Sizing Agile Projects Consistently

What Is a Story Point?

This often-asked question is really a trick—it doesn't have an answer, and more importantly, that doesn't matter! To see why, let's look at a more common unit of measure: the inch. While you probably don't know what it is, it would not surprise you that "inch" has a formal definition, an international standard. The international standard for an inch is exactly 0.0254 meters, which of course raises the question, what is the international standard definition of a meter? A meter is the length of the path travelled by light in a vacuum in 1/299792458 seconds. Now you know!

But when you use inches for practical purposes, the definition of an inch doesn't usually matter. If I need to cut 10 pieces of wood that are each 5 inches long together with 4 pieces of wood that are each 11 inches long, an 8-foot-long board (96 total inches) is just about right, especially if my saw is sharp enough to make clean cuts.

Likewise, story points can be used to measure the total size of items on an agile backlog. Like the example of cutting wood, we do not need to answer the question, "what is a story point" to do that. A story assigned 5 story points is a smaller than one assigned 11 story points. If I have a 96 Story Point epic, when I break it down to individual stories, I may end up with 10 small stories that are 5 Story Points each and 4 larger stories that are 11 story points each.

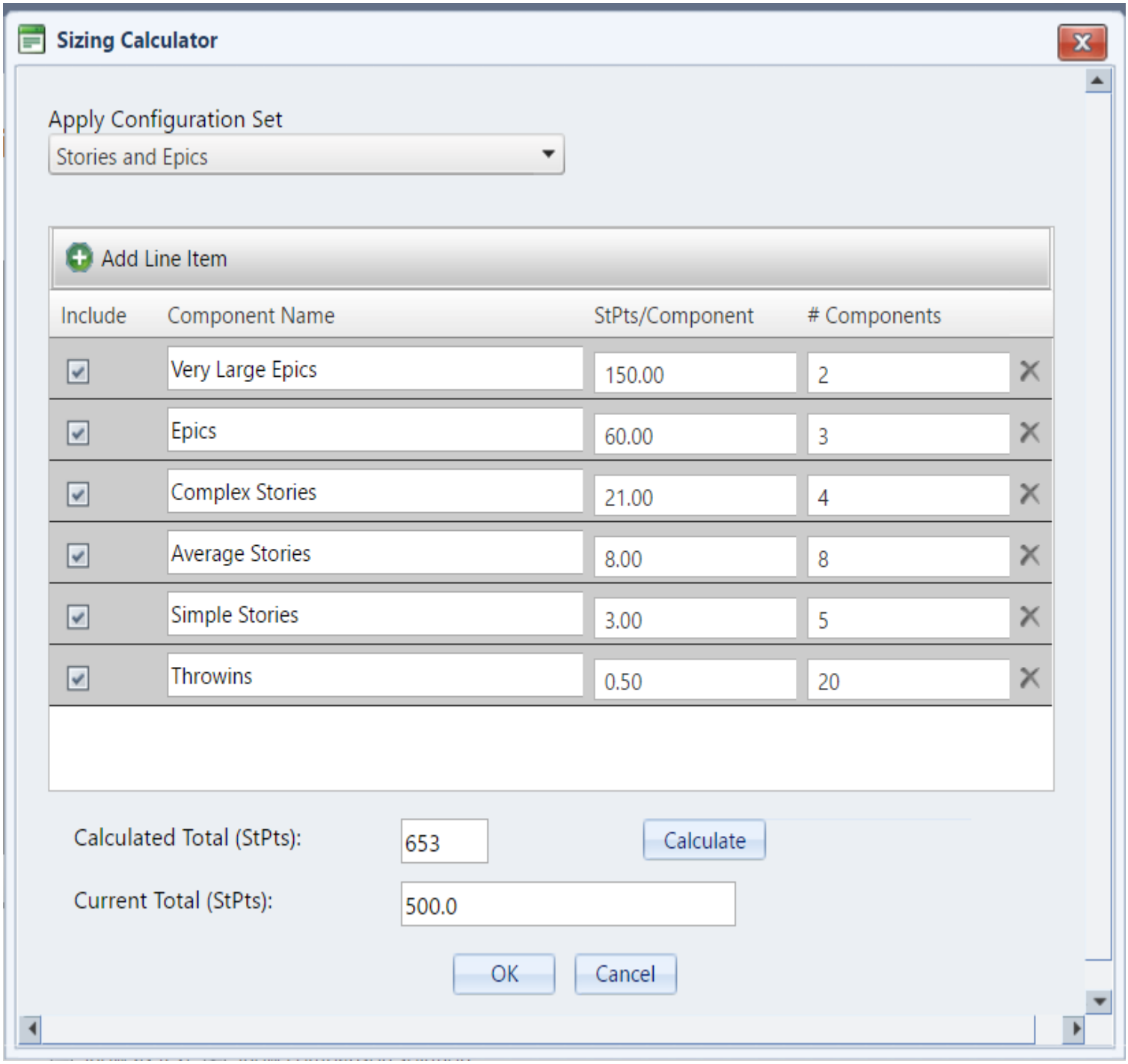
However, unlike measuring length in inches, there is no international standard to tell us how many story points to assign to any particular item. One team may assign a particular story 5 story points, and thus another comparable story also 5 story points, while assigning a third, larger story 11 story points. Another team, though, may assign exactly the same stories as 3, 3, and 7 story points. Both teams rate the first two stories the same **as each other**, while the third story is larger. The two teams agree on the relative size, but not how it's quantitatively measured. Many authors have explained why, when using story points for iteration planning on a particular project, this idea of relative measurement is enough. The only consistency that's needed for that purpose is within the team and the project. The members of the Agile Round Table found that some agile teams resented being asked to be consistent with other teams, wrongly (in our opinion) saying this goes against the agile principle that teams are self-governing.

Since agile teams are already using story points, the Agile Round Table looked at how that can be extended to get the consistency across projects needed for release level estimation, without violating the self-governing nature of agile teams.

Choose a Set of Size Bins

One technique that promotes consistency is already in widespread use by agile teams. Many agile coaches recommend using a fixed set of size bins. Some use 1,2,4, 8, 16, 32 and so on. Some use a scale from 1 to 10. The Fibonnaci series, 2, 3, 5, 8, 13, 21 etc., is quite popular. This way, we don't waste time discussing whether a story is 5 or 5 1/2 story points; we can put slightly different sized stories in the same bin. The members of the Agile Round Table confirmed that most of their teams were doing this.

The SLIM-Estimate and SLIM-Collaborate template for Agile Story Point Estimation has an example of this:



In addition to bins for the detailed stories that get developed in each release, also settle on a set of bins for the "big rocks," as described in the previous article in this series, Big Rock Estimation, in the section "Size Grounding." Since release level estimation is often needed before many detailed stories have been defined, the items in the backlog you need to size may mostly fall into these larger bins.

Make sure the teams are clear that you are not asking for consistency just for the sake of consistency! Adopting a consistent set of size bins extends the work they are already doing for an additional needed purpose.

Once the advantages are explained, settling on a common scale and set of size bins throughout your organization promotes consistency without violating the principle of self-governing teams, and will likely be accepted by your organization.

Pick “Exemplar” Stories or Epics for Each Bin

For iteration planning, the goal of putting stories into the chosen size bins is that all stories from the project that are in the same bin are roughly the same size.

For release estimation, it's also important that stories in each size bin be roughly the same size regardless of which team or project did the estimating. This kind of consistency can be difficult to achieve, since normally the backlogs on different projects are not compared to one another, nor estimated by the same people at the same time.

This consistency can be facilitated by picking example stories or epics to associate with each bin that will be familiar to the variety of teams that are doing the estimating. These “exemplars” will be used for estimating across projects and teams. When estimating the backlog of a particular project, instead of just comparing stories in the backlog to each other, you compare them to the exemplars to choose the appropriate bin. Your organization may be in a specific industry, like some companies represented in the Agile Round Table. In that case, the projects in your organization likely have a common domain, for example insurance, travel, or automotive engineering. For each bin, you can find an example story or epic from your domain that will be understood by all teams to use as the exemplar for that bin. In other cases, like some other members of the Agile Round Table, you may be a systems integrator or other type of organization that is involved with projects from many different industries. In that case, you can still pick exemplars for each bin from areas that are commonly understood by many teams. While these are not exactly like the items in any particular backlog you are estimating, the teams can still use them for comparison.

Note that promoting consistency *within* a single organization is not the same as setting an international standard to promote consistency *across* organizations. We do not think it is likely there will be an effort to set such a standard for story points, though that could change as agile sizing methods get more popular.

Dealing with Emerging Requirements: Calibrate the Big Rock Bins with “Estimated Actuals”

Several Round Table participants said their organizations were consistently underestimating agile projects. A chief cause was the lack of detail on the requirements at the beginning of the project—the lack of a “big upfront requirements phase.” Although epics were identified and sized, the size was underestimated because of the details that emerged only as those epics were refined from iteration to iteration.

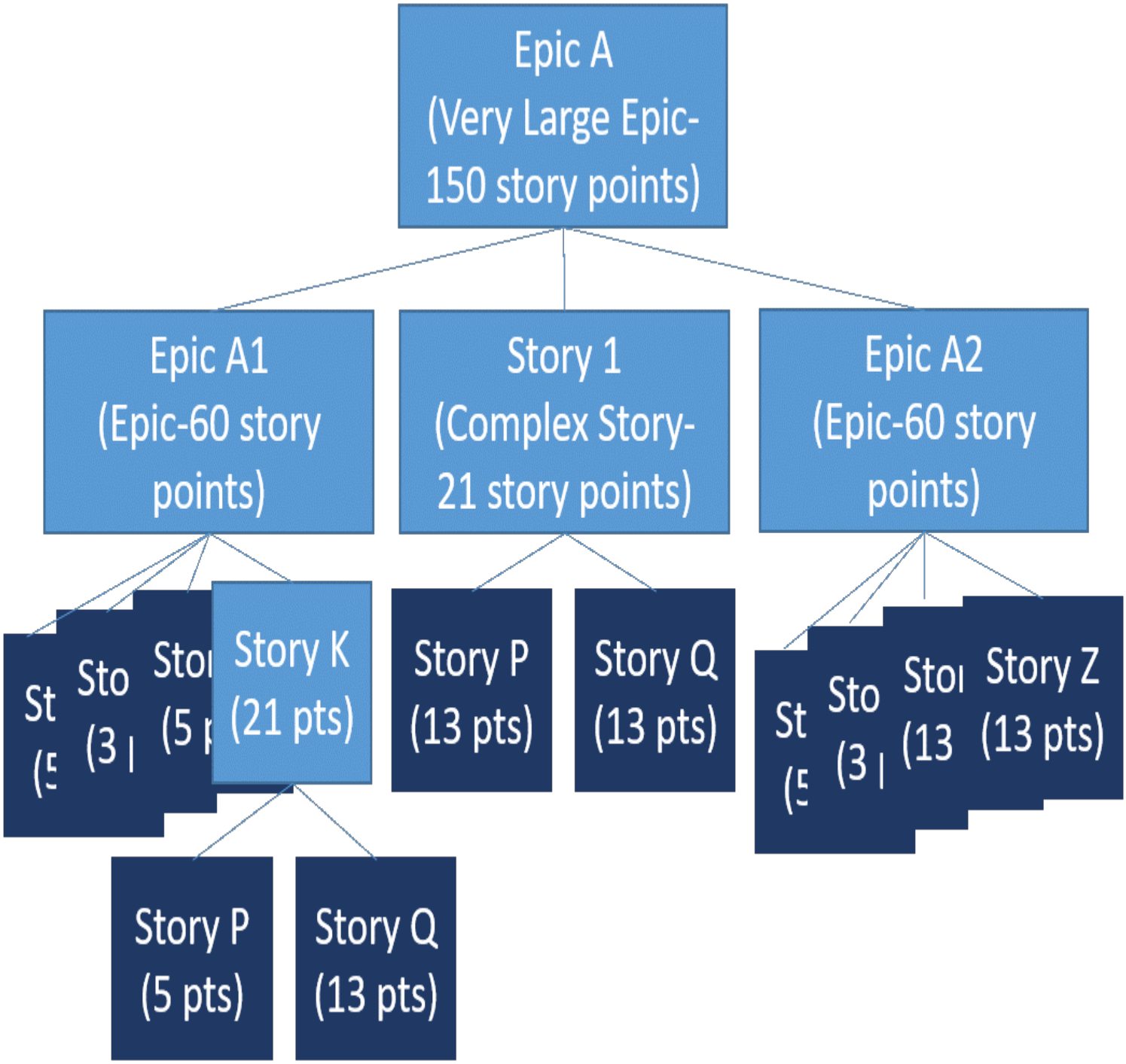
Emergent requirements are a hallmark of agile methods, and using this technique is probably the most important factor in, well, agility. This strength of agile methods means that to estimate in advance, you must be able to size accurately enough with only very high level requirements. This is the basis of the technique described in the earlier article in this series, Big Rock Estimation. The bins chosen for the big rocks must be assigned story points in order to get a total size.

To promote consistency, the bins and exemplars should remain stable over time. Course corrections can be achieved by adjusting the number of story points assigned to each bin. Here's how this works:

An agile project may start off with a backlog consisting mostly of big rocks. As the project progresses, those big rocks are broken down smaller, more detailed stories of various sizes. This forms a hierarchy of sorts, from the big rocks down to the developer sized bites which actually get developed. Many agile planning tools support tracking this hierarchy, or you can track it manually. Typically, then, agile teams will assign story points to each of those developer sized bites to plan the iterations in which they are developed.

There are, of course, other changes besides breaking large epics into small stories. Some stories are added to the backlog that were not broken out from the epics originally on the backlog, and other priority and scope changes mean that the hierarchy isn't perfect. But if you keep this information about a variety of projects, the individual changes will "average out" and that's good enough for the calibration.

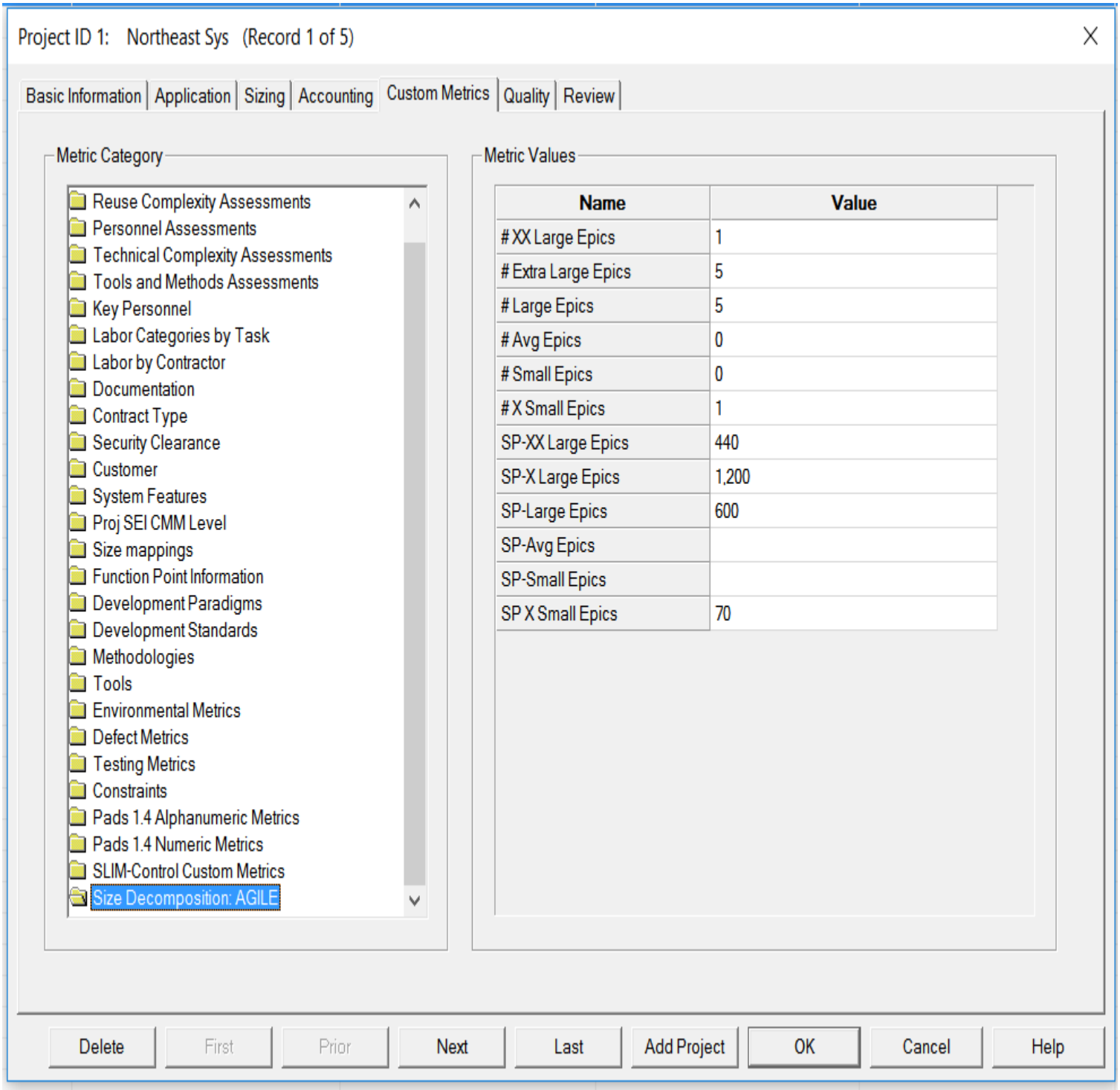
For each epic in the original scope, find all the developer sized bites that were derived from that epic. That is, find the small stories that actually made it into development that were broken out from that epic. Each of these actual stories was assigned a number of story points during iteration planning. Keeping in mind the typical rule that you don't re-estimate after a story is developed, we'll call these "estimated actuals." Total up these story points and track that total with the original epic or big rock, along with which bin to which that big rock was assigned.



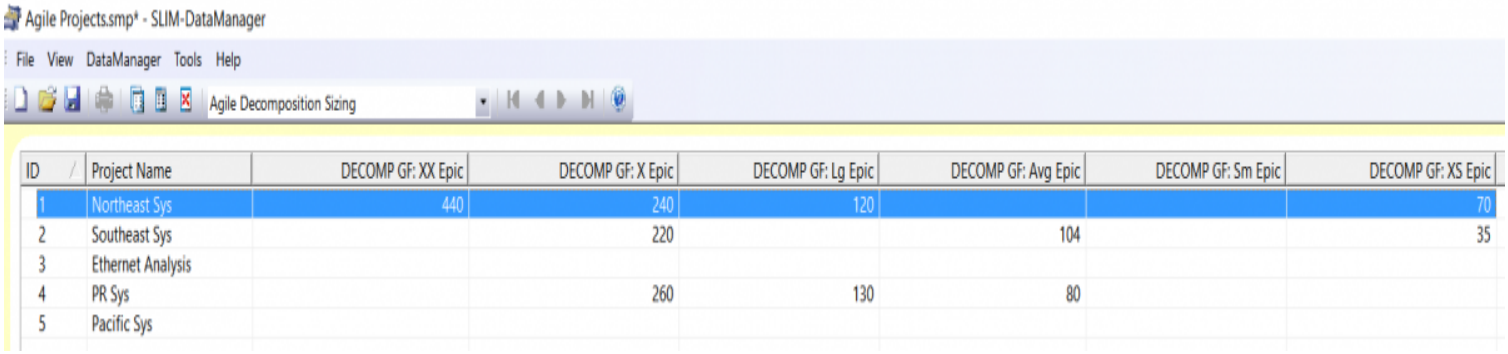
Add up the story points assigned to the detailed stories actually developed (illustrated in the darker color) and track that as the "estimated actuals" original "big rock."

Over time, you will collect data from multiple projects about a variety of epics within a single bin. While the bins you use stay the same over time, you can recalibrate the number of story points you assign to an epic in the bin for new estimates, perhaps, by taking the median or average of the estimated actuals for the completed epics already in the bin.

SLIM-DataManager v10 comes with a template to help you include the custom metrics and a data entry form needed for this computation. For future projects, use the same bins and exemplars, and use the calibrated size of the bins to get consistent enough sizing based on your history.



In SLIM-DataManager, you can enter the count and estimated actuals for each size bin in your project. SLIM?DataManager will then calculate the story points per epic for each bin to help you calibrate the bins for estimating future projects.



ID	Project Name	DECOMP GF: XX Epic	DECOMP GF: X Epic	DECOMP GF: Lg Epic	DECOMP GF: Avg Epic	DECOMP GF: Sm Epic	DECOMP GF: XS Epic
1	Northeast Sys	440	240	120			70
2	Southeast Sys		220		104		35
3	Ethernet Analysis						
4	PR Sys		260	130	80		
5	Pacific Sys						

Summary

We've described some techniques you can use to measure size of agile projects consistently enough to use for release estimation. You may need to overcome some objections the team may raise regarding some of these techniques. To do this, you must be clear about the purpose and value of what you are asking the team to do.

Using story points to measure size for release estimation fits with what the team already does for iteration planning, but you have to get consistency across your organization. Some key techniques to gain this consistency are "Big Rock Sizing," getting agreement on a set of size bins to use, and using "exemplar" stories to compare to stories and epics in your projects. However, there's likely not as much consistency from organization to organization, which presents challenges for using industry wide data.

We presented a method to calibrate the size used for the larger bins by tracing the big rocks used for initial sizing to the actual stories derived from them, thus getting historical data about the total size in story points actually developed for epics in a particular bin. This calibration can improve the accuracy of your estimates as your organization gathers data and gains experience.

In the next article in this series, we will look at more traditional units of measure for software, function points and source lines of code (SLOC), and their use for agile projects.