When Project Managers should Step In and when they should Step Back

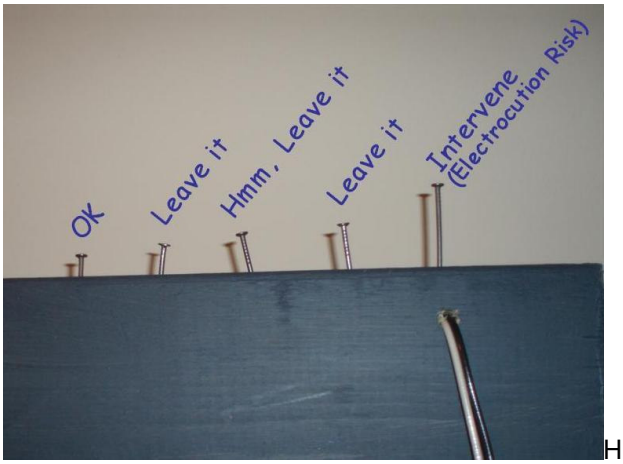
I had to stop myself today, I was about to step in and help the team work out an issue, but I recognized they were sorting it out just fine and it would be better to stay out of it.

Project managers working on agile projects know that they need to step back and let the team organize their work, resolve issues, and make decisions for themselves. This can be hard to do as the temptation is to "help" by organizing and directing people to achieve the project goals. However, this is not help at all as when the team has authority to make local decisions a step improvement in ownership and productivity occurs. The jury-is-in on this idea; teams should be empowered to make decisions; people volunteer for work and take responsibility for delivery. We tap into individual's capability to manage complexity and create a more rewarding environment for working.

However, project managers need to do more than the proverbial buy-pizza-and-keep-out-the-way. They should be removing impediments, providing resources, growing the team, communicating the project vision to stakeholders, and a host of other valuable tasks. One of which is stepping in to work more directly with the team if a major problem is encountered or impending. For example if the velocity of feature delivery is tracking too slowly over a number of iterations to meet the minimum required feature set by the project end date then something has to happen. Or, a business change or competitor product launch threatens the ROI of the project, clearly intervention is required.

Initially this balance between interfering and intervening can seem difficult to call. Should your alarm bells go off if team velocity drops one iteration? Should the team alone respond to external threats to project ROI? No and No. Fortunately there is a useful concept to help project managers make the right decision on when to step back and when to step in; and I am grateful to David Anderson for pointing me towards the work of W. Edwards Demming to understand this concept. http://www.agilemanagement.net/Articles/Papers/Agile_2005_Paper_DJA_v1_6.pdf

Deming promoted an idea that he called the Theory of Profound Knowledge (I guess he was not a modest guy) that talked about the variation that happens in a process. If you consider repeatedly hitting a nail, you never hit it exactly the same way each time, there are some variations. Deming breaks variations into Common Cause Variation and Special Cause Variation. Sometimes you may hit it exactly square on and others it may be slightly off to one side – this is just Common Cause Variation. However, if you break your arm then try hitting the nail with your arm in a cast, or if you are forced to work in a dark then the variance of nail hitting now has a Special Cause Variation, caused by these external factors.



His point is that the project manager should accept common cause variation, but look to intervene on special cause variation. Which can be reworded as: avoid micromanagement and instead work at removing bottlenecks and impediments. Asking developers why they have not coded 5 features this week when then completed 5 features last week is failing to accept common cause variation – this stuff just varies a bit and is not perfectly linear.

So focussing a lot of effort on tracking conformance to a rigid plan is not the best use project management time. Instead, look externally and to the daily stand-up meeting when the team reports any issues, impediments or blockers to their work, these might be pointers to special cause variations that need to be resolved.

Deming goes on to say that there are only two types of mistake a project manager can make:

1. To interfere with work that is varying within Common Cause Variation – this is micro management, like asking why the developers did not complete 5 features this week.
2. Not interfering (trying to fix) when a Special Cause Variation is detected – this is lack of service – the PM must help fix these external issues

Both of these mistakes lead to a reduction in team morale and seeing as the knowledge worker is largely driven by morale, since visible and tangible inventory is small, then projects really struggle when people are micromanaged or not serviced.

So when next trying to determine whether to step in and intervene or not, try classifying the issue as either Common Cause Variation or Special Cause Variation. If you can see the issue is really Common Cause variation then resist the temptation to meddle and instead reserve your energy for those Special Cause Variation issues.

Mike Griffiths is an independent consultant specializing in effective project management. Mike was involved in the creation of DSDM in 1994 and has been using agile methods (Scrum, FDD, XP, DSDM) for the last 13 years. He serves on the board of the Agile Alliance and the Agile Project Leadership Network (APLN). He maintains a leadership and agile project management blog at www.LeadingAnswers.com.

