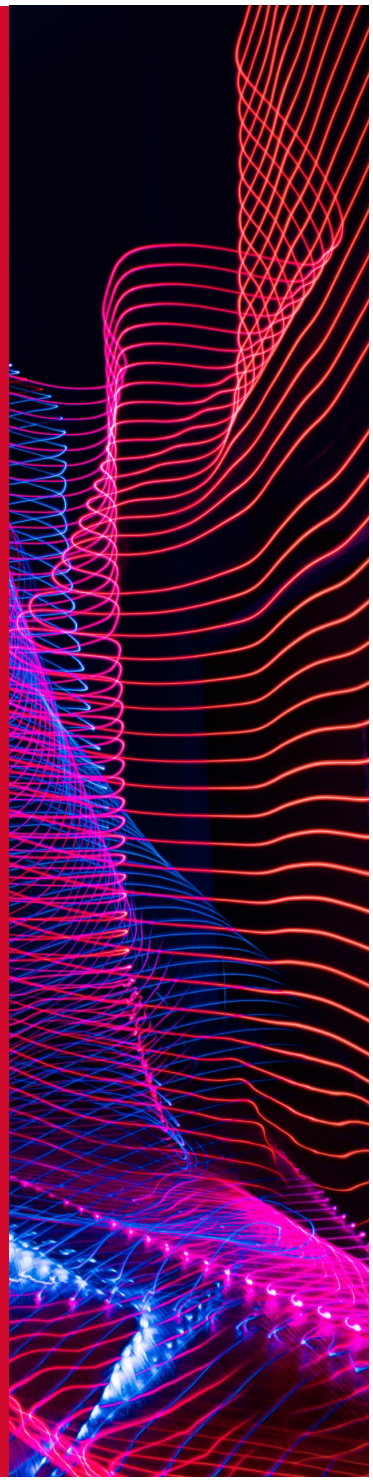
responsiv
simple · effective · distinctive

POINT OF VIEW

How to Improve Project Estimates By Underestimating Agile



Introduction

Improve Agile Project Estimates with good technique and solid evidence.

> **This point of view takes a look at some techniques and tips for approaching estimation and trying to avoid some of the pitfalls involved with maintaining time tracking that can impact Agile project planning.**

How often does estimation and project estimates come up in post project reviews as an improvement area? It is not exclusive to Agile projects but this is a common area where project teams get into difficulty.

Planning Poker

This is a technique for a project team to produce estimates that is used by Responsiv Consulting. The preparation for this involves production of a project backlog, which is a list of project deliverables broken down into manageable pieces of work. Manageable, here, refers to defining things at a scale that makes sense to the project team.

This can be in terms of the scale of the work involved as well as having a clear and measurable objective. The scale of work here might be agreed as something that should be achievable in a (man) week. With Agile project sprints regular run at two-week (or shorter) durations it makes little sense to have blocks that are larger. If in doubt break it down further and this should help overall precision and benefit the management of the project.

The planning poker technique involves team members picking size estimates for the backlog items. The size choices will be based on a Fibonacci sequence of sizes – for example 1, 2, 3, 5, 8 or 13 ideal development days or hours. You may wish to adjust this set of numbers closer to a decimal set of figures such as 0.5, 1, 2, 3, 5 and 10.

The main point is to have a basis for relative sizing and not to get hung up on the precision when going into the larger numbers. Another key point is not to work at too coarse a level. If items are being sized at more than ten ideal days effort, then break them down. This could be into stages of work (e.g. separate design, build and testing activities) or dividing up the functionality (e.g. front-end, server side).

As well as being agreed on what the size choices represent (such as ideal man hours or man days), it is also very important to be clear on the scope of the items being estimated. This connects with having a common understanding of what "done" means for backlog items. For example, does a backlog item to "develop the customer address interface" include design and unit testing? An agreed convention for the department or project will help ensure that team members are comparing like with like.

In order to make the most of the breadth of experience in a project team it is recommended to have members of the team pick their estimates independently of each other. This can still be done on a group planning call if care is taken to discuss the backlog item without specifically discussing the estimate until the team have made their selections. Such discussion would run the risk of anchoring – where team members could be influenced by a view of the estimation from another team member.

When the team have all made their selections the facilitator (e.g. scrum master) can go around the team one by one to discuss any differences between the choices. The result would be a consensus estimate size for the backlog item to be recorded against it. With a tool such as Jira this would be recorded as both the Original and the Remaining time estimates.

What about Contingency in Project Estimates

What about catering for unknowns and possible risks that could delay progress. The idea of the planning poker technique outlined above is to cater for contingency by having the team produce a realistic consensus estimate. Discussion of the backlog item could move the team to a higher estimate figure if a member of the team feels that allowance needs to be made for potential risks. A higher figure may be opted for where the team is working on a less familiar technical area.

Where a collective planning meeting or call is not going to be possible then another approach can be to have team members provide worst case and best-case estimates alongside their estimate for each backlog item. These three estimates (per backlog item) can be combined using a weighted average formula (for example take (*best* + 3 * *normal* + *worst*) and divide by 5).

This can allow the project leader/ manager to review the inputs from the team and get a view of where it would be prudent to use a higher estimate in the planning. In case of probable variations, it would still be recommended to review the items with large variations in a focussed call or meeting. This would form part of sprint planning.

Sprint Planning

The estimates assigned to backlog items are essential to the process of planning sprints, i.e. selecting a set of backlog items that should be achievable for the team to deliver within the sprint. Note that that is not just a case of adding up the estimates and comparing with the available team capacity for the sprint period. Some other factors will need to be considered.

Significantly there will be an adjustment factor to be applied to consider the difference between ideal time and actual. This can also be adjusted to reflect team experience with the technologies and tools being used for the project build. This could be set as a percentage figure to be applied (so less than 100%) - with a smaller figure where the team have inexperienced members who will be calling on other team members for support as they go.

Tracking and Maintaining Project Estimates

Having an estimated backlog could be seen as the end of estimation work until the team emerge from the sprint and review the outcome. However, that would be missing the chance for ongoing progress to be monitored by the project team, the project manager and by extension any other interested stakeholders (e.g. product owner).

Whilst it is good for the project team to be free from major planning sessions and discussions through the course of the actual sprint. It is nevertheless important that they do, as a minimum, maintain track of the progress made each day. In a tracking tool such as Jira this is done by having each team member complete the *Log Work* dialogue for each backlog item that they have worked on. This is just for the items progressed on the day and does not need to extend to all their items. Completing this type of dialogue (Figure 1) need not take long and so is not a lot to ask of developers and technicians.

Product Overview IBM App Connect Enterprise 1

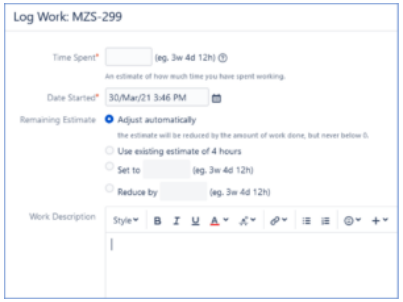


Figure 1; Time Tracking in Jira

The minimum mandatory entry is to put how much time has been spent on the item for the current (or previous) day. This is put into the *Time Spent* field.

The next main field to consider is the *remaining estimate*. If things are tracking well and in-line with your prior estimation, then the system will automatically adjust the previous *remaining estimate*. This is the default choice. The *time spent* amount will then be subtracted from the *remaining estimate*. If you feel that this is not going to maintain a realistic view of the *remaining estimate*, then you will need to use one of the other choices.

Also, the earlier you make this shift the better it will be for keeping a realistic view of progress within the sprint – something both the project manager and the team will appreciate. The figures will feed into the burndown chart for the sprint where overall progress will be apparent.

The alternative choices below *Adjust automatically* depend on your view of the remaining estimate. You can set the figure directly with the *Set to* choose. If you want to retain the previous remaining estimate, then choose Use existing estimate. Finally, you could choose a Reduce by value to subtract from the existing estimate. For example, if the existing estimate is 8 hours and having worked 4 hours today you believe there is still 6 hours remaining, you could enter 2 hours into the reduce by field (as *2h*).

It is important to remember that there are two quite distinct inputs here. Firstly, the *time spent* may appear to be more of interest for an accounting view of the time spent. However, it is also very important for improving future estimation accuracy. The level of assistance can be undermined if there is any reluctance to be accurate here – as can be the case if team members feel that there is any potential downside to increasing an estimate through the course of a sprint.

If there is a culture like that around estimation, then it can easily lead to team members under calling actual time spent. From the overall project view it can then appear that the team are falling short on meeting the planned capacity and lead to questions about unaccounted for time. So, this is the measure that will be most useful for improving future estimation by the team – based on a true comparison of time spent versus the original estimate.

Secondly the *remaining estimate* figure is vital for tracking the sprint progress against burndown and also informs the view of when the item or task is likely to be completed. This is clearly a very important picture for those interested in project progress.

A tip to encourage the regular maintenance of this information is to have a daily reminder in the calendars of the project team – either beginning or end of day. If the team are having daily stand-up meetings, then it can be useful to have the reminder set in advance of that. Without this it is quite likely that the project manager will start

to sound like a broken record, as they will inevitably have to regularly remind team members to fill in their time updates.

Conclusion

This point of view has covered some estimation techniques used on Agile projects. It gives reasons for having a consistent focus on keeping that time information up to date throughout the course of a project. From sprint to sprint to inform tracking in addition to the longer-term benefit of improving estimation accuracy. Practiced consistently it should reduce the likelihood of estimation featuring on the improvement point list from your sprint retrospectives and post project reviews.