Put a Tough Decision in Its Place

by Mike Cohn •

originally published in Better Software on 2006-01-01

In a recent discussion with a project sponsor, I asked whether it was more important to deliver on schedule or to deliver all of the desired features. The sponsor's answer was "Both." She refused to clarify for her team the relative importance of scope, schedule, resources, and quality on that project. She was insistent that all features be delivered on time, with no loss of quality, and without going over budget by increasing the team size.

I was gradually able to help her understand that this puts the team in an impossible position. I told her that every time a programmer passes a module to testing, he has made a tradeoff decision between scope, schedule, and quality. He has made a micro-level decision that the code is good enough, and that further improvements in quality are not justified by their cost in either a longer schedule or in delivering fewer features. I told her that these types of decisions are made dozens of times each day by all team members. And here's the kicker—I told her that, without guidance from her, these micro-level decisions were being made inconsistently. At that point, I was able to get her to prioritize scope, schedule, resources, and quality against one another.

This particular project sponsor was doing something I see happen much too frequently—she was pushing a difficult decision down the organization. If we cannot make a particularly difficult decision, what we should do instead is push the decision up—not down— the organization.

Multitasking, which I've written about in a previous column (see the July/August 2005 issue of *Better Software* magazine), is another example of inappropriately pushing a decision down the organization. The manager who assigns a tester to be on three projects is saying, "I can't decide which is the most important project for you to work on, so I'll make you decide." If the manager, who presumably knows more about the relative priorities of the projects, cannot decide how this tester should spend her time, how can she reasonably expect the tester to decide?

Of course, there are some issues that should be passed down the organization. This is the basis of empowerment. There is a fine line, however, between empowerment and shirking our own responsibilities. Here are three questions I use to help decide whether a decision should remain

with me, be pushed up, or be pushed down:

- **Who has the appropriate knowledge to make this decision?** Before passing a decision down, be sure that the employee has or can get all the necessary information to make an appropriate decision. If you pass it up, make sure your boss is in touch with the necessary day-to-day details.
- **Who owns the risk of a bad decision?** Is it my boss, my employee, or me? In the product sponsor's case, the risk of choosing incorrectly between scope, schedule, resources, and quality was hers. She needed to own that decision.
- **What would my boss or my employee think of my pushing the decision to the other person?** If I told my boss, "I just delegated this decision to my employee," would she agree? If I told my employee, "I can't make that decision; let me run it by my boss," would he agree?

Next time you are tempted to pass a difficult decision down the organization, stop and ask yourself these three questions. Similarly, next time a decision is inappropriately passed to you, see if you can get your boss to push the issue up instead of down.

Posted: January 1, 2006

Tagged: management, testing, prioritizing

About the Author

As the founder of Mountain Goat Software, Mike Cohn specializes in helping companies adopt and improve their use of agile processes and techniques to build extremely high-performance teams. He is the author of *User Stories Applied for Agile Software Development*, *Agile Estimating and Planning*, and *Succeeding with Agile*. Mike is a founding member of the Agile Alliance and Scrum Alliance. Mike can be reached at hello@mountaingoatsoftware.com. If you want to succeed with agile, you can also have Mike email you a short tip each week.
