

[RSS Feed](#) | [Twitter](#)

www.MichaelDKelly.com

Software Testing

- [Home](#)
- [Blog](#)
- [Events](#)
- [Writing](#)
- [Workshops](#)
- [Media](#)
- [Services](#)

Search

November 17, 2007 | [4 comments](#) | [Software Testing](#)

[Estimating testing using spreadsheets](#)

Have you ever seen one of these?



I bet you have. It's the staple of large software projects everywhere. It's the estimation spreadsheet; the silver bullet of project planning. Enter the factors and the functionality, guesstimate the size, and the total effort for the project is generated in a very nice clean column. Yea right...

Now perhaps I'm not being fair. The spreadsheet above was developed by David Tonne and I while working on an article titled "Managing the Test Project." At the time, I was a tester working under Dave on a project Dave was managing. I have a lot of respect for Dave; he delivered two large and very difficult projects while I worked with him. We consider our article to be a success because we both learned a bit about what motivated the other and how we both plan our work

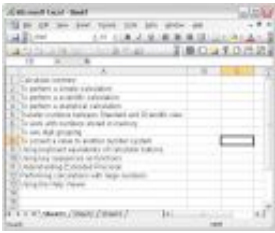
From Dave's point of view, most test managers fail to engage actively in the estimating process. They then complain about lack of resources or condensed test windows at the end of the project. While he's trying to deliver a product to the customer, we're complaining about why life's so hard and nobody listens to us. That may not be how you and I work, but many testers *do* work that way.

To be more specific, my problem is not with using spreadsheets for estimation. I use spreadsheets when I estimate. My problem is with how I typically see them used:

- Many times they oversimplify a complex problem
- They don't typically account for the rapidly changing focus of a test team
- They don't take into account the skills of the project team ("Just plug in any resource...")
- They don't account for what we don't know (which is often a lot)

Formulas and auto-calculating columns gloss over the messy details of smart people struggling with difficult problems. Instead, I prefer to get the people doing the work involved and to get into the details of each task (that I'm aware of).

For an example of how I might approach a problem, let's look at estimating testing for the Windows Calculator (we used the Calculator in Windows XP Pro for this example). I started by opening the Help file and pretending that was the list of requirements (or stories) I was given for estimation purposes.



This might actually make a fairly interesting set of tests:

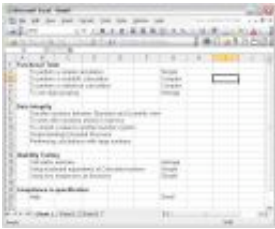
- The Calculator overview could equate to some sort of usability testing.
- You have the functional tests for these items:
 - o To perform a simple calculation
 - o To perform a scientific calculation
 - o To perform a statistical calculation
- You have tests for the integrity of temporal data with these items:
 - o Transfer numbers between Standard and Scientific view
 - o To work with numbers stored in memory
- Here's some bounds checking:
 - o Understanding Extended Precision
 - o Performing calculations with large numbers
- Compliance to a specification occurs with the Help Viewer

And that just gets us started! We could identify all sorts of tests based on this list alone.

Once you have your ideas listed, move them around according to size, complexity, risk, type of testing, etc. — find relationships. Draw deeply from your past experience. Even better, get your team involved and find a whiteboard. Based on some of the testing we identified above, we might change our idea list to look like the following figure.

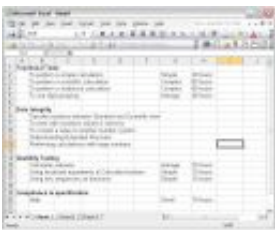


Within each category of tests, we might imagine that work effort will look similar, because each category we identified is a type of testing. If it makes sense, we assign a qualifier—Simple, Average, Complex, Small, Medium, Large, etc.—something that can be used as a multiplier and at the same time show the relationship of the items (see the following figure).



Notice we didn't identify a relationship for our data integrity tests. That may be because we just can't readily see that relationship yet. That's okay. This is the difference between using a spreadsheet to *help* your estimation and *relying* on a spreadsheet for your estimation. As long as we can justify our estimate in some other way, we don't need to get all of our numbers using the same method. In fact, many times it makes more sense to use several methods for estimation.

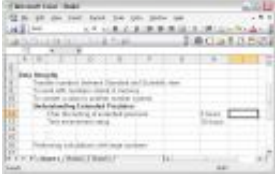
From here, drawing on our experience from testing the Calculator application from Windows 3.1 to 2000, we might determine that Average functional testing for any given feature takes one person 40 hours. Better yet, if we still have the testers who tested earlier version, we can have each of them enter their own estimates (much better than me guessing how long it will take). We might also know from experience that usability testing and compliance checking both average around 20 hours. If we say that our estimate is half of the average for simple types of tests, and double for more complex tests, we can come up with the numbers in the following figure.



The next step is to estimate some of the testing that may be new or more difficult to understand. For example, we may not have had 32-digit precision in past versions, so we're unsure what that means for our testing. Now things get more detailed.

You may want to add some tasks to better identify what this testing will entail. Will we need new tools or new methods? Do we have anyone on the team who can do this testing? Perhaps it will take a couple of hours to figure out some of this stuff, so we add a planning task. In addition, if we need tools (perhaps to

see memory values while we're testing), we have to get those installed and configured. We should give ourselves some time for that process. Therefore, we have two subtasks so far, as shown in the following figure.



This process continues until you have estimates for each step, up to and after the actual execution of the testing for this task. As you go through this process, think about other unique artifacts and efforts going on at the same time; think about test setup time, status reporting, management overhead, and so on. Depending on how the project is managed, these issues may all need to be recognized as a necessary part of the cost structure for the project.

In our estimation above, we've tried not to oversimplify (in fact we may have made the simple complex) and we tried to take into account the skills of the test team (have the testers actually estimate their work based on their own experience). What we haven't done yet is allow for rapidly changing focus or accounting for what we don't know.

Here's how I do that: *Don't put it in the spreadsheet.*

Don't add a contingency (unless you call it contingency – I don't mean you can't plan for the unexpected). Don't say "We don't know 10% of our stuff." Make it clear up front that you consider software development to be an intellectual effort where things change over time and that you plan on making continuous adjustments and plan on providing feedback in an ongoing basis for the duration of the project.

By not pretending to account for the things that no one can account for, your managers can better understand your *real* needs and will know to engage you in continuous communication as the project unfolds over time.

Comments

There are 4 comments for this post.

1.  [TesterQA](#) on [November 19, 2007 6:01 am](#)

excellent article

2.  [Amit Bhutani](#) on [January 14, 2008 7:19 am](#)

Hi, this is a nice article, but it's not complete unless one gets the chance to use this template. I arrived at this page after doing a regress search, but was disappointed as it's doesn't gave me help in terms of some artefacts. If it is possible for you to mail template to me on my above email id, it would be great!


3.  [Mike Kelly](#) on [January 14, 2008 8:14 am](#)

Amit,

The point of the article was that I don't recommend using templates for estimation. Instead, I provide an example of how I might approach estimation for a project and I walk through the steps that I take.

If you find that you've been repeatedly estimating the same type of project, then you may find that you'll save time with a template. I've found that I've rarely had two projects that were so similar that I would be able to reuse much documentation from the previous project.

Thanks,
Mike

4.  [Brian Osman](#) on [April 15, 2008 6:19 pm](#)

Hi Mike,

Great article! I've found estimation to be a tricky business because there are so many *what ifs*. One company i worked for used a simple spreadsheet for test estimation and insisted on using an *industry standard* 6 hours per day (i was sceptical of this figure as i haven't found any reference in relation to it) to calculate estimation. When i think back to it, i don't think we were even *close* to the estimate when testing finished (i know its an estimate but even a ballpark figure would've been more useful!. Often when the *estimate* was *calculated*, we were *held* to this timeframe which led to testers overly inflating testing effort just to be safe (sure you build some fat in but...). Another place wanted an *exact estimate* for each individual test case (100+ test cases) – the rebellion rose pretty quickly and the PM only just managed to avoid a coup d'etat! And yet others are not really interested and just plough ahead.

When i'm asked to estimate i tend to rely on my gut feel and then i remind myself that an estimate is exactly that! Thanks for the blog – have fun at CAST2008!

Write a Comment

Get The Latest News

Sign up to receive latest news

Interesting Topics

- [Categories](#)
- [Archives](#)