The Cost Estimation Conundrum

As a former System Program Management Office guy, or *SPO dog*, and now as a software engineering manager, aka *code toad*, there has rarely been a software topic that has given me pause more than cost estimating. Much attention has been focused on software projects that have gone terribly awry with budget and schedule overruns. These are truly unhappy situations that you never want to repeat. So what can you do? One approach is to apply a factor of ignorance, i.e., pad the estimate so that there is sufficient funding. Since estimation is by definition less than precise, this is ethically allowable, and besides, everyone does it. The trouble is everyone does it so managers know it and react to it. We may discount it to bid whatever it takes to win, or we may add to it to build a management reserve that we can blame on software. Either situation is not good.

We owe our customers quality software within cost and on schedule. Few things are harder for customers than trying to find more money for an over-budget project. Going to the well twice means admitting you were wrong originally, something most of us don't like doing. Then there is the often bitter debate about where to find the money. Another project will be taxed unfairly, or your project will be restructured, perhaps even be in danger of being terminated. Regardless of where the money is found, everyone loses. The customer didn't get what was needed when it was needed, and confidence has been lost in the project team's ability to deliver.

The sorry state of affairs that I have outlined above clearly needs to change. That is the emphasis of this issue – how we can do better cost estimating. In my view, we all have a great need for good cost estimation techniques. As a Capability Maturity Model® Integration Level 5 organization, WR-ALC/MAS is committed to constant process improvement. These techniques will aid in that endeavor, and they are beneficial regardless of maturity or capability level. I hope you will take the time not only to read about these techniques, but also apply them.

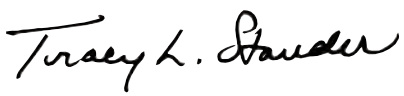
Thomas F. Christian Jr.
Warner Robins Air Logistics Center Co-Sponsor

Increasing Confidence in Estimates

Cost estimation is certainly one of the biggest challenges software managers face. With large software development or sustainment efforts, developers are increasingly dependent on automated tools to help quantify cost estimates. However, there is not one *silver bullet* modeling tool. As Capers Jones reports this month, a best practice for software cost estimation is to use a combination of estimation modeling tools in conjunction with project management tools.

This month's issue is aimed at increasing confidence in software estimates. Furthermore, industry experts discuss how several cost models are evolving to address technology and process improvements that impact the cost of developing military and commercial software today. William Roetzheim discusses project scope estimation and the difficulties early in the life cycle with indefinite requirements. Capers Jones defines estimation methods for large projects, including non-coding work. Carol A. Dekkers provides helpful guidance when working with incomplete requirements. Barry Boehm et al. present an overview of the Constructive Cost Model tool suite, which can now address commercial off-the-shelf integration, system engineering, and system of systems. Other authors address mapping requirements to units of work, an overview of the SEER-SEM model, and the reliability of lines of code to indicate software size.

I hope we've provided a better understanding of cost estimation, and how estimating models are evolving to keep pace with industry changes. As Lee Fischman et al. states, "The future of software project estimating has just begun."



Tracy Stauder
Publisher