Why the Three-Part User Story Template Works So Well

by Lisa Jacobson • 9 Comments

There is no magic template that must be used for user stories. They can be written in any number of ways. But the most popular way of writing user stories is with this template:

As a ..., I ... so that .....

This template originated with agile coach Rachel Davies at an English company, Connextra, in the early 2000s. It has since become a recognized standard for expressing user stories.

In this article, let's take a look at the three elements of this standard template, why that template has stood the test of time, and the strengths and weaknesses of the template.

The Three Elements of the Standard Template

I've described the three parts of the standard template in various ways over the years. In User Stories Applied, I described the three elements this way:

> As a (role), I want (function) so that (business value).

I later shifted to referring to the three elements as the role, goal, and reason. Ultimately I settled on just referring to them much more simply as who, what, and why. The three elements of the standard user story template address:

- Who wants the functionality
- What it is they want
- Why they want it

Let's look at each of these parts of the user story template.

Who: The First User Story Template Element

The users of a typical product or system can generally be categorized. As an example, I'm typing this into Google Docs right now. I could be considered a *casual user* of Docs. I don't use most of its features. I've never clicked on its Add-Ons menu to even see what's there. I don't do a lot of fancy formatting because most of what I write gets moved onto my website and is formatted there. So, let's call me a *Casual User*.

Others who do use those features could be called *Power Users*.

I usually send my blog drafts to an editor. And so, *Editor* could be another role when identifying the various different types of Google Docs users.

These user roles form the first part of the standard user story template. So someone developing a word processor might have a user story of "As a *power user*, I want a spellchecker…"

What: The Second User Story Template Element

The second part of the standard template states what is desired or needed. This is commonly stated as "I want…." In fact, for years my conference presentations and other trainings on user stories had a slide saying "I want…"

I eventually realized that was inaccurate. Sometimes the functionality being described is not something the user role wants at all. For example:

- As a member, I am required to enter a strong password....

No (normal) user *wants* to enter a password with lots of characters, three special symbols, no repeating characters, and at least a couple of uppercase letters. At best, we understand why it's needed, but personally I'd prefer the system should just magically know it's me and let only me in.

So, these days I no longer include *want* when showing the template in courses or presentations. Instead of always being I *want*, sometimes it's I'm *required, forced, need to* or more.

Why: The Third User Story Template Element

The final part of the standard template is *why* the user wants the functionality being described in the user story. This is provided after the *so-that* portion of the template. For example, a fully expressed version of the earlier spell checker story could be "As a power user, I want a spell checker so that I don't need to worry about spelling ~~misteaks~~ mistakes."

The *so-that* clause of a story is important because understanding why a user wants what is described in the *what* portion of the story helps the team decide how best to implement the functionality.

As an example, look no further than our spell checker story. Suppose it were provided to a team without a so-that clause as simply: *As a power user, I want a spell checker*. That very likely would lead a team to develop what all early spell checkers were: After-the-fact tools run on a document a er it was written.

But our power user doesn't want an extra step after finishing writing a document. Rather, the power user wants what seems more standard today: real-time correction of mistakes as they are made. What the user really wants is given by the so-that clause: the user doesn't want to worry about spelling mistakes.

Should the So-That Clause Be Optional?

When I present on user stories during in-person courses, I am often asked to justify my seemingly contradictory view that the so-that clause is the most important part of a story yet it shouldn't be mandatory.

I don't consider it mandatory because sometimes it just doesn't add any value. Consider a story about logging in: *As a member, I am required to log in*. What so-that clause can you add to this story that adds value or clarifies the intent of the story? Do any of these really help or are they just superfluous text added to comply with a template:

- As a member, I am required to log in so that only I can access my personal data.
- As a member, I am required to log in so that others can't access my personal data unless I've given them my credentials.
- As a member, I am required to log in so that the system knows it's me.
- As a member, I am required to log in so that hackers are kept out.

While I don't consider the *so-that* clause mandatory, I do write it nearly all of the time. I reviewed a recent backlog I'd written and 62 of 64 stories (97%) had *so-that* clauses. A small number of occasions prevent me from considering it mandatory.

Strengths of The 3-Part User Story Template

So what's so good about this template that it has stood the test of time? After all, a practice introduced in the early 2000s and growing in acceptance so many years later must have something going for it. I think there are four main strengths to the template.

It Covers Three of the Five Ws

I started university as a journalism major. I think I romanticized movies filled with cigar-chomping newspaper reporters—films like *Ace in the Hole, Citizen Kane, It Happened One Night,* and *All the President's Men*. Journalists are trained that any newspaper article needs to answer five questions that each begin with a W:

- Who?
- What?
- When?
- Where?
- Why?

User stories address three of the five--Who, What and Why. When discussing product or system requirements, it seems reasonable to leave When and Where out as the answers would usually be "right now" and "in the product."

The Elements Are Presented in the Right Order

I think the elements--who, what, and why--are presented in the right order. Think about any story--not a user story but a movie, a novel, or an anecdote or joke you want to tell a friend. The most important thing in that story is almost always who is doing it. We call that person the *protagonist*.

When we watch a movie, we need to care about the protagonist before we care about the plot. I don't care one way or the other the Death Star being blown up until I see a little of myself in Luke, Han or Leia and can *relate* to them.

Only after I know *who*, do I care about *what* and *why*. The standard user story template puts these in that order.

The Story Is Told from a First-Person Perspective

We like stories about ourselves. (Well, unless we're teenagers and our parents are telling stories about the not-so-cute things we did when we were babies.) The next best thing to a story about me is a story about you. The least interesting is a story about the guy across town whom I've never met.

Stories about I and you and we and she and he are interesting.

The Beatles knew this. They very deliberately stuffed as many personal pronouns as they could into their song titles. And if they couldn't fit a personal pronoun in the song title, they put as many as they could in the song's lyrics.

The Beatles' first British album had pronouns in 8 of 14 song titles (57%). In the 19 minutes and 30 seconds of those eight songs The Beatles used 325 personal pronouns, one pronoun every 3.6 seconds. That worked so well their second album had pronouns in 64% of the titles. The Beatles then pushed it to 79% on their third album.

In an interview with Billboard magazine, Beatle Paul McCartney said this was very deliberate: “All our early songs contained 'me' or 'you.' We were completely direct and shameless to the fans: *Love Me Do, Please Please Me, I Want to Hold Your Hand.*”

The standard user story template starts with *I*, the most personal of personal pronouns. I have no basis for this claim--I’m no brain scientist--but I swear something happens when we have a user story that starts with *I*. We relate to that story more closely than we would if the same thing were written as a traditional *The system shall…* statement.

Paul McCartney and John Lennon knew this and used it to boost their careers. Agile teams do the same when using the three-part user story template.

Our Stakeholders Are Immediately Comfortable Filling in the Blanks

Before user stories came along, I loved use cases. Or I tried to love use cases. I really did love them. But I could never get the stakeholders with whom I worked to love them as much as I did.

They were just too far afield from how stakeholders thought about their work. Stakeholders don’t think about secondary actors or preconditions or postconditions. And so use cases never worked as well in practice as I thought they should.

I’ve never had that problem with user stories. I can conduct a story-writing workshop with stakeholders simply by writing As a _____, I _____ so that _____ on a whiteboard and telling them we’ve gathered to fill in the blanks as many times as we can.

Stakeholders get it. It’s a natural way of speaking for them.

Other templates for expressing stories have been proposed, and some have advantages, but most are less natural ways of speaking. For example, Behavior-Driven Development is a great way for expressing tests or specifying by example. Martin Fowler describes its given-when-then syntax as “a style of representing tests.” And it’s fantastic for writing test specifications but it’s not as good for communicating with customers. In part this is because its template is less natural. I’ve been speaking since I was 2 or 3. I don’t know if I’ve ever started a sentence with *given*. I’ve definitely never used *given, when* and *then* in that order in a sentence. I’ve countlessly said “I want *this* so that *that*.”

Drawbacks of The 3-Part User Story Template

There are two drawbacks to the standard user story template that are worth mentioning.

Too Many Stories Are Written as Just “As a user…”

Too often team members fall into a habit of beginning each user story with “As a user…” Sometimes this is the result of lazy thinking and the story writers need to better understand the product’s users before writing so many “as a user…” stories.

But other times it may indicate a system not well suited to user stories. This happens because too many people associate being agile with writing user stories. To be agile, they reason, you have to write user stories even when there are no users.

I worked on a financial compliance tracking system. The vast majority of what the system did would never be seen or reported. If a compliance issue was discovered, however, reports would be generated and individuals would be notified. This system benefited from user stories for that small subset of the product's overall functionality. But user stories were inappropriate for the rest of the system.

In cases like these, teams need to use alternative ways of expressing what the product needs to do. The syntax used by job stories or Feature-Driven Development's features might be better choices.

The Template Is Too Often Slavishly Followed

By all means, common sense needs to be a guiding principle for agile teams. When expressing something in the standard user story template doesn't make sense, don't use that template. Write it another way, including very possibly just free-form text.

Earlier today I wrote a product backlog item of "Change how webinar replay reminders are sent on the last day the way we discussed on Friday's call."

As a product backlog item, that sucks. It's not a user story. It's not BDD or even a job story. It's horrible. And if it doesn't get fixed soon, no one will even remember what bug was talked about on some forgotten phone call.

But, I knew the team would fix it soon and so what I wrote was good enough. The last thing I need would be a Scrum Master insisting I write it to follow a template created around the turn of the millennium, no matter how well that template works for most product backlog items.

What Do You Think

As a reader of this blog, I want to know what you think of the three-part story template so that I can learn how you're using it (or not). (See what I did there?) Please share your experiences in the comments section below.

Comments

I found it quite useful to combine the story pattern with BDD for acceptance criteria. We are using story mapping and as such, a user story looks like this:

Story
In order to...
as a...
I would like to...

Acceptance criteria
(i) Given...
When...
Then...

This provides a testable story and fulfilling acceptance criteria is part of our definition of done.