

4.4: Placeholder Stories

One of the most common issues for scrum teams is what to do about work that we expect to have to do during a Sprint, but don't actually know the details about yet, such as bugs we have to fix in existing systems, or expected sales support efforts. In this chapter we introduce a method to manage these "known unknowns".

Table of Contents

<i>General Budget / Planning Problem</i>	1
<i>Placeholder Stories to Manage Budget</i>	2
<i>Must have Robust BackBurner</i>	3
<i>Discussion</i>	4

When I'm coaching or training there is one question that I invariably get, which is "what do I do about the bugs I have to fix in other systems I'm maintaining? Every Sprint I get new bugs I hadn't planned for, and it throws my planning off. I get the same impediments Sprint after Sprint! Is there something I can do about it?"

Well, I think that it's crazy that something that happens every Sprint throws our planning off – why are we surprised? When we wind up having the same impediments over and over again, management is going to stop paying attention to us – we're like the little boy that cried "Wolf!"

What we are supposed to do is "inspect and adapt" and modify our process so that the surprises aren't surprises. Something that's just part of the "normal way of doing business" should never be described as an impediment – our job is to modify our process so that it adapts for it, and it actually becomes part of the "normal way of doing business."

General Budget / Planning Problem

This issue about bugs is just the simplest, and most common, example of a more general problem, which is the problem about how to manage the "known unknowns;" that is, those things that we know will happen, but we don't know exactly what they are. Of course, the most common example is bugs in other systems, but there are other examples, such as:

- Going on sales calls with virtually no warning,
- Helping our systems people with common hardware and software maintenance issues, and
- Doing the day-to-day things that our managers find for us to do, like reviewing resumes or doing interviews.

Issues like these are quite common, and they cause us to have problems with our budgeting and planning for the Sprint. In general, what these things can do to us if we don't find a way to make it part of "the normal way of doing business" is throw off our plans and cause us to miss our Sprint goals. Since they're so common, there must be a simple way to manage them, and there is.

Placeholder Stories to Manage Budget

The solution is so simple that it's actually elegant. What we do is put stories in our Backlog in order to hold (or contain) a budget for the Story Points that we know we're going to need to spend on these things. We call these stories Placeholder¹ stories, as they hold a place in our Sprint Backlog for stories that we don't know the details about yet.

There are two basic ways to do this. The first is to put actual stories in the Sprint Backlog that will be replaced by the real stories once they show up. For example we might put stories in the database called "SouvSite bug 1", "SouvSite bug 2", "SouvSite bug 3", and so on, each of which is a medium-sized story that has been given a size of 4 Story Points. I call this the "bugs to be named later" solution, and it's a fairly common pattern to use.

The second way to do it – which I now prefer – is to put in Placeholder stories (actually epics) that contain a number of Story Points that will be consumed as the actual stories show up (these are the stories within the epic).

For example, we may have a story called "Souvsite Bugs" which we assign a value of 20 Story Points to. As the actual bugs show up they consume the Story Points (based on

¹ I have heard some people call them "Overhead" stories, but I prefer to focus on the fact they are holding space in our Sprint Backlog, rather than making an assumption about the type of work that is being done.

their actual size) and the number of Story Points remaining in the “Souvsite Bugs” epic is decreased accordingly. Some other example stories of this type might be:

- “Sales Support”, which may contain 20 Story Points that we plan to spend on the “random” sales calls will be asked to assist in,
- “Admin Support”, which could contain Story Points we plan to spend helping our admin folks, or
- “Management Support”, which will contain Story Points we will be required to spend doing things like reviewing resumes, interviews for hiring new people, and other management-type overhead work that our Team’s people are required to do.

We might even go all out and have a story just called “Chores” that will hold the 30% of our total budgeted Story Points that we expect to spend on chores within the Sprint. There are lots of ways to do this, but they all involve putting epics in the Sprint Backlog that hold Story Points that we are budgeting for future, unknown-but-expected, stories.

As we’ll see in section 6 on Release Planning, this notion of Placeholder epics is at the core of the mechanics of Release Planning the way I recommend doing it.

Must have a Robust BackBurner

Of course, there is no guarantee that we will actually spend these Story Points that we have set aside. So what we do is make sure that the Back Burner part of our Backlog has some stories that are “ready to go” if we get to the end of our Sprint and still have some Story Points left over.

This shouldn’t be a surprise, this is what our Back Burner is for, anyway. Remember that it contains stories that are “one conversation away” from being committed to – they are basically ready to go.

So, if we have a robust Back Burner, as were supposed to, to make and use this pattern of having the Placeholder stories with no problem at all. Note that “robust” does not mean “big”. We still must make sure we don’t have *too many* stories on the Back Burner, as this would be considered waste, from a Lean perspective. We discussed this issue earlier (in chapter xx on the Backlog), and the guidance still stands. So, we need enough to fill in any gaps we may create, and be ready for the next sprint’s planning meeting, but not much more than that.

Discussion

The problem we're discussing here is quite common for scrum teams, and the use of Placeholder stories is a very simple and elegant solution. There is a variant of scrum that eliminates these problems altogether, and I refer to that variant as the Kanban(ish) variant. We will discuss this variation in Chapter 4.7.