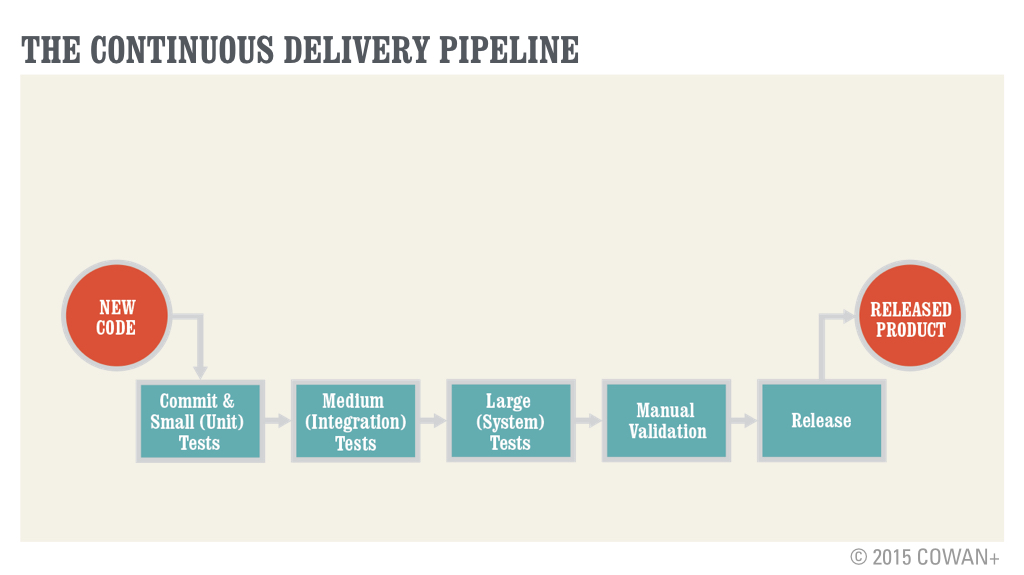
# Your Continuous Delivery Assignment: Tutorial & Facilitator Guide

You learned in the module ‘Does it break?’ that one of the most important foundations/first steps to building your continuous delivery capability is to develop a shared understanding of your delivery process. It doesn’t necessarily have to be very detailed, but it should describe the major steps from code inception to release and it make it easy to discuss what’s hard, what takes time (and the reverse).

A typical way to start this is getting your team together to grind out a view of your process on a whiteboard. If that option is available, I’d start there. If you’re working solo or otherwise can’t get your team together, patch it together as best you can. Try to talk to someone in a development role, in test, in ops/DevOps about their work and how code moves from being written to being released. If you end up having to take pieces from different projects, that’s still helpful.

Your diagram may or may not resemble the one below, but I would take note of:

* The idea that the processes and infrastructure you’re looking at are those between new code (input) and product released to customers (output)
* The various general steps which you may want to use as a guide to stepping through your process



In terms of the diagram, you can submit a photo of a whiteboard (make sure it’s clear) or you can put something together in Balsamiq, LucidChart, OmniGraffle, (yes) Power Point, or whatever you like to use to diagram. Just remember, it’s a sketch for explanation purposes. Don’t worry about making it look nice.

You should also submit notes on what you think is working well (and is a candidate for doing more of), what’s not working well (and is a candidate for trying something new), and, most importantly, you and your team’s thinking on how you might improve the delivery process. You’ll find a ‘facilitator guide’ below. The idea with that is to help you frame, focus, and facilitate the discussion. The elements of the guide are just suggestions--feel free to use them or not depending on what you think will get you to the best outcome. (If you took Course 1, hopefully the line of questioning has some familiar attributes.)

Facilitator Guide

|  |  |
| --- | --- |
| **Area** | **Notes & Questions** |
| **Introduction** | This is just an example but you may want to hit on some of these points:  Our first goal here is just to get a shared view of how our process works from the point where someone commits new code to where we’re released product. Our second goal is to look at the rough parts- the parts that are eating up big chunks of the team’s time, parts that seem like they should be better. Our final goal is to look at where we want to focus next on improving, be that changes to our process and/or changes in supporting infrastructure, like investing in more automation.  I’m not trying to sell any particular point of view here, just that it’s worth our while to look at this as a team and think about how we keep making it better. |
| **Commit & Small Tests** | Establish what you mean by small/unit tests.  How does this work now?  How does the commit process work?  What happens next? Who does that?  What’s our point of view and practices on unit tests?  [For each or other discrete topics that come up]  Which are the top 3 best things that we do or that we’ve learned work well for us? Are we doing as much of that as you think we should now?  What are the top 3 hardest, most annoying things about this process? Which take up a lot of your time?  If you had a whole day free to work on improving this area, what would you do? What if you had a week? What other support or infrastructure would you need to do it? |
| **Medium (Integration) Tests** | Establish what you mean by medium/integration tests.  How does this work now? What are the inputs? What are the outputs? What happens next?  Who’s involved? How?  [For each or other discrete topics that come up]  Which are the top 3 best things that we do or that we’ve learned work well for us? Are we doing as much of that as you think we should now?  What are the top 3 hardest, most annoying things about this process? Which take up a lot of your time?  If you had a whole day free to work on improving this area, what would you do? What if you had a week? What other support or infrastructure would you need to do it? |
| **Large (System) Tests** | Establish what you mean by large/system tests.  How does this work now? What are the inputs? What are the outputs? What happens next?  Who’s involved? How?  [For each or other discrete topics that come up]  Which are the top 3 best things that we do or that we’ve learned work well for us? Are we doing as much of that as you think we should now?  What are the top 3 hardest, most annoying things about this process? Which take up a lot of your time?  If you had a whole day free to work on improving this area, what would you do? What if you had a week? What other support or infrastructure would you need to do it? |
| **Manual Validation** | How does this work now? What are the inputs? What are the outputs? What happens next?  Who’s involved? How?  [For each or other discrete topics that come up]  Which are the top 3 best things that we do or that we’ve learned work well for us? Are we doing as much of that as you think we should now?  What are the top 3 hardest, most annoying things about this process? Which take up a lot of your time?  If you had a whole day free to work on improving this area, what would you do? What if you had a week? What other support or infrastructure would you need to do it? |
| **Release** | How does this work now? What are the inputs? How do we decide when we’re ready to release? What’s on the checklist? Who checks it?  Who does the update?  How do we make sure it’s working? How do we roll back if it’s not? Who’s involved in assessing and resolving issues?  [For each or other discrete topics that come up]  Which are the top 3 best things that we do or that we’ve learned work well for us? Are we doing as much of that as you think we should now?  What are the top 3 hardest, most annoying things about this process? Which take up a lot of your time?  If you had a whole day free to work on improving this area, what would you do? What if you had a week? What other support or infrastructure would you need to do it? |