"Story points work great at a team level, but when we use them at an Epic level they don't really work that well. How can we make them work at this level?"

November 2016

Scope of this Report

Within the agile world story points are considered the metric to go to if we talk about teams that want to estimate the relative effort for their user stories. However, within organizations that use multiple agile teams, when we extrapolate those story points to an Epic level, the aggregate metric starts to show some flaws that need to be acknowledged.

This report analyzes the use of story points at the Epic level and proposes some alternative sizing solutions.

Story points and estimation by experience

Story points are usually defined as a metric used in agile project management to estimate the effort that a certain item of our backlog will require. Usually, they are used to measure user stories in a Fibonacci-like format (0, 0.5, 1, 2, 3, 5, 8, 13, 20, 40, etc.). The aim is to start by assigning a value to one of our user stories (using techniques such as planning poker[i]); and after that the team should be able to estimate the rest of the user stories. Therefore, it is a relative measure that tries to estimate the effort required to implement a user story compared to other user stories being considered by that team in that session. To fairly estimate effort, we should consider not only the amount of work to do, but also things like the complexity or the risk of doing that work.

Some people may wonder why is it useful to use story points instead of estimating directly in hours and the main factor has to do with how our mind works, meaning:

1) It's difficult to estimate time correctly, more so with the uncertainties and pressure that exist in software development environment.

2) It's easier for us to compare user stories once we have defined a reference than to estimate the time for each user story separately.

If we want to use this effort estimate to help us do the planning within our team, first we need to obtain our velocity. The velocity is defined by calculating the story points per sprint that the team completed during a reasonably sized sample of iterations or sprints. Once we have enough historical data, the team can estimate the velocity (Fig. 1) during the remaining sprints or iterations (assuming that the velocity will remain the same).

270 West Lancaster Ave., Liberty Square B-2, Malvern, PA 19355 • 1-610-644-2856
info@softwarevalue.com • www.softwarevalue.com

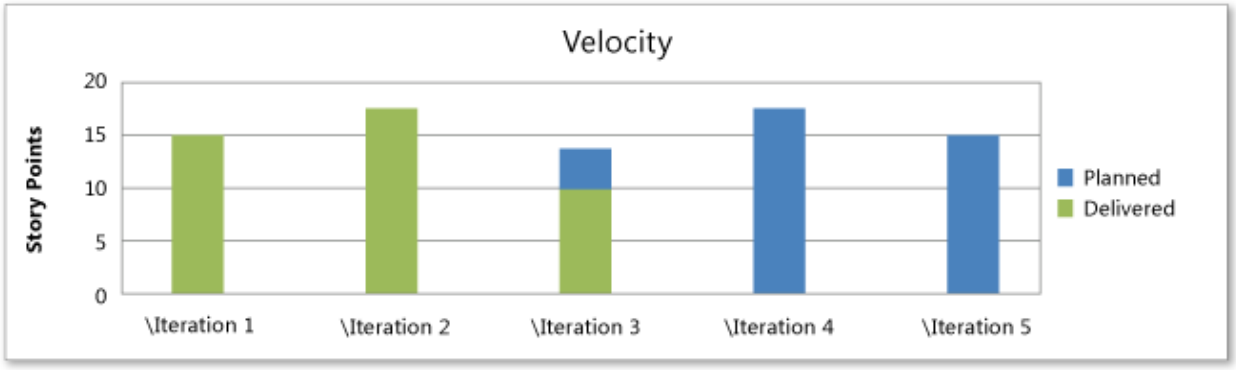

Figure 1: Team Velocity (Source: Product Planning Workbook, Microsoft)

This will help us plan and limit the work for the following iterations, but it has a few drawbacks that need to be considered:

- It takes time to estimate with story points
- Velocity fluctuation is usually difficult to explain
- It's not possible to compare the velocity between teams
- If you want to plan a project based on story points, all the user stories need to be defined and measured beforehand either by the same team or against a standard reference story (or two).

Story points at an Epic level

Epic is the term used to define a large user story. It usually provides some business or architectural value and generally takes more than one sprint to be implemented. Having that in mind, Epics are usually too large to estimate directly by using story points. To do so successfully, we would need to break down the Epic into the smaller user stories and we would need to have the same team working on all those user stories.

However, the reality nowadays is that Epics are developed by multiple development teams at the same time, working across different systems, platforms and technologies. In this environment, using story points to plan or to see the progress status of an Epic using graphics such as the Epic progress measure (Fig. 2) proposed by the Scaled Agile Framework (SAFe)[ii] becomes more inaccurate since different teams will estimate in different ways, which will lead to false accuracy and wasting time in estimation.

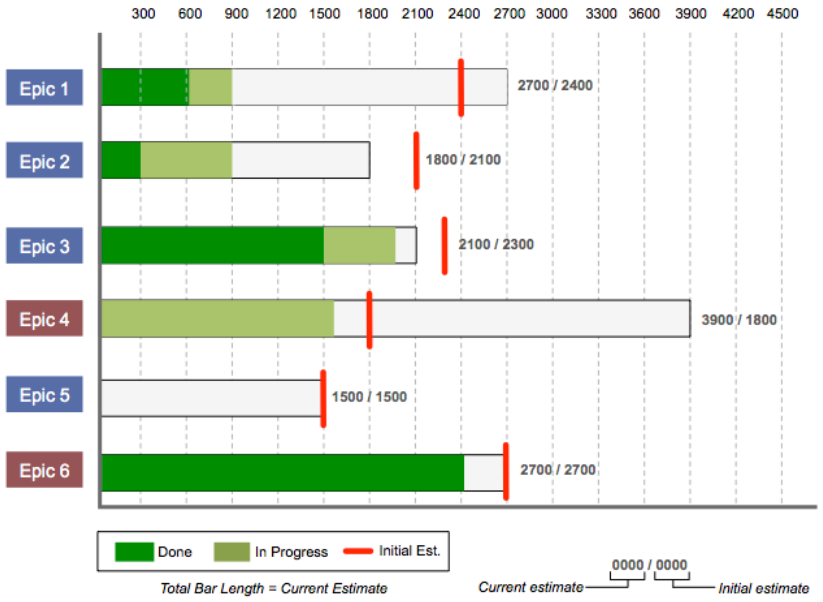

Figure 2: Epic Progress Measure (Source: Scaled Agile Inc.)

So, is there anything we can do to make story points work at an Epic level within this kind of environment?

Well, there is no easy answer for that question, but we will propose two different alternatives to solve this problem.

The first one would be to keep using story points. To do that we would need to change the way in which we usually estimate those story points. The idea is to find a common ground where all teams can refer when measuring their user stories. First we need to find a set of user stories that cover most or all of the teams that are involved in the development. Next we need to agree to size that set of stories so that it serves as a baseline for every team. Lastly we need to adjust the user stories developed within each of the teams so that they comply with the user stories used as a reference. This should also be a live process, meaning that it needs to be readjusted as the process goes on. Additionally, we would need to re-estimate the user stories after every sprint. This provides a series of benefits:

- The teams get more experience and get better at estimating, making metrics like capacity and velocity more stable values
- The planning for the following sprints become more accurate
- The feedback from previous sprints helps us adjust the process

The benefit of this approach is that at least the degree of inaccuracy will decrease and, to a certain point, we will be able to use this metric.

Alternate measure – estimating functional size

The second option would be to use an alternate metric to measure the Epics. Ideally, a successful estimation process should be repeatable and consistent. It should also have a common way to quantify what we are measuring. Moreover, the metric used should be easy to calculate, objective and provide some added value to the business.

That's why the second recommendation is a metric that can capture the size of the functionality that we are developing. There are a number of potential size measures but the industry standard approach is to use IFPUG Function Points[iii], since this metric successfully fulfils all those characteristics. FPs are agnostic of any technology or methodology and can be counted from any type of documentation or having interviews with the SMEs. We could also use them to count at a user story level (see "Can function points be counted/estimated from user stories?" by Lori Limbacher).
So, what are the main benefits of using function points as a metric within an agile environment?

Function points are:

- An accurate metric to measure the functionality that is being delivered
- More useful than story points for early estimation, since less data is needed to provide that estimation
- Used to measure all the backlog and then capture functionality that is being delivered
- Compatible with the use of other methodologies
- A tool for the developer to manage the project outcomes
- Used to help the end user to understand the functionality that is being delivered
- Used to identify what features and functions are being developed and deployed
- Used at an early stage to help with the planning and after the functionality is being delivered to measure things like productivity.
- A consistent process and easy to validate
- Used for benchmarking outside of the project / company.

Actually, why not use both?

It may seem like given the two choices above, we are supposed to choose between story points or function points, but that's far from the truth. Both metrics are useful and have different intended purposes within the organization.

Story points work great at a team level and FPs can work at that level and at higher levels as well. But it's really up to every organization to decide which of those metrics fits better within their environment depending on their objectives.

The use of both metrics would allow us to do things like benchmarking using FPs and at the same time have a specific metric that every team can rely on to see their progress using story points, and that's just one example of what we could do.

Conclusion:

Is it possible to use story points at an Epic level? It is, we just need to be aware of the uncertainties that it creates and work towards solving them. Usually, we would be better off if we used a complementary metric that helps us tackle those uncertainties. We suggested the use of FPs since the information that they provide can be used to both estimate at an early stage and to measure performance upon delivery. But in the end, it's really up to every organization to analyze and decide which metric is best to help them with their software development process.

Sources:

[i] https://en.wikipedia.org/wiki/Planning_poker

[ii] http://scaledagileframework.com/

[iii] http://www.ifpug.org/about-ifpug/about-function-point-analysis/