10 ways to avoid stupid project estimates

Version 1.0
April 6, 2009

By Jerry Loza

Projects run on estimates. You either have to provide estimates of things such as how much effort your task will require or you will need such information from others. Unless you're in one of a small handful of roles, you will surely come to rely on estimates of others, even if it's just to frame your own estimates. And if you're a project manager, the estimates of others are probably all you have to work with.

Overall, project estimates are only as good as the individual estimates associated with the component tasks. As much discussion as there is on the estimation process, there isn't a great deal of information on getting better estimates from others. Here are a few ideas to help you see this aspect of estimating more clearly -- and help you avoid leading your project astray before it ever starts.

Vetting the estimates you get today

1 Let history be your guide

It is true: History does repeat itself. I can't tell you the number of times I've seen developers give an estimate of two days and actually take four. There's nothing wrong with that, because there was probably four days' worth of work to begin with. The problem is that I also see people in meetings repeatedly believing them. If it has taken twice as long as the estimate the last five times an estimate was given, why would anyone think the estimate will hit this time? Beats me, but some people continue to fall for it. Unlike the stock market, past performance *can* be an indicator of the future.

2 Ask for details (proof)

Learn the justification for an estimate. If you ask why they think it will take two days, and you get a reasonable task-by-task break down of the two days' work, two days may be right on the mark. But if your developers just shrug their shoulders, it may be time to worry. This also provides a great opportunity to assess their assumptions.

3 Challenge the numbers

Push back. Ask questions. Project managers seem to do this pretty naturally, but challenge the developer or development team to explain why they think it will take so long. If they can justify their estimate, at least you will know that they gave it some thought. If they can't, perhaps they should be thinking about their estimate some more before you start relying on their figures.

Don't be surprised if the estimate actually goes up. It would not be the first time I've seen a discussion with developers expose previously undiscovered project tasks. While not great news, it's still better to find out about this sooner rather than later. Left hidden, those tasks would have surely destroyed the timeline when they came to the surface and jumped onto the critical path.

4 Measure

Okay, you know your developers are always missing their estimates, but by how much? If you aren't measuring it, you really don't know whether you have a big problem -- or maybe don't have a problem at all. This also ties into the history suggestion above. How long does it usually take? You don't know unless you track this information.

Metrics also go a long way toward improving estimates. Like any process, you can't improve it if you don't measure it. Unfortunately, a deep dive into this topic would be an entire article (or two) in itself. But if you don't have a clue where to start, just start capturing expected date versus actual delivery date. You might be surprised by the patterns that develop.

5 Bracket the work

This is for those who just can't come up with a time estimate for the work. First, try helping them break down their work into smaller, more manageable, tasks. Next, you may find it useful to use a simple bracketing technique to help the person zero in on the figure that is appropriate. "Is five days enough time?" "Can you do it in less than 10?" Continue in this fashion until you get a figure that everyone is comfortable with. Of course, you should be challenging and asking for details throughout this process.

6 Get a second opinion

When you're hiring a builder, they say you should get several estimates, throw out the highest and the lowest, and then go with the one in the middle. There might be something to that approach. When working with a team, you're probably getting your estimates from the team's point person -- a lead developer, for example. While this is efficient, it may add some risk to the accuracy of any estimate you receive, especially if there is a wide variance in the estimates of the individual team members. You may be getting the result of a faulty team consensus, a blatant falsehood to avoid having to face some development reality, or an estimate flawed by an innocent miscommunication.

If you have reason to suspect an estimate may be off, or if the team has a bad track record for this sort of thing, a second opinion may be in order. Seek an informal read from other members of the team. If you run the team estimate past an individual developer, and he wrinkles his nose at it, you may have some more digging to do. If everyone is of a like mind, you probably have as good a number as you can expect.

7 Remember that two heads are better than one

Run your figures past someone who might have experience with your type of project. If this person is not on your project, all the better. Perhaps you're a sucker for good news. Perhaps they are telling you what you want to hear. An objective third party is exactly what you need to free you of your bias and give you a clearer view of reality.

Improving future estimates

8 Give feedback

You have captured task estimates, and you have tracked actual. From this, you may have learned that certain groups always underestimate by 25 percent. That is certainly good to know, but don't let the value stop there. Provide feedback to the team members. Help them identify patterns in how they estimate and challenge them to find the root cause of their bad estimates. Generally speaking, people really do want to do a good job, so provide them the tools and a few suggestions. Your folks will be able to take it from there.

9 Reward and penalize

Penalize the bad estimate, not that a task estimate is too long. When estimates are shortened, it's generally because that shorter number is what's expected. That is where the "If all goes well, it will take…" comes from. Be serious. When was the last time "all went well"? There's nothing wrong with qualifying an estimate, but adding an unrealistic assumption as a way to give a bad estimate only hurts the project. You want to drive toward behaviors that give better estimates. Celebrate when estimates are hit. When estimates are missed, you don't need to tar and feather the offenders, but you can treat it as a remedial education opportunity (the very act of which will be a penalty of sorts).

Be careful about rewarding beating an estimate differently from hitting it. If you create the situation where it is better to beat an estimate than simply meet it, you will create an incentive for overestimating.

10 Develop a culture

Using the correct rewards and penalties goes a long way toward establishing the right culture, but what is the right culture? Certainly, it's one in which those providing estimates feel safe in providing accurate information as opposed to providing the information they believe supervisors want to hear. However, it must also be one in which everyone strives to improve the quality of their estimates and the notion of providing a bad estimate is as bad as providing bad program code. Managers must also respect those providing them with information, once those people have proven themselves. Ultimately, if you can get peer pressure to “manage” the estimates, your job will become considerably easier.

Improve your project... one estimate at a time

You can't run a project well with bad estimates, any more than you can build a house with faulty bricks or lumber. Fortunately, you don't need to do everything yourself (nor could you) to get estimates you can trust. Exercising the simple ideas above will help you filter out the bad numbers you get today, improve the numbers you get tomorrow, and generally make your life a whole lot better -- at least as far as your project is concerned.

Additional resources

- TechRepublic's Downloads RSS Feed XML
- Sign up for the Downloads at TechRepublic newsletter
- Sign up for our 10 Things Newsletter
- Check out all of TechRepublic's free newsletters
- 10 ways to get a slipping project back on track
- 10 things you should know about working with an offshore team
- 10 things you should do near the end of a project

Version history

Version: 1.0
Published: April 6, 2009

Tell us what you think

TechRepublic downloads are designed to help you get your job done as painlessly and effectively as possible. Because we're continually looking for ways to improve the usefulness of these tools, we need your feedback. Please take a minute to drop us a line and tell us how well this download worked for you and offer your suggestions for improvement.

Thanks!

—The TechRepublic Content Team