

People on projects

VOLUME 2 | ISSUE 6 | JUNE 2004

The Project Management Best Practices Report

The Superior Project Manager

Why the Lean in Lean Six Sigma?

Applying process improvement tools to knowledge work

by Mary Poppendieck | A free-wheeling mid-sized company ran smack up against the Sarbanes-Oxley Act a few months ago and found that it had to pause and take stock. (See “Better Safe Than Sorry,” by Jeffrey Rothfeder, *CIO Insight*, February 2004.) The company commissioned a small team to map out every place that financial data moved, and to no one’s surprise, it uncovered a lot of uncoordinated, manual processes. Next the team put into place some simple automation that managed inventory and financial data, an online order entry system, and even some automated human resources capabilities. The company found that with these more streamlined processes, days were cut out of order processing, inventories were reduced and productivity was significantly improved.

The irony was that the fast-moving company had to slow down to speed up; more discipline led to higher speed. This is not an isolated case. There is a direct relationship between speed and discipline. Adding discipline should streamline a process, and streamlined processes don’t work without discipline.

Smooth and Speedy

When I first heard the term Lean Six Sigma, I wondered what Lean added

to Six Sigma. I found that the answer is speed. The first principle of Lean Six Sigma is: “Delight your customers with speed and quality.” The second principle says: “Improve process flow and speed.” Lean Six Sigma emphasizes that speed is directly tied to excellence.

Speed is not the same thing as schedule. Schedule is about when something is supposed to get done, speed is about

Speed has a bad reputation; it is often equated with hasty, undisciplined work. But if Lean Six Sigma has anything to teach us, it is that we should be looking for opportunities to streamline our core processes.

how fast it gets done. Speed has a bad reputation; it is often equated with hasty, undisciplined work. But if Lean Six Sigma has anything to teach us, it is that we should be looking for opportunities to streamline our core processes. This does not mean we should be compressing already tight schedules. It means that we first determine what our core processes are, and then focus on making them flow smoothly.

For example, core processes in software development would be naming conventions and coding standards, a configuration management system, an automated build process, a suite of automated unit tests that are built and maintained as part of the code, daily build/integrate/test cycles, acceptance testing integrated into the development process, and usability testing immediately after the features are implemented. Assuring that these disciplines are in place is fundamental to the smooth flow of any software development process.

However, the most important process to streamline in a development project is the knowledge creating process. Whether we are developing a new product or a new software system, the fundamental thing we are doing is discovering what needs to be in the system in order to delight the customer. Lean thinking supports two basic disciplines for speeding up the knowledge creation process: short, frequent learning cycles and delayed commitment.

Short, frequent learning cycles are the antithesis of thorough front-end



Mary Poppendieck

planning, but they are the best approach to processes which have the word 'development' in their title. In product development, for example, the typical process is to define requirements, choose a solution, refine the solution and implement the solution. Although this process is common, it is not the best way to generate knowledge.

Delayed commitment. Toyota has a different approach, one that is much faster and delivers products of superior quality that consistently outsell the competition. (See *Product Development for the Lean Enterprise*, by Michael N. Kennedy, Oaklea Press, 2003.) Toyota builds sets of possibilities to satisfy customer needs and then, through a series of combining and narrowing, the new product emerges. The combining and narrowing process is paced by time-boxed milestones that define stages of the narrowing process. Milestones are always met, despite the fact that there are no task breakouts or tracking. Decisions are delayed as long as possible, so that they can be based on the maximum amount of information.

When a project involves knowledge creation, rather than just knowledge replication, speed and quality come from improving the flow of creating knowledge. Many of our project management practices have a tendency to impede knowledge creation by forcing early choices and reducing the number of possibilities explored. This often leads us down blind alleys: after a lot of work has gone into a solution, we learn that it has

a fatal flaw and we have to retrace our steps and repeat a whole lot of work.

The key to streamlining a development process is to clearly distinguish between true knowledge creating iterations and iterations that lead down blind alleys. Knowledge-creating iterations explore multiple options and leave as many possibilities open as possible, delaying decisions until the last responsible moment.

Returning to our software development example, lean practices promote speed and flexibility by implementing core disciplines that promote change tolerance and allow decisions to be delayed as long as possible. Lean software development changes the focus from gathering requirements to encoding all requirements in tests. It introduces the concept of refactoring, that is, creating a simple design at the beginning of development to handle early requirements, and then improving the design later as more requirements are discovered. Finally, lean software development requires full testing and integration of code as soon as it is developed, on a daily basis or more often. The result is easily maintainable code that has been designed to be flexible and built to be rapidly changed.

In lean software development, scope is not set at the beginning; small features sets are added based on priority determined by their ROI. (See *Software by Numbers*, by Mark Denne and Jane Cleland-Huang, Prentice Hall, 2004.) This tends to lead to a significant increase in

both speed and productivity for a simple reason: most of the features we put into software systems are never going to be used. How can this be? When we freeze scope early, we encourage our customers, who don't really know what they want, to ask for everything they can imagine. When we delay commitment on scope until we are well into the knowledge generating process, we end up reducing scope down to the minimum set that is really going to pay off.

IF WE'RE NOT CAREFUL, Six Sigma might lead us to apply practices aimed at improving replication processes to our knowledge-generating processes. This often leads to slow, unresponsive, change-intolerant practices that are not appropriate for knowledge creation. Quite often the slow, deliberate nature of these practices is mistaken to be a sign of good discipline. But when we add Lean to Six Sigma, we discover that we need to re-think these slow processes, because we have come to understand that speed, discipline and excellence go hand-in-hand. ■

Mary Poppendieck, a Cutter Consortium Consultant, a seasoned leader in both operations and new product development with more than 25 years' of IT experience, is president of Poppendieck LLC in Minnesota. Her book *Lean Software Development: An Agile Toolkit* (Addison-Wesley, 2003), which brings lean principles to software development, won the Software Development Productivity Award in 2004.