CULTURE / DATA SCIENCE / SOFTWARE DEVELOPMENT

Mary Poppendieck on Why You Should Just Burn Your Backlog

23 Dec 2020 6:00am, by Jennifer Riggins

Jennifer Riggins both hosted this lean coffee event and is a co-organizer of Agile Tour London.

“People have become way too comfortable with backlogs. It’s just a bad, bad concept.”

Co-author of the “Lean Software Development,” Mary Poppendieck warned attendees of Agile Tour London in an end of day lean coffee that backlogs are a sign of an inefficient team. But that’s OK. She says that since the output is almost always measured consistently by almost every organization, and that output is reasonably consistent over releases, that’s your only metric that matters.

Without a backlog, teams and even organizations stay focused on the opportunities they want to pursue right now — or within the next six weeks.

With this rate-based system, Mary says you can focus on managing throughput, workflow and immediate value, not efficiency.

The Essential Metrics of Software Must Be Throughput and Capacity

accepts will be done. If she only writes seven, she's not keeping up pace. And if she ever writes nine, she might want to start accepting nine articles a month. But she manages her entire job by the fact that she writes about two articles a week.

"Every single software organization can take a look at the number of units of things they put into production in a window of time. I've never met one that can't," Mary explained.

She said you could do this project-sized, but typically it is small or medium-sized deliverables. How many of those units do you actually deploy in a week or a month? Most companies know this or at least can make a pretty good estimate.

Mary refers to this as the **throughput rate**.

She said, "You have a history and you have a good feel for it, if it's anything like what you've done before — the rate at which your group of people can get that thing done."

Once you've established your current output rate, then you have to stop accepting anything beyond that rate. No backlog.

If you can only do ten a month, you only accept ten a month. And you don't accept work beyond a month or two.

Then Mary says you put those ten units in a queue, "but don't waste your time prioritizing" — it's first in, first out. You want to have as small a buffer of things waiting to go out as possible.

She further explained this flow system as: "Accept that as the rate that you put out. You don't estimate. You don't prioritize. You make a decision when somebody asks you to do something if you have the capacity to accept it, and your buffer in between is so short that you don't have to decide what to do first, second and third. You do it in the order coming in."

Mary says you're not worrying about tasks.

"You're worrying about: Am I still going ten a month? And am I accepting only ten a month? And do I have as short of a first-in and first-out queue as I can have so that I get stuff done fast?" she said.

Could a Focus on Flow Make for Happier Teams?

She says you will always have more demand than you can do, so you have to take the power back and say "Yes, and it'll be done in six weeks." Or "No, I just don't have time." Express how you'd love to help them and you can brainstorm with them other ways to solve their problem, but you will not add it to your plate.

Mary emphasizes that you always can save space for emergencies, but fundamentally, you have only three options:

- Now
- Next
- Never

When you do that, you are moving things through that are very current — just the stuff that you were just asked to do. Then you make a decision on the input flow and you manage at a rate. There's no task management. No worry about task size.

Whenever you have a backlog, you can track, say in Jira, when tasks came through and when you actually did them. The goal is to shrink that time.

"What you want to do is have very new stuff that's waiting to be done and a way to make that decision right away," Mary said.

She says it's not just nicer to the people you are managing, but for the customers too. They won't be dealing with an unknown wishlist anymore. They will always be certain with one of two answers:

- Yes, and it'll be done in six weeks.
- No, I'm sorry that isn't something we are going to do.

Keep your teammates free enough to manage their own input queue.

> "If people ask you to do something and you don't have the capacity, saying 'I'll put it on my backlog' does them no good and it does you no good. It's just a lie. You know your output capacity. You know the rate at which you get stuff done. And that's only as fast as you're going to move." — Mary Poppendieck, "The Lean Mindset."

capacity, output is a measure of demand. At full demand, output becomes a measure of capacity.

She said, “Since almost all software engineering organizations are operating at full demand almost all of the time, output is primarily a measurement of capacity, not of performance.”

Mary made sure to re-emphasize that an individual or team’s output rate should never be treated as a performance metric.

She said, “It is an insult to teams to assume that they are not working as hard as they can, so if demand exceeds capacity, a team’s output should be assumed to be its current capacity.”

Mary continued that managing the capacity of an organization is one of the primary jobs of management. In the common case of demand exceeding capacity, it becomes the team lead’s job to not push that demand down not the teams. Excess demand creates an overflow or “thrashing” situation which in turn decreases capacity.

It’s also the job of management to increase the capacity of the team by finding and removing any bottlenecks.

“If a team does not seem to be delivering output ‘fast enough,’ managers should look at why the system is slowing things down, not why the team appears to be slow. Because capacity problems are almost always a management problem,” Mary said.

Flow Management Is the Best Way to Respond to Complex Systems

“Lean Software Development” co-author Tom Poppendieck joined his wife for the discussion. He pointed to how complex systems thwart being able to predict what will work.

“That’s what makes it complex — there are too many unknowns and interactions.”

Tom says the only way to approach this is through experimentation in an effort toward progress.

> “Anything that you do in a big batch, a big commitment to a long list of tasks, is going to be wrong for most of those tasks. And that is especially true when you have kinds of systems you’ve never done before or the complexity of the domain is high.” — Tom Poppendieck, “The Lean Mindset.”

1. Pick a candidate.
2. Invest a little in it.
3. Get feedback.
4. Observe how it works in your environment.
5. Adapt.
6. Decide to continue or move on.
7. Start again.

Mary says in this rate-based system you are done when the time is up or very close to when the time is up.

"Working against hard deadlines as opposed to the task complete and doing best efforts against a deadline is the classic engineering approach," she said.

She pointed to SpaceX's work trying to make reusable parts for booster rockets. For a while there, it was crashing every three to four months with each launch. But that launch date never changed. Once you know the throughput of your team, Mary says you can divide larger projects into smaller ones, but that deadlines never change. The engineers have a few different options that they test out and move ahead on delivering.

"Those deadlines don't change and the teams bring forward their best effort by that deadline," Mary explained.

If you're working in this sort of rate-based system, you can have a little leeway on either side, but you are finished when the deadline is there. You accomplish the best you can in the given time period for that problem at that time.

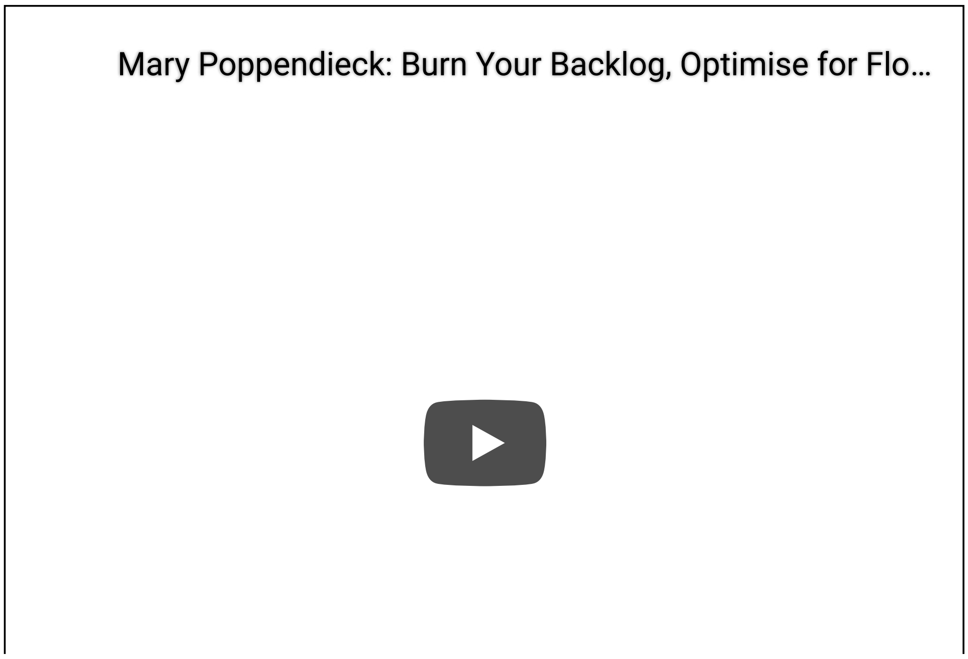
Tom says it comes down to the difference between push and pull. When somebody gives you a deadline, they are pushing a list onto you. But if they are saying that every three months your team is going to release something with better versions each time, that's pulling.

He said, "Pull is much safer. It's a lot faster. And there's much less waste because everything you do is focused on getting your part to do what it's supposed to do in the context of all of the other evolving parts by fixed dates."

There is one caveat — open source communities. In that follow-up email interview, Mary said open source software doesn't follow the same pattern. In an open source community, a long backlog is actually a good sign that people are active users who care about it, so they want to find bugs and to continuously improve it.

"This only happens in healthy open source projects. In open source, capacity is a foggy idea — for that matter, so is output. The most important thing is demand, and given enough demand, there will be plenty of people to fix bugs and make changes — because they improve the project for their own purposes, not so much for the project," Mary wrote.

View the entire presentation here:



Open Mary Poppendieck: Burn Your Backlog, Optimise for Flow, Pandemic Response, AI and Future of Work on YouTube.

PROFILE

More From Our Sponsors

bmc **Gene Kim Talks DevOps and the Mainframe**
29 July 2022

KASTEN by Veeam **Protect Kasten K10 Snapshots Using AWS KMS**
25 July 2022