

Sprint burn down graph signatures

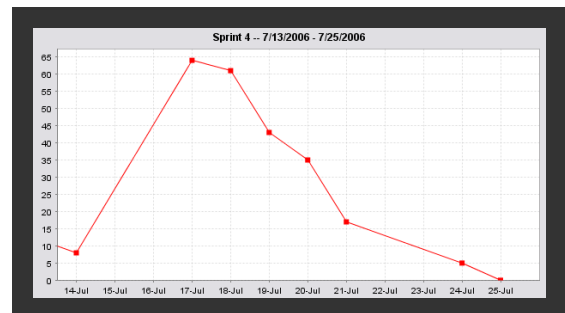
Hi,

Today I want to talk briefly about some common Sprint burn down Graph signatures. I've collected Sprint burn down graphs from different teams for awhile, and I've noticed several common patterns emerge. This article, which I wrote for my blog a few years ago, discusses what these patterns are and what causes them.

Burn down graphs are commonly used in Scrum projects to give the team an understanding of the amount of work remaining for the Sprint (or Iteration). In Ken's own words:

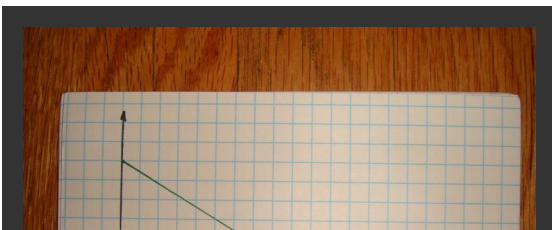
"As a team works together, it develops its own style of creating and maintaining the Sprint Backlog. It also demonstrates unique work patterns, some working consistently, some in bursts, some at the end of a Sprint. Some seek pressure, while others seek regularity. Across time, the backlog charts of each team develop predictable patterns. They stabilize as the team learns the technology, the business or product domain, and each other. These chart patterns are called Sprint signatures." - ControlChaos.com

Although burn down charts are very individual to the project team, I have noticed some similar signatures in many projects. The graphs, I've drawn are intended to show the general trend and are greatly stylized. Real graphs are more chaotic with datapoints both above and below the trendlines. Here's an example of what a real burn down graph often looks like:

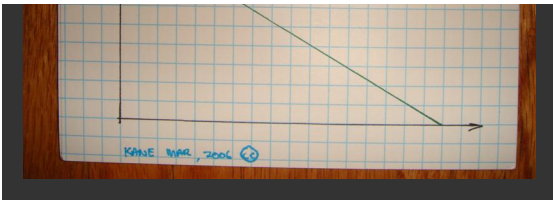


This article is a collection of 7 common burn down signatures that I hope you will recognize in your own projects.

The Fakey-Fakey

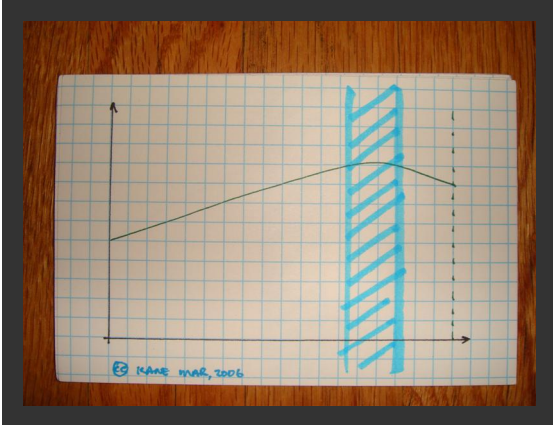


The Fakey-Fakey is characterised by a strong and regular descent towards completion. Most software is sufficiently complex that there needs to be some discovery along the way. Presenting regular and predictable progress in a complex and ever-changing field is, at best, disingenuous, and, at worst,



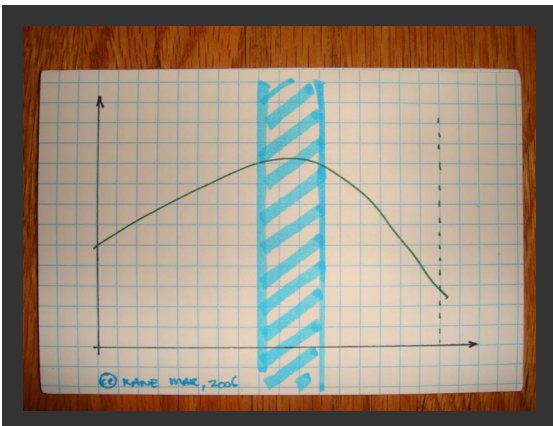
untruthful, presenting a false impression on the state of the project. The Fakey-Falsey is often presented by teams that are operating in a very "command and control" environment where they are uncomfortable being open and honest.

The Late-Learner



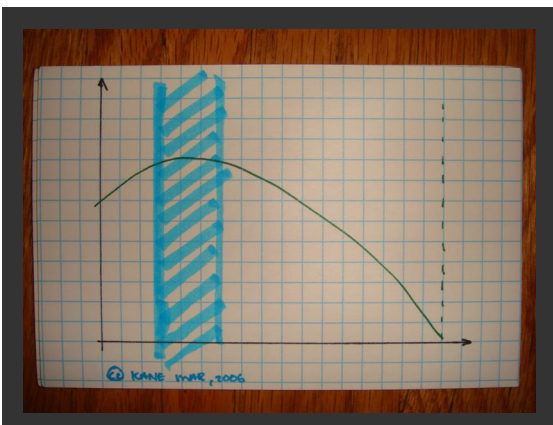
The Late-Learner has a learning hump (indicated by the blue shading) towards the very end of an Iteration. The Late-Learner is common for new teams that are still trying to communicate effectively and create software that is "Done" by the end of the Iteration. In newly formed teams, the late hump is often due to a very late realization that testing is an important part of delivering demonstrable software at the end of an Iteration.

The Middle-Learner



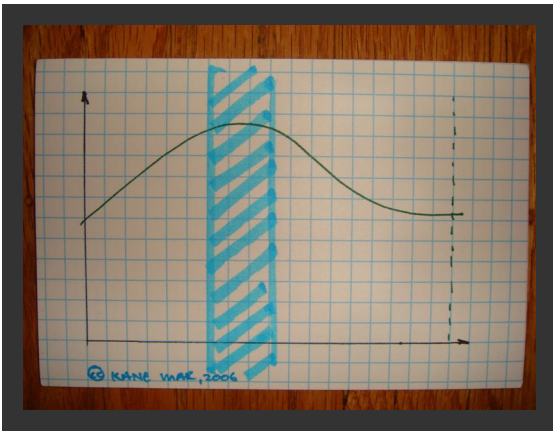
As the team starts to mature, emphasis is placed on early discovery, especially the early definition of what needs to be tested. This helps move the bulk of the work into the middle of the Sprint as shown below.

The Early-Learner



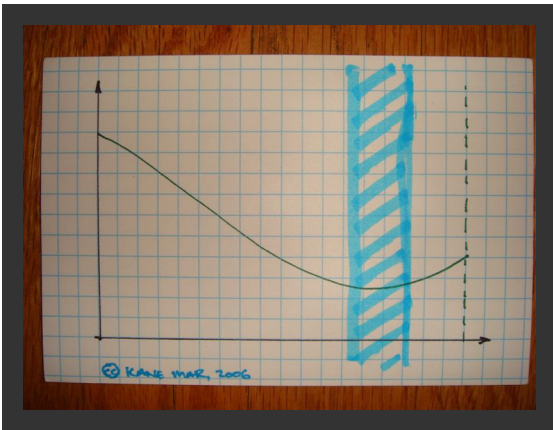
Scrum teams that are performing well will often have a Sprint burn down graph with an early hump and then a gradual burn down. In this case, the team has learnt the importance of early discovery usually by early definition of what needs to be tested. Once they have a more substantial definition of what needs to be accomplished in the Sprint, they work steadily to achieve it.

The Plateau



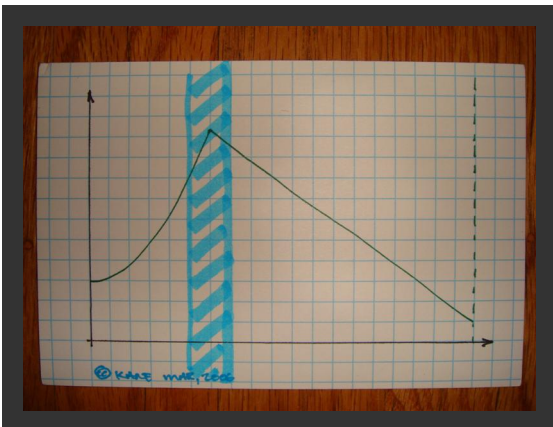
While teams are trying to find a balance between Early-Learner and Late-Learner, they will often go through a phase where they initially make good progress, but this progress is not carried through to the end of the Sprint. The burn down signature will look something like the Plateau.

The Never-Never



Sometimes a team that is working well together will have a surprise at the end of the Sprint. Perhaps the team sought clarification too late in the process, or perhaps the Product Owner wanted to change the scope of the Sprint. A sudden increase in the amount of work outstanding very late in the Sprint (whether due to late discovery or scope change) will make it difficult for a team to be able to meet its commitments. These late changes need to be raised and resolved as part of the retrospective.

The Scope Increase



This graph is characterised by a sudden increase in the estimated amount of work remaining. It's usually a sign that the team did not fully appreciate the scope of work committed to during the Sprint planning meeting. There are several approaches to dealing with very large scope increases.

My preferred approach is to negotiate with the Product Owner, but in situations where there is a complete disconnect in understanding the team may want to consider terminating the Sprint.

Looking ahead

Burn down graphs aren't used only to show the progression of a Sprint; they're also useful for tracking the rate at which work is being completed and the new work that is being added. Next week, I'll discuss the concept of release burn down graphs in more detail.