Continuous Integration (CI) and Deployment (CD) is here. Who wants Scrum?

I happened to be at a social gathering last weekend. A VP of a software product team and a good friend of mine was elaborating on how her team does Continuous Integration and Continuous Deployment and does not need Scrum. Her team found the Scrum time-box - Sprint, too restrictive in product delivery. Why wait for Sprint to complete in 2 weeks when you can deploy something today?

“How many times do you deploy in a week?”, I asked her. “A week?”, she almost exclaimed, “We deploy thrice a day. We deploy with every check-in.”

“How much effort does it take to do the deployment?”, was my next question. “None. It is all automated. As soon as there is a check-in, our scripts run automated tests and automated deployment happens on its own on Salesforce Platform.”

“Sweet! How big is the product? How many instances do you have? How many customers do you have on each instance on Salesforce? Do your deployment scripts deploy updates on all instance all at once or is there disparity in the instances? What if some customers do not want to update so frequently? Do you bring down the instance when you deploy, or have ability to update live?” My flurry of questions poured down on her.

She responded, “ We just released MVP of our product last month. We do not give our customers a choice. We deploy on all instances whenever we are ready. We have it in our customer agreement. Our customers expect that. We currently have about 40 customers on 10 instances. For some updates we do need to bring down the instance. We provide 15 minutes notice to the customer before bringing the instance down.”

“There,” I said, “is your sweet spot. You are nimble and have a small enough product with not a very diverse and large customer base. Additionally, your user agreements are simple and straightforward. However, think about how would you scale this model, when you have a mature product with large customer base? What eventually plays into this, is the cost-effectiveness of deploying thrice a day vs. once a week. Secondly, on a scale larger than the MVP, how do you ensure that every deployment has only the code that was reviewed and tested and no other code?”

“She got thinking now, “Hmmm… You are right. We might end up branching the code or have feature switches, which will take us away from continuous integration at the current frequency.” “Exactly,” I said.

“Additionally, how do you make sure, your team is not repeating the same mistakes, is working towards continuous improvement of work culture, team dynamics as well as product? Does the team have a retrospective at some point? How often is that?”

“We are in startup mode. We do not have retrospectives. Our goal is to deliver the MVP to the market as proof of concept.” she responded. The conversation ended at that point as we got pulled into another conversation, but the thought lingered.

Continuous Integration and Deployment work well with a completely new product with architecture catered to deployment on new technology, with very small deployment-set and a small customer-base. As soon as the company grows, the product grows, and the customer base grows, a more robust framework is needed to plan the work. Another way to look at continuous integration and deployment is a Sprint that is one day long. You start the day with planning what you will accomplish today, and you complete the day with a working feature that is checked-in and deployed. As we know Sprint length is usually 1-4 weeks, but it is up to the team to decide. If a one day Sprint works for some very high-performing team, it is good. However, there should also continuous stake-holder involvement to build the right product and continuous product review & team retrospection to go with it, in order to have a sustainable continuous integration and deployment. So, in reality do one day Sprints work? The tipping point of when a Sprint is too short for increasing ROI, varies from by domain, customer base, product size, product maturity, technology, culture, company size, and many such things.

As always, thoughts from the readers are welcome.