Making better estimates: range estimates

Part of a series on Making better estimates. Single point estimates don’t accurately represent the natural variation in a task. A range estimate of low and high is a good first step to improve accuracy. With a little definition and some simple math, range estimates can be used to much more accurately and responsibly estimate a project.

Range estimates

Single point estimates do a poor job of representing the variability in the actual time required for a task. The detailed explanation of why this is so involves understanding the form of the probability distribution function of a typical software task’s completion time. Steve McConnell’s book on Estimating has a good explanation of this.

A range estimate (low and high, say) gives a lot more information about the nature of the task being estimated. The difference between low and high indicates the uncertainty the team has in the estimate, or the natural variability of the task itself. A big spread indicates more uncertainty; a lower difference indicates more confidence and less variability.

Our experience has taught us that simply asking developers to make two estimates (“low” and “high”) doesn’t really result in much more information than a single point estimate. Developers tend to be optimistic when it comes to individual estimates. So without a little structure, the “low” estimate usually means the absolute smallest amount of time the task could take – an event with a very low probability of coming true. The “high” estimate means what it’s most likely to take to get done. This leaves a lot of stuff that could push the actual task completion time out unaccounted for (the long right tail of the estimate’s probability distribution function), and your overall project likely to be under-estimated.

If we’re making coarse, large-grain estimates, then we’ll use what we refer to as the “50/90 technique” described in Chapter 17 of Mike Cohn’s book Agile Estimating and Planning. This approach to project buffering is closely related to Goldratt’s critical chain project management techniques.

In this approach, our “low” estimate is the value at which we have a 50% chance of the task taking longer than the estimate, and a 50% chance of the task taking less than the estimate. If the distribution were symmetrical (unfortunately, it’s not), this would be the midpoint. A useful, but slightly inaccurate way of thinking of this is as the “most likely” case estimate. The “high” estimate is the value at which we have a 90% chance of completing the work in less time. Making the 90% estimate requires either a lot of confidence and knowledge, or a really high estimate. On a task of high variability or unknown elements, the spread between “low” (50%) and “high” (90%) can be quite large.

So how should we use the range estimate? Summing the 90% estimates for all tasks will give a very large estimate for the project. After all, it’s very unlikely that you’ll hit the high estimate on every single task. It might feel like you’ve been on such projects before, but that’s probably because you weren’t making 90% estimates for the high estimate. If you’ve done your 90% estimates accurately, then a 10 task project only has a one in 1 billion chance of exceeding the high estimate on every single task. Using the sum of the high estimates would be terrible sandbagging.

Using the sum of the 50% estimates is also a problem. Doing so doesn’t account for any of the natural variation in the tasks, or the asymmetry of the completion time distribution function. The approach we use is to add a project buffer to the sum of the 50% estimates. You can think about the project buffer as receiving a contribution from each task in the project. You don’t know in advance which tasks will draw upon the project buffer, but you want to make sure that each task has contributed to it in proportion to the likelihood of need. The spread between the 50% and 90% estimates indicates the potential for a task to go over and make a withdrawal from the project buffer.

The project estimate is simply the sum of all the most likely task estimates (the 50% estimates), plus a project buffer. Since the buffer is sized based on the spread between the low and high estimates, it protects the project from variability in a responsible manner. With this approach you're neither sandbagging, nor irresponsibly underestimating.

The calculation we favor for project buffer is:

$$\text{Buffer} = \sqrt{\sum_{i=1}^{\text{ALL}} (90\,\%\ \text{Estimate}_i - 50\,\%\ \text{Estimate}_i)^2}$$

The overall project estimate then is:

$$\text{Project Estimate} = \text{Buffer} + \sum_{i=1}^{\text{ALL}} 50\,\%\ \text{Estimate}_i$$

90% can be reduced to 80% or 70% or 60% for lower risk programs.