Predictability and Agile projects: Are they Oxymoron?
Balasubramaniam Muthusamy, Sr Program Manager
Yahoo! Finance, Bangalore mbal@Yahoo-inc.com

Yahoo! Finance is the premier website in finance world. This paper shares the experiences in Finance modernization projects, where the challenges in release predictability of Agile projects were managed by focusing on key aspects that helped bring in predictability and still adhere to Agile core principles.

In the enterprise world, majority of the product releases are driven by road maps, time bound, defined early in the cycle. Where as Agile teams owning a product release wouldn't have enough visibility at the beginning to commit to these dates, especially on large projects that would span few quarters to launch.

Too much of upfront analysis and design would make projects water-fall-ish. At the same time, being pure Agile focusing one sprint at a time and no upfront, is big challenge for predictability and commit release. So, there is always conflict of being able to predict release date early in the cycle and still adhering to Agile principles. For this very reason, Agile projects and predictability often becomes oxymoron.
This experience report focuses on the journey of connecting these extremes in Yahoo! Finance site modernization projects.

The following focus areas helped us make the Agile projects more predictable:

1. Just enough design/requirements and release planning
2. Velocity and optimize team capacity
3. Requirements mind set

1. Just enough design/requirements and release planning

Typically in waterfall projects, Requirements and Design phase would be approx. 30-35% of project duration before the development starts. This is like boiling the ocean. At the same time, in purist Agile the requirements/design are handled one sprint at a time. In the enterprises setting, it is critical to strike balance between these two extremes and define scope, size it and perform release planning. This was 13 sprints (2 weeks) project and approx. 7-8% time (2 weeks) was spent on putting together just enough requirement/design, sizing and release planning.

The initial upfront time for Just enough requirement/design definition was very essential:
- To define product/release scope: Loosely defined, one liner backlog items leads to confusion. Just enough requirement capturing the boundary of the feature such inputs, outputs, hand drawn mocks/wire frames.
- To identify long pole dependency items, with platform/foundation teams as
the products are not built in isolation in large-scale enterprises. The platform/foundation teams support the needs across the company. It is absolutely critical to identify these dependencies upfront and put it in their road map.

- To size the requirements: Sizing was very important to determine team sizes and release plan to meet expected release date. This helped to stitch the work involved, capacity and arrive at high level sprint plan. This enabled us to commit the release date with 70-80% confidence level.
- To assess the architectural/design risks early.

2. Velocity and optimize team capacity

Velocity is an indicator of the rate at which the team is delivering. This metric becomes even more powerful when combined with team capacity. Unit of measure for the requirement size, team capacity can be anything that team feels confident and comfortable such as Story Point, Ideal Days, OOUM (Our Own Unit of Measure).

We had used simple and very common representation for velocity using Cricket analogy to help everyone relate the metrics/ velocity like Current run rate, required run rate. During initial high-level sprint planning, we arrived at planned velocity to hit release date by prioritized user story size and team capacity by sprint. As we moved forward, the actual velocity was tracked and used to arrive at the current run rate (cumulative velocity) and required run rate (required velocity per sprint). After Sprint 5, the required run rate increased to 53, where as current run rate was at 24. Graphs to show Release burn up chart and Velocity run rate can be found in the appendix below.

The increase in required run rate (more than double, 24 to 53) to reach the release target was translated into several actionable items proactively such as step up team capacity or revisit/resize user stories or alter what is in release backlog keeping MVP – Minimum Viable Product. MVP was very useful practice that helped us to be on top of what are the minimum feature set/requirement it takes to launch the product. This is subset of the product backlog that defines the must have items to launch the product.

3. Requirements mind set

This is no brainer that the collaboration between Product Owner and the team is key success factor for any Agile projects. Even though the collaboration between Product Owner and the team was effective, the team was still into the mind set of requirements (what ever it takes, deliver everything stated) than analyzing the optimal way to deliver faster and quicker by brain storming what users want and effective trade off decisions.
This realization surfaced some where around Sprint 4 of 12, when the required run rate started creeping up. As first step, like every other team we explored the option of stepping team size. The important break through came in when team started debating why do we need to put in 5-6 sprints (35-40% of the project duration) for some of the pages where the current page views/traffic is only 2%. It didn't justify the effort that we need to put into modernizing these pages which has over past few years attracted very least traffic.

The team and Product Owner had revisited all the requirements related to these pages to simplify engineering complexities by effective trade off decisions and still delivering similar user experience. After this wonderful, turn around event, these pages were modernized within 30% of the earlier plan ie delivered in 2 sprints as against the initial plan of 6 sprints. This mind set change had enabled the team to maneuver through rest of the release.

**Conclusion:**

Through this journey, attempting to connect two extreme ends, the focus areas that helped us bridge the gap between these extremes were very simple concepts and the challenge was in implementing which vary from project to project, enterprise to enterprise. The key challenge is to maintain the right balance to make sure the project is neither ‘too front loaded’ nor ‘handle one sprint at a time’.
Appendix

At the end of Sprint 5, the actual delivered is below planned. After every sprint, the delta is analyzed.

![Burn Up Chart](chart1.png)

The chart below shows in more detail on what is the required run rate that we need to scale up, to hit release date. In this case the scale up required was more than double (from 24 to 53).

![Run Rate Chart](chart2.png)