```
Errno::ENOENT: No such file or directory @ rb_sysopen - /Users/rpowell/Sites/ee.
```

Three Approaches to Estimating the Impact of Holidays and Time Off on Velocity

by Mike Cohn • 16 Comments

Image not found or type unknown

Three Approaches to Estimating the Impact of Holidays and Time Off on Velocity

Velocity can be a very useful predictor of how much work an agile team will complete in the future. It can be especially helpful when looking forward at least four or five iterations.

This is because over the short term, velocity can vary from iteration to iteration. But, velocity is more stable over the long-term as the law of large numbers kicks in and errors in individual product backlog items are balanced out.

This means velocity can work as a reliable predictor as it allows teams to state things like, “In the next 5 sprints we’ll probably finish between 100 and 120 story points.”

Such statements are based on a team knowing its velocity (perhaps as a single, averaged value but even better as a range).

But how should a team estimate its future velocity when team members are taking time off, whether personally or because of a company holiday?

There are a couple of different approaches to this. And each is appropriate in different contexts.

Approach 1: Ignore It

When adjusting velocity for holidays, vacation or personal time, a simple but valid strategy may apply: Ignore it.

Ignoring the impact of various forms of time off can be a valid approach when planning over a sufficiently long enough period. The idea is that *team members will probably take time off in the future at the same rate they took time off in the past*

.

If that seems true for the period for which you are planning, you don’t need to make any adjustments to velocity at all.

An Example

As a trivial example, suppose you are planning a project for all of next year. (I don’t recommend you commit to a fixed date and scope that far out, but it makes a good example.)

In this case, a good starting point will be total velocity for all of this year. Any times of reduced productivity (such as Christmas or summer vacation time) will balance out over a long term.

Approach 2: Adjust Proportionately to the Full Team

In some cases, however, you do want to adjust for holidays and team member vacations. You should do this if you’re in a situation such as:

- the project is shorter (perhaps 3-6 months) and you don’t think time off during that period will be consistent with the past
- You know that one or more team members will be out for a longer than normal period such as a sabbatical, birth of a new child, or just an extended vacation.
- One or more holidays with greater than normal impact occur during the period

The Simplest Approach

The simplest approach is to assume that that everyone on a team can do everything. This is probably not true but it can be a starting point in deciding how to adjust velocity.

When making this assumption, the formula for predicted velocity in a sprint is:

Predicted Velocity = Average Velocity × (Planned Working Days ÷ Average Working Days)

An Example

An Improvement

In this example, I have used 60 days as the team's average number of working days. For a six-person team running two-week iterations, 60 is very unlikely as it assumes no one is ever sick.

As an example, suppose a six-person team has an average velocity of 40. They are running two-week iterations. In the coming iteration, one team member will be out the full two weeks, and the rest of the team will be gone for one day for a national holiday.

This team normally has 60 working days in an iteration (6 people × 10 days). In the coming iteration they will lose 15 days to time off. (One person for ten days; the other five for a day each.) This leaves them 45 working days in the coming iteration.

This means their predicted velocity will be 30. This is determined by taking their historical average velocity (40) and multiplying it by the percentage of days planned to be worked (45÷60=0.75). So

$40 \times (45 \div 60) = 30$

You should begin tracking the actual number of working days per iteration and use that value rather than the theoretical maximum number of days.

Approach 3: Adjust Proportionately Based on Skill

When team members are not highly interchangeable, a better approach can be to reduce velocity by the same percentage that a person's absence represents within their primary skill.

For example, if one of two database administrators on a team will be gone for the entire sprint and no one else can do that work, reduce velocity by 50% since total DBA capacity is down that much.

For most roles this will be too dramatic of a reduction. The best approach I've found is to use somewhere between this amount and the amount given by assuming everyone is equal.

Combining Adjustments

Let's return to the case of one of two DBAs being gone for an entire iteration. And let's assume the team is made up of six individuals running two-week (ten-day) iterations.

In this case, one DBA being gone represents a 50% reduction if the amount of work the DBAs will finish (1÷2). It also represents 17% less total capacity in the sprint (1÷6).

So, one DBA being out will likely reduce velocity between 17% and 50%. Choose a specific amount for your project based on whether DBA work is on or near the critical path and how interchangeable others are at helping with the DBA's work.

What to Do When Multiple People Will Be Absent

If multiple people representing different skills will be absent during a coming sprint, reduce velocity only by the absence with the largest impact.

Think about it with this extreme example: Suppose 90% of your programming capacity will be gone if the coming sprint. And suppose 10% of your testing capacity will also be gone. The fact that you have slightly lower testing capacity won't matter in the sprint. The total work done in this case will be determined by the much larger reduction in programming capacity.

An Example

As an example, let's assume we have an 8-person team with two DBAs and three full-stack developers. Of these, one DBA and one developer will be absent for the entire sprint.

In this case, the DBAs will lose 50% of their normal capacity (1÷2) and the developers will lose 33% of theirs (1÷3). Treat those as upper limits on likely loss and then reduce velocity only by the larger of the two.

Will You Help Me by Sharing Your Biggest Estimating Challenges?

Reliably planning an agile project is completely feasible. And for many teams it’s necessary. Perhaps someday those teams can operate in a “you’ll get it when you get it” environment.

But today many teams need to create and communicate plans with the expectation that the plans are sufficiently accurate for good decisions to be made.

Right now, I’m working on something that deals with the problems of creating estimates particularly for fixed-date, fixed-scope, fixed-price (or even fixed-everything!) projects.

Have you ever struggled with estimating in these situations?

If so, I’d love to hear from you.

Would you click the link below and answer some quick questions?

Tell me your fixed-price challenges.

Posted: November 27, 2018

Tagged: estimating, velocity, planning