Table of Contents

How Do We Initially Setup Our Definition of Done?

Premise

Often Teams have trouble establishing their initial Definition of Done. Firstly, each person on the Team has a different perspective and all are correct. But the reality is that “if the customer does not get the deliverable then we really have not finished and we have not delivered value to the customer”. What this means is that we need a way to collect all these different views of what it means to be done. And secondly, we need to establish the discipline that before we have demonstrated or delivered something to the customer, that we have in fact done everything required to say this is a quality deliverable. If we don't have a common understanding of what it means to be done, we set false expectations and create Technical Debt.

Many people think that because you are doing all this work you are slowing delivery down. Reality is that re-work takes a lot more time than completing work correctly in the first place, but the work is often hidden and so it really just seems this way. Bottom line is that the Lean discussion has taught us that if you focus on quality you will increase how much you deliver over time. This is a classic example of “go slower to go faster.”

> “A stronger 'definition of done' will always increase velocity and improve quality” - Jeff Sutherland at Agile 2008

Background

Purpose of Definition of Done is that in every increment (iteration, program increment, etc. with different definitions at different levels) the system is potentially shippable to the customer, that there is no known remaining work to be done. The idea is that we want the decision to release the product / solution to be a business decision that can be made at any time, rather than an issue that is raised when one of customers say they want something (“Can I have it now”; “What! No! We haven't finished - it only works on this machine …”)

More importantly we want to release value knowing that we have completed all the work and not pretend to ourselves that we will come to back to it later. After all we all know LeBlanc's Law:

> Later = Never

Creating Your Definition of Done

The basic idea to get your initial Definition of Done is to get everyone's perspective on what it means to do complete, quality work from their perspective. Perhaps the analyst has one view, QA another, the architect a third, and the customer another view.

This leads to the following facilitated event:

- Brain-write down candidate items for definition of done, with one idea per sticky-note. Make sure people consider their complete perspective first, then consider other stakeholder. Again, this is aimed at “potentially shippable” software – include work required before released into production. Note: These are not features of the product but rather aimed at “intrinsic” quality:
    - It could be a part of a process. For example, code review, security review, etc
    - It could be a deliverable example: For example, an automated test, end-user documentation, etc.
- Have each person read out each other their items, stick them up on a wall.
- Affinity map like items
- For each item review and discard those that don’t have value. Ask questions like
    - “Does this make sense to do?”
    - “Do we all agree this is part of our DoD?”

Once complete, this is your Definition of Done. Document it and make sure its part of every planning session you have.

How Do We Ensure Done is Actually Done?

Once we’ve defined “done”, how does it actually get done. Lets look at the Team level (similar thinking can be applied at the Train level). At the heart of this work is a Team who is going to do the work. Generally Teams will break down the work in the form of Tasks. Now some of these tasks might come from the definition of done. For example, perhaps an automated test is part of the definition of done for every Story. Then their might be a task that says “develop automated test” in this way as you are working the board you are insuring quality every step of the way.

What this effectively means is that you treat your Definition of Done as a checklist. For each item of work that comes in you are “does this DoD item make sense for this story?” If yes, add a task to the board. If not, continue. And so on.

Evolution of the Definition of Done

The Definition of Done does not remain static. The Team is always looking at ways to improve how they deliver quality and, as they get better, they will adjust the Definition of Done. So it is a living definition that you will revisit on a regular basis.

Sometimes you will encounter the situation where you would like to improve the Definition of Done, but as yet you cannot as you don't have the capability. For example, perhaps your Team has never done automated testing before and now would like to do it. In this case your could create a Feature or Story that defined the work to figure our how to do work (automated testing in this case), and the acceptance criteria might include something like “Must become part of our Definition of Done”.

Consultant, Tools, DefinitionOfDone, DoD, FirstSprint, FAQ