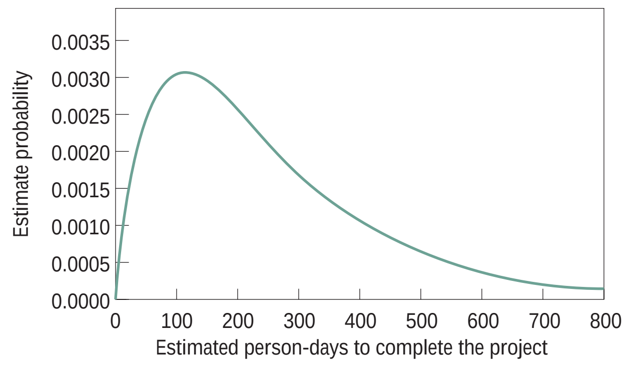


Figure 1. *Estimating a project's completion time using a Gamma distribution with median = 200. The Gamma function is often used to model waiting or service times in queuing theory and, because we are estimating the effort to perform a task, it is a plausible model.*

SOURCES OF ESTIMATE UNCERTAINTY

For many years, software estimation experts have pointed out that an estimated value is one of a range of values with a specific probability of being realized. For example, DeMarco[2] proposed the following definition:

An estimate is a prediction that is equally likely to be above or below the actual result.

Estimate uncertainty occurs because an estimate is a *probabilistic* assessment of a future condition. Estimators produce an estimate using an estimation *model*. The model may be formulated as a standard productivity rate for a specified task, a set of ad hoc rules, or a mathematical formula. But however it is derived, the model itself can be a source of estimate error. We examine this source along with three other sources of estimate uncertainty and risk: measurement error, assumption error, and scope error.

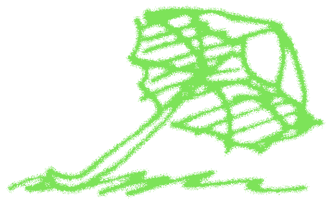
Measurement error. This occurs if some of the input variables in a model have inherent accuracy limitations. For example, as a result of Chris Kemerer's work,[3] function points are assumed to be at least 12 percent inaccurate. Thus, if you estimate a product size of 1,000 function points, measurement error could mean that the real size is anywhere between 880 and 1,120 function points. So applying a model of 0.2 person-days per function point means your estimate will have a range of uncertainty between 176 and 224 person-days, with a most likely value of 200 person-days.

Model error. This occurs because all empirical models are abstractions of reality. No estimation model can include all the factors that affect the effort required to produce a software product. Factors that affect effort but are not included explicitly in the model contribute to the model error. For example, a model such as 0.2 person-days per function point is usually obtained from results observed or recalled from previous projects. It is unlikely that any future projects will achieve the same ratio, but the model is expected to be all right "on average." If you base a model on past project data, you should calculate the associated inaccuracy by using, for example, the mean magnitude relative error.[4] Thus, if you have an estimation model with an inherent 20 percent inaccuracy and your product is 1,000 function points in size, your estimate is likely to be between 160 and 240 person-days. In addition, measurement inaccuracy and model inaccuracy are additive, so the real range of values for your estimate is between 140.8 (160 – 19.2) and 268.8 person-days, with a most likely value (still) of 200 person-days.

However, using the mean magnitude of the relative error to assess model accuracy can be misleading. There is no *a priori* reason for underestimates and overestimates to be of equal size; in fact, the reverse is much more likely. An underestimate is naturally bounded: the minimum effort required for an activity is zero. An overestimate, on the other hand, is not naturally bounded: for any value above the most likely there is a nonzero probability that the overestimate will be greater than that value. So a probability distribution for the effort estimate in our example might look something like Figure 1, which is based on a Gamma distribution[5]:

$$f(x) = \alpha(\alpha x)^{\beta-1}\exp(-\alpha x)/\Gamma(\beta)$$
$$\text{where } x \geq 0$$

where α and β are both real and positive; and $\Gamma(\beta)$ is the Gamma function, which is a generalization of the factorial function, with $\alpha = 2$ and $\beta = 119$. The Gamma function is often used to model waiting or service times in queuing theory and, because we are estimating the effort to perform a task, it is a plausible model. However, any other distribution used to assess wait-



ing times is also appropriate.

The "most likely value" is the estimate that has a 50-50 chance of being greater or less than the actual (median). In this case, the median is 200. However, the modal value of this distribution (the turning point) has the value 120 and the theoretical mean has the value 238 (= β/α).

Assuming the Gamma distribution is correct, you can assess the probabilities associated with the informal upper and lower bounds calculated earlier. In this case, there is a probability of about 0.33 of completing the project in less than the lower bound of 141 person-days, and a probability of about 0.34 of taking more effort than the upper bound of 269 person-days. An important property of a skewed probability distribution is that although you are equally likely to overrun as to underrun the median value, the overruns and underruns will not always balance out. For example, if you estimate 200 person-days with the Gamma distribution there is an equal chance—0.21 probability—of either saving 100 person-days or overrunning by 150 person-days.

Assumption error. This occurs when we make incorrect assumptions about a model's input parameters. For example, your assessment that a product's size is 1,000 function points rests on the assumption that you have correctly identified all the customer's requirements. If you can identify your assumptions, you can investigate the effect of their being invalid by assessing both the probability that an assumption is incorrect and the resulting impact on the estimate. This is a form of risk analysis. For example, suppose you believe that there is a 0.1 probability that the requirements' complexity has been underestimated and, if it has, you estimate another 100 function points will be required. You can obtain the basic estimate's risk exposure from the following formula:

Risk exposure = (E2 – E1) × P2

where

- ♦ E1 = effort if the original assumption is true,
- ♦ E2 = effort if the alternative assumption is true, and
- ♦ P2 = probability that the alternative assumption is true.

In our example, assuming E1 is the most likely estimate for the original assumption (200 person-days) and E2 is the most likely estimate for the alternative assumption (220 = 1,100 × 0.2 person-day), then

Risk exposure =
(220 – 200) × 0.1 = 2 person-days

This risk exposure corresponds to the *contingency* you need to protect yourself against the assumption error. However, the probabilistic nature of risk means that the allowed contingency cannot protect a project if the original assumption is invalid.

Risk exposure and model error are independent: identifying the impact of a wrong assumption does not increase or decrease the estimate uncertainty due to model or measurement error. This implies that you should avoid double-counting the effects of uncertainty. In particular, if you include uncertainty in the model, you should not include it in the risk assessment. For example, during a project's early stages and during preproject feasibility and bidding, you may not know which specific software engineers will work on the project. Typically, you might assume the work will be done by an "average" or even a novice team and factor your risk accordingly. However, depending on the estimation model you use, you may have to consider the impact of novice engineers separately.

Suppose you are using a model of the form 0.2 person-day per function point. There is no need to add any contingency for engineering experience *if* you have obtained the model by averaging across all your projects, irrespective of team capability. In this case, the effect of different team experience is included in the model error. However, suppose you have a more sophisticated model, such as three different productivity models: 0.1 person-day per function point for an experienced team, 0.2 person-day per function point for an average team, and 0.5 person-day per function point for an inexperienced team. If the models were obtained by averaging *within* the team experience groups, the effect of team experience is catered for in the model and does not contribute to the model error. In this case, if you assume you'll have an average team, you can explicitly assess the impact if that assumption is wrong.

Scope error. This special case of assumption error is incurred when your project is outside your estimating model's domain. For example, if you derive your estimating models from 4GL Information Systems applications, the model is probably inappropriate for knowledge-based applications. In this case, however, it may be difficult to quantify the impact of the scope error.

You should avoid double-counting the effects of uncertainty: If you include it in the model, you should not include it in the risk assessment.

If your estimation models or methods are completely out of scope, you cannot produce a meaningful effort estimate. Under such circumstances, it is sensible

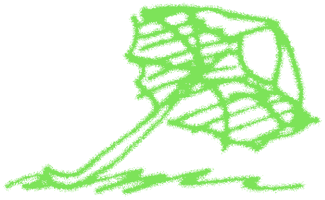


TABLE 1
ACTUAL PROJECT EFFORT FOR 10 PROJECTS ESTIMATED AT 200 PERSON-DAYS

Project number	Actual effort (hypothetical)	Residual (actual – estimate)
1	55	–145
2	80	–120
3	115	–85
4	150	–50
5	183	–17
6	220	20
7	265	65
8	320	120
9	400	200
10	570	370
Total	2358	358

to avoid producing any estimates. You might consider fixed budget feasibility studies or requirements analyses, joint ventures with customers, or subcontracting as alternative strategies for dealing with "unestimatable" projects.

MANAGING UNCERTAINTY AND RISK

There is no way to manage risks and uncertainty for a single, one-off project. If a single project had to ensure against all possible risks and uncertainty, its price would be prohibitive. If you estimate for the worst case in the previous example, you would use the upper bound of 269 person-days of effort and then add the effect of incorrect requirements assessment—another 20 person-days—for a total of 289 person-days. If you compare this with the most likely effort (200 person-days) you can see that bids based on the most pessimistic estimates are unlikely to succeed. Alternatively, if you bid projects using the lower bound and ignore risks—in this case, using the value of 141 person-days—you will win a lot of bids and soon go bankrupt.

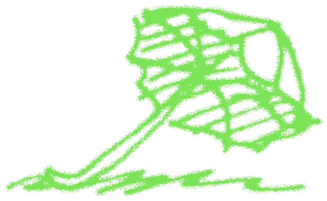
Organizational focus. The only way to successfully manage project risk is to manage risk across your organization's entire project portfolio. This is the basic principle used by insurance companies. It is easiest to understand this method if you consider the risk exposure derived from incorrect assumptions. In our example, we showed that a 0.1 probability of missing requirements that amount to 100 function points would imply a risk exposure of 2 person-days, although if the requirements were actually missed you would need an additional 20 person-days to complete the product. Two person-days is an appropriate contingency because the problem is not likely to happen. You can manage the risk by considering a group of projects, each with its own contingency. If you have 10 projects, each with a similar risk in terms of probability and impact, you would have a contingency fund of $10 \times 2 = 20$ person-days. In the course of completing those projects, you would expect only one of the projects to call on the contingency funds (using up the 20 person-days).

This approach to managing contingency has two important implications:

♦ The contingency fund must be handled by the company (or department), not by the individual projects. In fact, the individual projects *must not* include the contingency element in their own plans.

♦ The risk exposure must be similar across the set of projects contributing to the contingency fund. If one project in 10 has a contingency of 50 person-days against a potential loss of 500 person-days, and the remaining projects each have a contingency of 2 person-days against a potential 20 person-days' loss, the contingency fund will be 68 person-days. Thus, if the project requiring 500 person-days calls on the contingency fund, the fund will be insufficient. Unusually large projects pose a disproportionate risk.

Mean, not median. Model and measurement uncertainty are slightly different. If your estimate error is symmetric, the best estimation policy



is to use the most likely estimate. In this case you have a 50-50 chance of underestimating and of overestimating; more importantly, the values of underestimates and overestimates will even out. So, over a set of projects your gains would balance your losses. However, it is much more likely that estimate error is not symmetric. In this case, to ensure that losses and gains balance out you must use an estimate larger than the median value as the basis for pricing. For example, if your estimate error followed the Gamma distribution shown in Figure 1, and your company had 10 projects estimated to have the same median value (200 person-days), you might expect to see a set of actual values as shown in Table 1. If all those projects were budgeted to cost 200 person-days, you would overrun by 358 person-days. However, if each project were budgeted to cost 238 person-days (its theoretical mean), your budget would exceed your costs by 22 person-days. Thus, to manage model error, you must ensure that you base your price and budgets on the mean rather than the median.

This analysis assumes that all the projects had the same estimate value. If one project is much larger than the rest, the averaging effect may not be sufficient to cover your potential risks, as we saw when considering the effects of assumption error.

One way to reduce the problem of skewed error terms is to reduce the estimate variance. If you can identify factors that systematically affect project effort, you should include them in your models to reduce the residual variation. If the value of any of the model variables is unknown, you can make assumptions and use assumption analysis to assess project contingency.

If you cannot improve your estimation model, you still must ensure that the estimate you produce refers to the mean of your estimate distribution.

EFFORT ESTIMATION PROCEDURES

If you use statistical regression to develop an estimation model, your estimate will correspond to the mean value. If, however, you use an estimation method without a statistical base, you may need to *correct* your estimate.

When you are developing an estimation model (or process), you should always check your model's performance against past data. To assess whether your estimation model needs a correction factor, calculate the residual for each of your past projects. Residuals are the deviation between the value predicted by the estimation model and the actual value, that is,

$$r_i = y_i - y_i(\text{est})$$

where y_i is the actual effort or duration, y_i(est) is the estimated effort or duration, and r_i is the residual for the *i*th project in the dataset.

If your estimation model is well-behaved, the sum of the residuals will be approximately zero. If the sum is not close to zero, your estimation model is biased and you must apply a correction factor. To correct your estimate, you add the mean residual to each estimate. Unlike goodness-of-fit statistics such as the mean magnitude relative error, the residual sum and mean *do not* use absolute values. Negative and positive values are supposed to cancel one another out! For example, the residual for each of 10 hypothetical projects is shown in Table 1. The sum of the residuals is 358 person-days and the mean of the residuals is 35.8; that is, projects take on average 35.8 person-days more than their estimate. Statistically, the residual is not significantly different from zero since the standard error of the mean residual is 50.4 person-days. However, the average residual is big enough to cause a major problem over a project portfolio. In this example, a 10-project portfolio would have been underbudgeted by 18 percent. Thus, you would increase the next project's pricing and budgeting estimate by 35.8 person-days to correct for the estimate bias.

Estimates are produced for a reason. During bidding, they are used to assess a project's likely cost and determine appropriate prices or budgets. However, estimates are certain to be inaccurate, so software managers need to know how to minimize the risks caused by estimate inaccuracy. With the exception of companies that can offset the cost of a project's development with high-volume product sales, estimate inaccuracy can only be managed across an organization's total project portfolio.

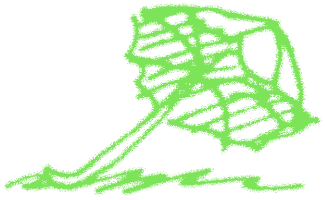
To handle estimate inaccuracy, you need the following strategies.

1. If possible, restrict your organization to projects of a similar size—one large project presents more of a risk than several small projects.

2. Use the mean, not the median or mode, to estimate effort. If you are not using a statistically based model, you may need to correct your estimates by adding the average residual to the estimate produced by the model.

Except for high-volume products, estimate inaccuracy is best managed across an organization's total portfolio.

3. Use estimation models that include all relevant factors. If some of the factors are not known when the estimate is required, make an assumption and adjust your risk assessment to account for the impact



of incorrect assumptions.

4. As suggested by J. D. Edgar,[6] hold the contingency on a company/organization level, not at the project level. Project-level plans should be based on an unbiased estimate of effort, not on the risk-adjusted estimate.

These suggestions distinguish between the estimate that should be the basis of a project budget (an unbiased estimate of the expected staff effort) and the estimate that should be used as part of a pricing exercise (the risk-adjusted estimate). We view pricing as a separate commercial activity from costing. Pricing policies and bidding procedures should be based on the risk-adjusted estimate, but should also consider the amount of the contingency, the worst-case costs, and the extent of model error. In addition, there are always commercial factors that must be considered. For example, it is perfectly acceptable for senior managers to price a product below its expected cost for commercial reasons. However, they should not expect a project manager to be responsible for the risk inherent in that decision. The project budget should be based on the unbiased estimate irrespective of the pricing policy, and the organization should accept responsibility for the risks that result from its pricing policy.

At first sight it might appear that our suggestions are difficult to adopt, but unless companies organize their management structures in ways that recognize the different roles of project management and portfolio management, it is difficult to see how they can benefit from improved estimation or better risk assessment. ◆

REFERENCES

1. T. DeMarco, "Function and Disfunction," *Proc. European Software Control and Measurement Conf.*, ESCOM Science Publishers B.V., Leiden, Netherlands, 1995.
2. T. DeMarco, *Controlling Software Projects*, Prentice Hall, Englewood Cliffs, N.J., 1982.
3. C. Kemerer, "Reliability of Function Points Measurement: A Field Experiment," *Comm. ACM*, Vol. 36, No. 2, 1993, pp. 85-97.
4. S.D. Conte, H.E. Dunsmore, and V.Y. Shen, *Software Engineering Metrics and Models*, Chap. 3, Sect. 3.5, Benjamin/Cummings, Menlo Park, Calif., 1986.
5. L.J. Bain. and M. Englehart, *Introduction to Probability and Mathematical Statistics*, PWS Publishers, Boston, 1987.
6. J.D. Edgar, "How to Budget for Program Risk," *Concepts*, Summer 1982, pp. 6-73.

Barbara Kitchenham is a principal researcher in software engineering at the University of Keele. Her research interests are centered on software metrics and their application to project management, quality control, and evaluation of software technologies. She has worked both in industry and academia and has been involved with several joint national and international software engineering research projects. She has written numerous articles and published a book on the topic of software measurement.

Kitchenham received a PhD from the University of Leeds. She is an associate fellow of the Institute of Mathematics and Its Applications and is a fellow of the Royal Statistical Society.

Stephen Linkman is principal researcher in software engineering in the Computer Science Department at the University of Keele. He has interests in metrics, risk estimation, quality, and evaluation and assessment; he has published papers in all these areas. Prior to joining the University of Keele, Linkman was a principal consultant with the United Kingdom National Computing Centre. Prior to this he had worked in various development and research roles for International Computers Ltd. and its parent company Standard Telephone and Cable.

Linkman received a BSc in Engineering from the University of Leicester.

Address questions about this article to Kitchenham or Linkman at Dept. Computer Science, Keele University, Keele, Staffs ST5 5BG, UK; barbara@cs.keele.ac.uk.