Unheralded Alternatives to User Stories

The Scrum guide does not mention User Stories. Still, all the Scrum teams I worked with use them. Many Scrum teams apply User Stories to a fault: User Stories may not be the right tool for the job.

Maarten Dalmijn - Follow - Published in Serious Scrum, Nov 20, 2018

Contents

1. Why are User Stories so popular

According to the Scrum guide, few attributes are mandatory for Product Backlog items:
Product Backlog items have the attributes of a description, order, estimate, and value. Product Backlog items often include test descriptions that will prove its completeness when "Done".
As you can see, User Stories are not mentioned anywhere. They are also not mentioned in the rest of the Scrum guide. Why are User Stories so popular if Scrum teams are under no obligation to use them?

There are **three** reasons for User Stories being so popular:

1. **User Stories are easy to understand and to get started with**. A lot of information is available on User Stories. With a quick search you can find documentation, books and courses to learn more about User Stories.
2. **User Stories are versatile.** You can use them for discovery. You can use them for delivery. It is easy to split User Stories up in smaller ones. It is hard to go wrong, when you choose to write a User Story.

3. **People are unfamiliar with alternatives and when to best use them**. It is easy to do what you did before.

For the last reason, I am writing this article. I want to help spread knowledge about alternatives to User Stories. It pays off to know the right time to use each template for your Product Backlog items and you will get a better result by doing so.
I will discuss 5 alternatives to User Stories together with their pro's and con's.

- Initiatives
- Job Stories
- Problem Stories
- Improvement Stories
- Feature-Driven Development features

2. Initiative: Super-Epic for grouping Epics together

Initiatives allow teams to work together on functionality that spans across multiple teams. Let's say you want to deliver a functionality that requires collaboration by four different teams. You start by creating an initiative that is shared by the four teams. It is then up to the teams to split their work up in one or more Epics. These Epics will then need to be refined and split up further.

Pros:

- Initiatives help manage uncertainty and unclarity across different teams. It is up to the different teams to coordinate and split the work up in Epics and even smaller backlog items

Cons:

- No well-defined template for initiatives.
- Initiatives have a tendency to become high-level and far removed from the actual work. All teams need to have same understanding what the initiative will contribute to before breaking it down in Epics.
-

Further reading:

Jira Portfolio - The Fundamentals - Atlassian Blogs
Getting to know the fundamentals of Jira Portfolio, starting with understanding themes and initiatives. What are they…
www.atlassian.com

3. Job Stories: what is the context and causality of your feature?

Job stories are rooted in jobs-to-be done and have the following template:
When < situation>, I want < motivation > so that < expected outcome>

Pros:

- Job Stories are great for discovery. You really need to understand the context of your user and the job they want to perform in order to write a good Job Story.
- Job Stories do not introduce implicit assumptions about the user. You need to make the situation of the user, their motivation and the progress they are looking for in their lives explicit. This means everybody will be on the same page.
- Can easily be split up in smaller Product Backlog items by using User Stories or Problem Stories.

Cons:

- Job Stories are hard to split up in smaller Product Backlog items by using Job Stories. The situation, motivation and expected outcome often remain the same for the overall feature. So Job Stories work best on Epic level.

- Job Stories may introduce confusion in delivery, because they are really geared toward the problem domain. In delivery, you need to understand the problem but you need to describe the solution really well. Job Stories provide a poor template conveying the specific solution you came up with.

Further reading:
Replacing The User Story With The Job Story
Too many assumptions are dangerous
jtbd.info

5 Tips For Writing A Job Story
Learning from a new way to define features and products.
jtbd.info

4. Problem Stories: what is the problem you are trying to solve and how do you intend to solve it?

A Problem Story has the following template:
In order to < solve problem >, we will <build solution >
The solution is added by the team after discussing the issue at refinement.
Pros:

- Problem Stories make it easy to separate the problem and solution space.
- Easy to create and understand.
- Works great for more technical features. Sometimes it is really hard to write a User Story, because there isn't just one user affected it is unclear who is affected.
- Easy to split up in smaller Problem Stories.

Cons:

- You need to understand the problem you want to solve really well before you create a Problem Story. Discovery is necessary prior to creating a problem story.
- Lacks context of the user. Dangerous to use when the context of the user matters, because you lose this information and you may build the wrong solution. So beware!

Further reading:
https://www.linkedin.com/pulse/new-way-writing-user-stories-brought-me-back-realising-adrian-kerry

5. Improvement Stories: what is the current situation and the desired situation you want to have?

An Improvement Story has the following template:
We have<current situation>, we want to have <desired situation>
Pros:

- Easy to create and understand.
- Works really well for incremental improvements, where using a User Story or Problem Story would take effort without adding value. Especially if you have User Story it relates to where the context is already adequately described.
- Great if you want to prevent the whole 'Is it a bug or not?' discussion. Who cares if it is a bug or not, just as long as someone picks it up.

- Prevent duplication of effort by complementing User Stories. You can link to existing User Stories that already describe the context of the user really well.

Cons:

- Improvement Stories focus on the solution, so not good for discovery. To be used when you have a simple solution and it is obvious why it needs to happen.
- Not very versatile. To be used in specific circumstances.

Further reading:

Improvement Stories: a simple alternative to User Stories
You are the Product Owner for a Scrum Team. Your team has just released a brilliant feature they have been working on…
medium.com

6. Feature-Driven Development features: great for picking up technical work

FDD features have the following format:
<action> the <result><by|for|of|to|in><object>
This may sound quite cryptic, so here are some concrete examples:

- Generate unique identifier for uploaded images
- Send advance shipping notice to the warehouse
- Reduce stock levels of sold items

This template works great for technical tasks, where what you need to do is far removed from users.

Pros:

- Easy to use and understand.
- Easy to split up in smaller chunks.
- Work great when what you are building things that are far removed from users. Should be used when writing a User Story feels silly and convoluted.

Cons:

- Context of both the user and the problem are missing. You need to be extremely confident what you are building is the right thing.
- Focused on the solution, so poor for discovery.

Further reading:

Not Everything Needs to Be a User Story: Using FDD Features
User stories are great. When you've got users, that is. Sometimes, though, the users of a system or product are so far…
www.mountaingoatsoftware.com

7. Different backlog item types serve a different purpose: pick the right one for your situation

It is easy to keep on using User Stories and Epics. It is safe to do, as you have done many times before. However, there are clear situations where it may pay off to try something new. Each backlog item type serves a specific purpose and picking the right one makes your life easier.
Building something big that requires frequent collaboration between multiple teams? An initiative will help start the discussion and provides a good platform for breaking down work in multiple Epics.

Building a new product and busy with discovery? Job stories are better suited than User Stories for figuring out how to make the lives of your users better.
Have a list of small improvements that build upon something you've released? Improvement Stories will save you a lot of time and easier to understand than User Stories.
Have a clear problem you need to solve that affects multiple users or the type of user does not really matter? Try a problem story. More signal and less noise.
Busy writing a user story that feels silly and convoluted because it is far removed from the end users? Use the Feature-Driven Development Feature template.

Written by Maarten Dalmijn

66K Followers
·Writer for

Serious Scrum

https://dalmyn.com
Follow