Table of Contents

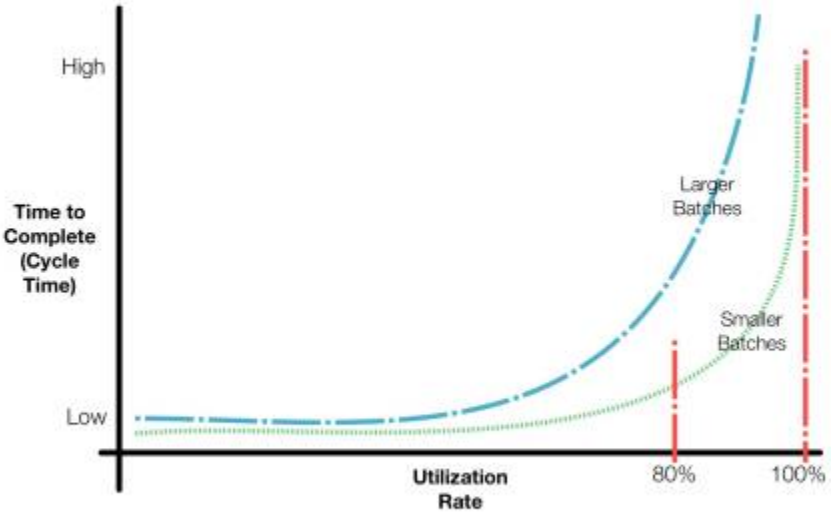
What Is The Effect of Batch Size On How Long It Takes to Get Something Done?

Premise

I've been working in the software and IT industry for a lot of years. One thing we'd do is that we would work a yearly plan for budgeting purposes. September would be the start of the process. There would be big meetings to get everything figured out, and final approval would happen a couple of months in the new year. We would them be held accountable for meeting these plans. Of course these plans would need to change as, during the year, the situation changed - competitors released capabilities that we needed to respond to, customers changed their direction, business conditions changed, and so on. In many cases the old plans never really got re-worked, they were just assumed, and we'd end up delaying the delivery of these capabilities.

I never really thought about the effect this process had on our ability to deliver. It seemed this process made sense. We are a business. We needed these plans. But I suspect if people really understood the effect this process was having on our ability to deliver they might have thought that we should rethink the process.

Understanding Real Cause and Effect of Large Batch Sizes



I want you to have a look at the chart[1)].

Basically what this tells you is that for the same utilization rate (see What Is Wrong With 100% Utilization Thinking? for more on this factor) the larger the work you have coming in, the longer it will take for you to get it complete. Again, this is an application of “queuing theory”.

Interestingly, the traditional software development process is all about having large batches of work go through the system. Think about your planning process (yearly budgeting cycle where everything is approved at once). Think about your contracting process. Or just about any approval process, where waiting on completion before approval happens forces larges batches of work. Our ways to manage projects is traditionally based on large batches of work. And when we combine this with high utilization rates is it any wonder that our projects are often in trouble?

You instinctively understand this effect even if you have not applied this to software. If you have a road system filled with cars versus a road system filled with trucks and you are a car trying to get from point “A” to point “B”, which road system are you more likely to move quickly on, and which road system is more likely to slow you down - the one with the cars or the one with the trucks. Big things moving through systems tend to slow things down.

Now What Are You Going To Do?

One of the reasons that agile approaches work at the team level is that they help you reduce the size of work we bring into the system (a team in this case). It does this at multiple levels. For example:

- We deliver thin, complete slices of functionality in the form of user stories - a small “feature” batch.
- We deliver in 2 week sprints / iterations - a small “time” batch.
- And so on

This is the kind of thinking that you can apply up and down an organization, down to a person, and up to a portfolio and enterprise view of life.

For the discussion mentioned in the “premise” what we might want to do is set things up so that we release money to do work on a quarterly basis instead of a yearly basis. This would result in smaller batches of work and would probably provide other benefits as well. For example, we could make decisions to change direction in response to changing business conditions quarterly instead of yearly. This would have the additional effect of reducing the amount of emergency situations we had to deal with - we could deal with changes that need response quarterly as part of normal business. Since we are planning for a shorter time horizon, we are planning less, can spend less time working the plan, and be more responsive. The quarterly approach would be a good start. We could then work to monthly, weekly, daily … Can you say “DevOps”? And finally, think of the level of additional control you will have over the budget if you do work in quarterly increments instead of yearly - 4 solid check points instead of 1.

Note that I am not suggesting that you magically just say “do quarterly planning”. The reality is that, in any decent sized organization this kind of change will require a bunch of work to make happen. What I am suggestion is that you need to start thinking this way and take the steps required.

Enterprise, Utilization, BatchSize, RiskManagement, FAQ, PresentationIdea

~~LINKBACK~~ ~~DISCUSSION~~

1)

Basis of chart results come from The Principles of Product Development Flow: Second Generation Lean Product Development - Don Reinertsen although it has been liberally interpreted to aid in understanding.