JOIN OVER 100,000 PROS! Join Log In | Member Benefits

STICKYMINDS

Agile Testing DevOps Security AI

Tracking what Matters with Burn Down Charts

ARTICLE

By Steve Berczuk | February 7, 2010

SUMMARY

Burn down charts help agile development teams track sprint and release progress. The basic idea of a burn down chart is that the team starts with estimates for all of the tasks in the sprint, and then on daily (or more frequent) basis re-estimates the amount of work remaining.

I'm a big fan of burn down charts for tracking sprint and release progress. The basic idea of a burn down chart is that the team starts with estimates for all of the tasks in the sprint, and then on daily (or more frequent) basis re-estimates the amount of work remaining.

With a burn down chart, you are tracking the new estimate based on the work that you have done. As you work on the sprint backlog you get a better understanding of the tasks, and thus you can revise estimates for tasks that span more than one day. This is reasonable since the original estimate is, well, an *estimate*.

Sometimes if you spend 4 hours on an 8 hour task, you'll have 4 hours of work left. Most of the time the time left will not be the original estimate less the time spent, but more or less. At the end of 4 hours, the remaining work estimate for the same 8 hour task could be 2 hours, or it could be 10 hours if you discovered a snag. This is important information for everyone involved in the project and allows the team to identify a problem at the daily scrum. Re-estimating is harder than just doing subtraction, but it's valuable.

One thing that happens when teams use an issue tracking tool (like Jira and Greenhopper) to manage their backlog is that re-estimating and effort tracking are combined. The only way to re-estimate is to "log work." You're required to enter the amount of time spent, and the tool will kindly offer to change your estimate to the difference between the original estimate and the time spent. There are two problems

We use cookies to ensure you get the best experience on our website.

Accept

Read More

new estimate" is always wrong.
- The "time spent" value isn't really useful to stakeholders. In the best case you are only doing one thing during the time in question, so your time spent entry is accurate, but doesn't answer the question: "when will it be done!" In the worst case, you're not tracking your time accurately, and the time spent number is inaccurate, and provide no real information.

Like all things agile, when looking at your project tracking approach you need to be clear about what you want to track and why. The main concern for stakeholders on an agile project is whether they will get the functionality they want at the end of the sprint. So the time-remaining number is important.

There are some good reasons for tracking time spent including evaluating estimation accuracy and billing
But in both of these cases you need to evaluate the overhead of the tracking time relative to the value. Tracking total effort for the sprint relative to estimated work done may be more useful than per-task effort to estimate tracking, and analyzing the results in a retrospective may yield more useful information than per-task tracking.

When doing sprint tracking:

- Make sure that everyone understands the goals of the tracking process so that you get uniformly valuable results. You definitely want to track how close you are to "done," but explain how important tracking effort is.
- Make sure that, whatever the goals, that the data are updated daily. If the burn down chart doesn't change for 2 days is it because people didn't update their estimates, or that the project is at a stand-still?
- Remind everyone that the estimates are just that: "estimates," and an informed guess that turns out to be wrong is better than no estimate at all. (And the inaccuracy of the estimate helps to identify unknown complexity.)

burn down charts can be a simple, valuable, tool to identify problems during a sprint as long are your teams breaks out of the habit of tracking "effort" as opposed to "effort remaining."

Topics: agile, scrum,

We use cookies to ensure you get the best experience on our website.

ABOUT THE AUTHOR

Steve Berczuk

Steve Berczuk is a Principal Software Engineer with experience as a manager, Scrum Master and technical lead in Boston, MA. The author of *Software Configuration Management Patterns: Effective Teamwork, Practical Integration*, he is a recognized expert in software configuration management and agile software development. Steve is passionate about helping teams work effectively to produce quality software. He has an M.S. in operations research from Stanford University and an S.B. in Electrical Engineering from MIT, and is a Certified ScrumMaster. Contact Steve at steve@berczuk.com or visit berczuk.com and follow his blog at blog.berczuk.com.

UPCOMING WEBINAR

LETS HANG!

Log In or Register to post comments

We use cookies to ensure you get the best experience on our website.

0 comments

Join the crew for FREE! Access exclusive content, contribute, and more.

See All Member Benefits

We use cookies to ensure you get the best experience on our website.