BABY YODA AND STRANGER THINGS: THE CASE FOR SHORTER MAINFRAME SOFTWARE RELEASE CYCLES

Overview: Careful thought must be paid to how we want our users to consume our mainframe software updates. Releasing incrementally has definite advantages over big releases. Just look at how we consume popular TV shows.

BABY YODA! Now that I have your attention, let's consider how Baby Yoda became a big thing over the last few months, growing with each new episode of Disney's *The Mandalorian.* Baby Yoda was able to stay in our minds for the period that the episodes where dropped weekly. What Disney did here was smart: they put out incremental weekly releases of the show. This meant that we couldn't binge; we had to watch each week and wait for more. We had time to think about and discuss each episode. Each episode could stand on its own and have a week to be dissected before the next was presented. It provides for a much longer period in front of the public.

Contrast that with a series which is dropped all on one day like Netflix's *Stranger Things*. This produces a big buzz, but mostly for a few days until everyone has seen it. The interest does not last as long as if it were stretched out over several weeks. There isn't the time to review and discuss each episode because you wait until everyone has caught up and then you can discuss the whole season.

These two different strategies to release a series affect how we consume them. It determines the attention, the discussion and especially the length of time the show is in focus. What does this have to do with software development? It concerns how we release features. There is a tendency on the mainframe to do the all-in-one-day drop. The features may be ready sooner, but we wait and put

them out all at once. This gets a lot of attention when we drop the release. We can use that focus to get the attention of the end users on the fact that there is a new release and there is a lot in it. But that's the thing, there is a lot in it and many details will get overlooked. It becomes overwhelming.

Breaking the Bingeing Habit

We really have to think about how we want our users to consume our software updates. We don't want our users to be overwhelmed. We would want them to understand and benefit from what we have provided.

Here are some reasons why you should move to smaller, more frequent deliveries:

- **Each release can be understood.** Users will be able to quickly understand what has been delivered and be able to try each of the new features. Your valuable improvements won't get lost among the others.
- **Faster feedback.** As you put out releases you will get feedback sooner to use to fine tune your upcoming releases. You put a lot of effort into each improvement and need to understand if users are adopting the changes and they're getting the benefit you intended.
- **Easier to support.** Having a smaller defined release can make it easier to support. This is because you can determine when an issue was introduced with certainty and you limit the number of variable factors involved. Also going back to the previous point of faster feedback, it can enable you to provide a resolution sooner.
- **Build anticipation and users will see there is a plan**. There is something to be said for anticipation. Incremental delivery means that users will receive some benefits sooner, they will feel part of the process, and they'll look for the next improvements.
- **It's less disruptive to make incremental changes than a big one.** We will often find it easier to block out an hour for a new episode than a weekend for a binge. It can be the same with the changes you provide. Users can allocate the small amount of time to try out a new feature, one at a time.
- **Lastly, it provides choice.** When shows are released weekly many will watch them before the next installment, but others will wait to binge on them all. Depending on how you choose to install, you may have the option to implement incremental improvements as they come, or hold off and batch up a few releases to install. It provides a way for you to install smaller releases in test and learn from them, and then make the decision as to when they will be rolled out.

Just like the networks, we have choices in how we deliver our creative products. With all the choices we make in our development, the delivery frequency can be one of the most important. It can determine how the users will see and interact with what we provide. It's a decision that needs to be made up front because of the ramifications to the development lifecycle. Just like *The Mandalorian* and Baby Yoda are best consumed (figuratively, of course) over time, regular, incremental improvements have clear benefits over one big software release.