

What does Lean Software Development really boil down to?

Anders Sixtensson, senior consultant and lean expert at SoftHouse Consulting, has collected a number of frequently asked questions from managers and others during his lean assignments and tutorials. In a discussion with Mary Poppendieck, creator of Lean Software Development, he tries to walk through some issues that always come up.

Mary, many teams claim they are “agile” or “lean” but often they miss some of the underlying understanding. Do you have any quicktest to check whether a team is really “lean”?

I would ask these questions:

Is the team a Whole Team – composed of everyone necessary to deliver value to the ultimate customer?

Does everyone on the team understand what customers really value?

Is the team focused on delivering small increments of real value to end customers?

Does the team reliably and repeatedly deliver on its promises?

I would also take a look at the team and see if it has both technical and marketing leadership. I believe that teams need a leader who understands and cares deeply about the customers and their problems. They also need a leader who understands and cares deeply about the technical integrity of the product.

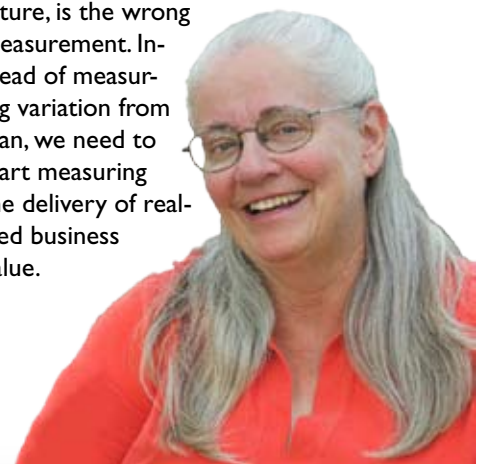
This leadership may reside in one or two people, or in small teams it may be distributed among several people. But teams which lack market and technical leadership tend to produce mediocre results.

I have heard that the traditional way of developing projects is called “plan-driven”, and these approaches are now being replaced by Agile. How would you say planning is handled in Agile?

I don't like the term ‘plan-driven’ when it is used as the opposite of Agile, because there is plenty of planning in Agile. However in Agile, there is no attempt to forecast the future long in advance and then plan as if that forecast were reality. Agile planning is based on cycles of discovery: do something, obtain feedback, and adapt. Non-agile approaches tend to be driven by forecasts of the future and they are notoriously unable to deal with reality as it unfolds. In domains where the forecasts are likely to be accurate, you can assume that the forecast is fact and devise a plan based on that assumption. However, when a forecast is a mere guess, it is far better to use an approach which adapts to the future as it unfolds.

Is there a “best way” for a company to switch from forecast-driven planning to the Agile/Lean approach based on feedback?

The first step in moving from forecast-driven projects to feedback-driven agile/lean methods is to change the measurements. The book “Rebirth of American Industry” by William Waddell and Norman Bodek makes a good case that the measurements imposed by traditional cost-accounting methods are the biggest impediment to the successful implementation of lean manufacturing. Similarly, I believe that the measurements imposed by traditional project management methods are the biggest impediment to the successful implementation of lean development. In particular, measuring conformance to plan, as if the plan is an accurate forecast of the future, is the wrong measurement. Instead of measuring variation from plan, we need to start measuring the delivery of realized business value.



OK, so after changing the measurements, what's next?

If a company wants to move from a forecast-driven development to a feedback-driven development, it needs to shorten the cycle time from customer request to software delivery, or from concept to cash. The shorter your delivery cycle time, the more responsive you can be to feedback. Queuing theory says that short delivery cycle times depend on having very short queues of work-in-process, so a good approach for switching to agile is to take a hard look at how much partially done work you have in your system.

Start by looking for churn (rework): if you have test-and-fix churn, you are testing too late. If you have requirements churn, you are specifying requirements too soon. Next look at the defect list: test-driven development finds and fixes defects before they need to go on a list. Finally, look for queues of work between departments: a cross-functional team can develop an increment of value from idea to deployment without using an inter-departmental list or queue.

How does a lean manager act in her daily business life? What is new and what should you remove?

I was the leader of a team that invented and commercialized a novel new product over a three year timeframe. The team was made up mostly of volunteers who could have left and returned to their regular jobs at any time. So I spent most of my time trying to understand what motivated each individual on the team and providing for that motivation. For instance, I made sure that the team members got to brag regularly to their peers and to their managers about the great things they were doing. We celebrated every step of progress and found ways to make it very visible.

I spent a lot of time walking around renewing in everyone the marketing and technical vision of the product. But I almost never told anyone what to do – in-

stead I set up weekly meetings where people figured out by talking to each other what needed to be done. At these meetings, team members made commitments to each other – and I found that everyone was much more likely to come through on a personal commitment than an assigned task.

Does a lean project manager commit to deliver a certain scope to a certain date?

In business, there is only so much money, and it must be invested wisely. And there are deadlines that cannot be moved and must be met. So project managers have an obligation to deliver the return for which an investment has been made, and to meet critical deadlines. However, the development team should have the freedom to discover the best way to deliver the expected business return within the deadline.

Cost, schedule and scope are sub-optimizing measurements. In particular, in software development, focusing on pre-defined scope is almost always a bad approach and usually leads to excess scope. Generally the best way to develop software is to deliver the minimum amount of scope that will provide the expected business value – no more. This is done by delivering minimum useful feature sets in priority order and deploying them to see what business value they return.

The right measurements for software development are delivered business value, cycle time to deliver that value, and customer satisfaction. If you charter a team to deliver the best value for the money within a timeline, then that's what you get.

How is testing and verification done in a lean environment?

In all lean areas – manufacturing, operations, retail, software development, – mistake-proofing the way the system works is fundamental. The job of QA in a lean organization is to create an environment that makes mistakes virtually im-

possible. In software development, this is accomplished by writing tests before development takes place, automating the tests, and then testing code as soon as it is written. Code is integrated into the code base as frequently as practical, and test harnesses are run against the integrated code to the maximum depth that is possible, as often as possible, depending on the domain. To be sure, there is still a place for intelligent testers to explore the code and find problems. However, rote manual testing long after code is written is a sure sign that the system of development is not lean or agile.

Finally, your new book is released now, which are the differences between your two books on Lean Software Development?

Our first book, *Lean Software Development*, is aimed at people who do not understand why agile development is a good approach. It provides the underlying justification for rapid, incremental development. The second book, *Implementing Lean Software Development*, makes the assumption that the reader has bought into agile development, and wants to figure out what to do next.

The second book is filled with things we have learned over the last few years: including answers to questions we have often heard and (anonymous) stories of enlightening situations that we have encountered.

Mary Poppendieck has thirty years of experience from the IT industry and has in recent years been one of the world's leading profiles in Lean Software Development. Her second book on the subject, "Implementing Lean Software Development", was published in September this year, and is available from all major online bookstores.