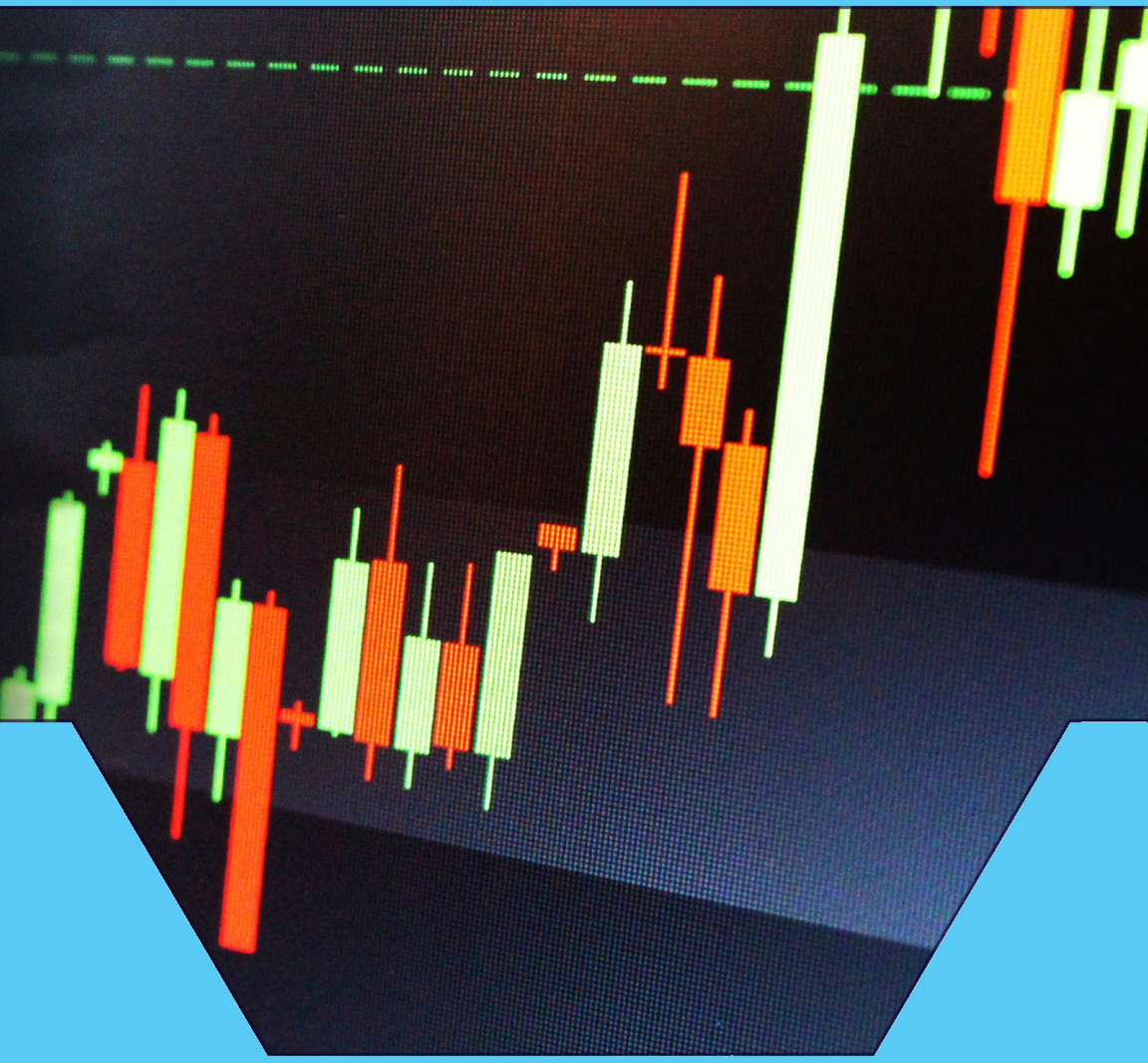
PREDICTING THE UNPREDICTABLE

Pragmatic Approaches to Estimating Project Schedule or Cost



AUTHOR OF *"MANAGE IT!*
YOUR GUIDE TO MODERN, PRAGMATIC PROJECT MANAGEMENT"

JOHANNA ROTHMAN

Predicting the Unpredictable

Pragmatic Approaches to Estimating Project Schedule or Cost

Johanna Rothman

This version was published on 2015-05-26

ISBN 978-1-943487-03-5

Practical ink

To everyone who was ever asked,
"How much will this project cost?" or
"How long will this project take?"

Contents

Acknowledgements

I thank my Managing Product Development blog readers. You asked me enough questions that I had to write the answers.

I thank my editors, Rebecca Airmet and Nancy Groth. I thank Jean Jesensky for her indexing for the print book. I thank Karen Billipp for her layout for the print book.

Any errors are mine.

7. Rolling Wave Planning

Project teams can use rolling wave planning to deliver interim milestones and then replan the next chunk of the project.

Here's how rolling wave planning works:

Loop:

- Plan what you know for the next few weeks (I use a 3-4 week rolling wave). If you're managing a traditionally planned project, make this as detailed a Work Breakdown Schedule (WBS) as you like. If you're managing an agile project, you may not have to do any more planning than what you already have done.
- As each week goes by, use the knowledge you've gained about the project to replan the already-planned weeks and plan the next week at the end of the current schedule.

Endloop

As the project proceeds, you'll replan frequently, but you won't replan a lot of the work.

7.1 Learn as the Project Proceeds

The idea behind rolling wave planning is that you can't know everything about the project in advance, so don't bother

trying to plan a lot in detail. Plan the next few weeks in detail, always staying about three to four weeks ahead of the project. In my experience, it's not worth trying to look more than four weeks ahead. Things will change too much.

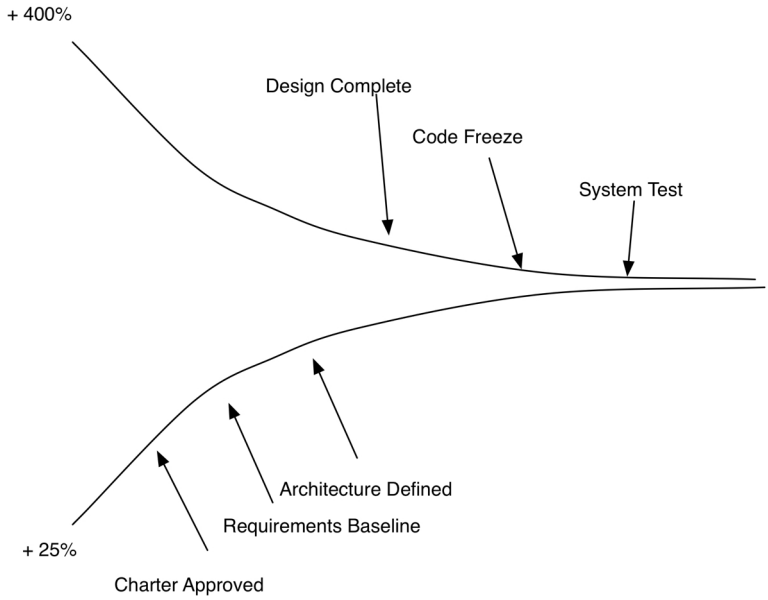
Of course, if you know you have hard dates like end-of-quarter or a trade show, put those events in the schedule. But rolling wave planning is much more likely to help you achieve those hard dates.

I incorporate adaptive planning into my rolling waves, by using the knowledge I've gained about the project to (re)organize the work as necessary.

If you haven't tried rolling wave planning, give it a shot. I find it especially helpful when I want to timebox to meet a specific date and I want an early warning if the date is impossible.

8. There Is No Correct Estimation Model

For years, we bought the cone of uncertainty for estimation—that is, our estimates were just as likely to be over as under.



Cone of Uncertainty

Laurent Bossavit, in The Leprechauns of Software Engineering (BOS), shows us how that assumption is wrong. (It was an assumption that some people, including me, assumed was real.)

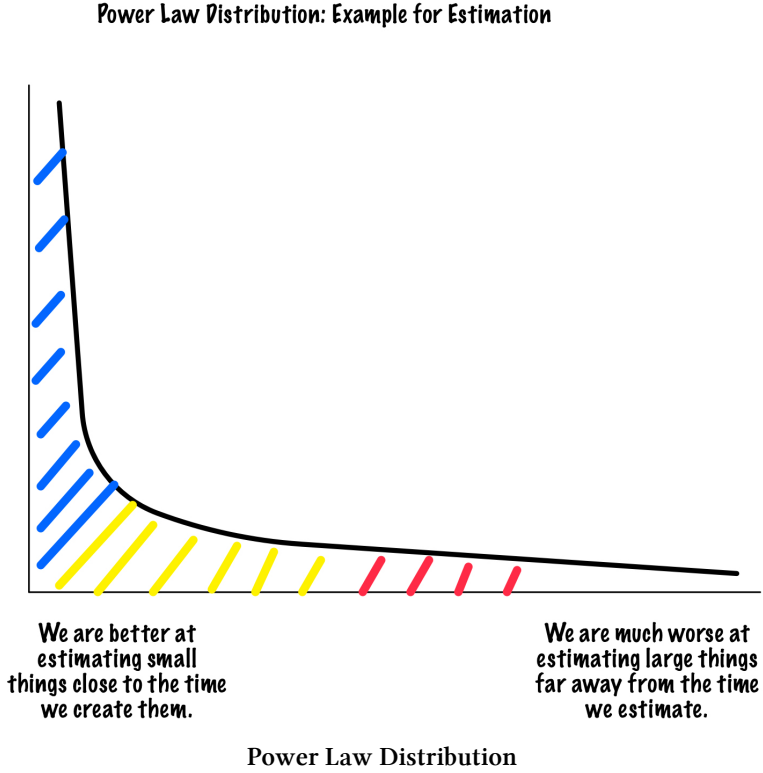
This is a Gaussian (normal) distribution. It's what we expect.

But, it's almost never right. As Laurent says,

> "Many projects stay in 90% done for a long time."

What curve do our estimates follow if they don't follow a Gaussian distribution?

Troy Magennis, in The Economic Impact of Software Development Process Choice - Cycle Time Analysis and Monte Carlo Simulation Results (MAG), suggests we should look at the Power-Law (Weibull) distribution.



Power Law Distribution

What this distribution says with respect to estimation is this: We are good at estimating small things. We get much worse with our estimation quickly, and for the long tail (larger and larger chunks of work), we are quite bad.

Why? Because creating software is innovation. Building software is about learning. We better our learning as we proceed, assuming we finish features. Maybe take another look at Software is Learning, Not Construction.

8.1 We Invent; We Don't Repeat

We rarely, if ever, do the same thing again. We can't apply precise estimation approaches to something we have never done before.

The question is this: What effect does understanding an estimation model have on our estimates?

If we know that the Gaussian (normal) distribution is wrong, then we won't apply it. Right, why would you do something you know to be wrong? You would not estimate large chunks and expect to have a +/- 10% estimate. It doesn't make sense to do that.

But what can we do? In Troy's paper, he says that if you have large, unique work items or you have large WIP, you will have poor predictability.

My suggestions for your estimation:

- Estimate small chunks of work that a team can complete in a day or so.
- Keep WIP low.

- Replan as you finish work.
- Watch your cycle time.
- No multitasking.

What should you do when people ask you for estimates? What kind of requirements do you have? If you have large requirements, follow my advice and use the percentage confidence, as in Estimating a Program.

You can predict a little for estimates. You can refine your prediction. And, you may have to predict a large effort. In that case, it helps to know what distribution model might reflect your estimate.