Editorial Board Thoughts

Who Will Use This and Why? User Stories and Use Cases

Kevin M. Ford

Perhaps I'm *that guy*. The one always asking for either a "user story" or a "use case," and sometimes both. They are tools employed in software or system engineering to capture how, and importantly why, actors (often human users, but not necessarily) interact with a system. Both have protagonists, but one is more a creative narrative, the other like a strict, unvarnished retelling. User stories relate what an actor wants to do and why. Use cases detail to varying degrees how that actor might go about realizing his desire. The concepts, though distinct, are often confused and conflated. And, because they classify as jargon, the concepts have sometimes been employed outside of technology to capture what an actor needs, the path the actor takes to his or her objective, including any decisions that might be made along the way, and all of this effort is undertaken in order to identify the best solution. By giving the actors a starring role, user stories and use cases ensure focus is on the actors, their inputs, and the expected outcome. They protect against incorporating unnecessary elements, which could clutter and, even worse, weaken the end product, and they create a baseline understanding by which the result can be measured. And so I find myself frequently asking in meetings, and mumbling in hallways: "What's the use case for that?" or "Is there a user story? If not, then why are we doing it?" You get the idea.

It's a little ironic that I would become this person. Not because I didn't believe in user stories and use cases – quite the contrary, I've always believed in the importance and utility of them – but because of a book I was assigned during graduate coursework for my LIS degree and my initial reaction. It's not just an unassuming book, it has a downright boring appearance, as one might expect of a book entitled "Use Case Modeling."[1] It's a shocking 347 pages. It was a joint endeavor by two authors: Kurt Bittner and Ian Spence. I think I read it, but I can't honestly recall. I assume I did because I was that type of student and I had a long Chicago El commute at the time. In any case, I know beyond doubt that I was assigned this book, dutifully obtained it, and then picked it up, thumbed through it, rolled my eyes, and probably said, "Ugh, really?"

And that's just it. The joke's on me. The concepts, and as such the book, which I've moved across the country a couple of times, remain near-daily constants in my life. As a developer, I basically don't do anything without a user story and a use case, especially one whose steps (including preconditions, alternatives, variables, triggers, and final outcome) haven't been reasonably sketched out.

"Sketched out" is an interesting phrase because one would think that if entire books were being authored on the topic of use cases, for example, then use cases would be complicated and involved affairs. They can be, but they need not be. The same holds for user stories. Imagine you were designing a cataloging system, here's an example of the latter:

> As a librarian I want my student catalogers to be guided through selection of vocabulary terms to improve both their accuracy and speed.[2]

Kevin M. Ford (kefo@loc.gov) is Librarian, Linked Data Specialist, Library of Congress.

That single-sentence user story identifies the actors (student catalogers), what they need (a "guided … selection of vocabulary terms"), and why ("to improve their accuracy and speed"). The use case would explore how the student catalogers (the actors) would interact with the system to realize that user story. The use case might be narrowly defined ("Adding controlled terms to records") or might be part of a broader use case ("Cataloging records"), but in either instance the use case might go to some length to describe the interaction between the student catalogers and the system in order to generate a clear understanding of the various interactions. By doing this, the use case helps to identify functional requirements and it clearly articulates user/system expectations, which can be reviewed by stakeholders *before* work begins and used to verify delivery of the final product.

As I have presented this, using these tools might strike you as overly formal and time-consuming. In many circumstances they might be, if the developer has sufficient user and domain knowledge (rare, very, very rare) and especially if the "solution" is not an entirely new system but just an enhancement or augmentation to an existing system. Yet, whether it is a completely new system being developed by someone who has long and profound experience with the domain or a simple enhancement, it may be worth entertaining the questions/process if even informally. I find it is often sufficient to ask "Who will use this and why?" Essentially I'm asking for the "user story" but dispensing with the jargon. Doing so may lead to additional questions, the answers to which would likely check the boxes of a "use case" even if the effort is not identified as such, and it certainly ensures the user-driven nature and need of the request.

This might all sound obvious, but I like to think of it as defensive programming, which is like defensive driving. Yes, the driver coming up to the stop sign on my right is going to stop, but I take my foot off the gas and position it over the brake just in case. Likewise, I'm confident the functional requirements I'm being handed have been fully considered and address a user need, but I'm going to ask for the user story anyway. I'm also leery of scope creep which, if I were to continue the driving analogy, would be equivalent to driving to one store because you need to, but then also driving to two additional stores for items you think might be good to have but for which you have no present need. It's time-consuming, you've complicated your project, you've added expense to your budget, and the extra items might be of little or no use in the end. The number of times I've been in meetings in which new, additional features are discussed because the designers think it is a good idea (that is, there has been no actual user request or input sought) is alarmingly high. That's when I pipe up, "Is there a user story? If not, then why are we doing it?"

User stories and use cases help focus any development project on those who stand to benefit, i.e. the project's stakeholders, and can guard simultaneously against insufficient planning and software bloat. And the concepts, though most often thought of with respect to large-scale projects, apply in all circumstances, from the smallest feature request to an existing system to the redesign of a complex system. If you are not in the habit of asking, try it next time: Who will use this and why?

ENDNOTES

[1] Kurt Bittner and Ian Spence, *Use Case Modeling* (Boston: Addison-Wesley, 2003). Also useful: Alistair Cockburn, *Writing Effective Use Cases* (Boston: Addison-Wesley, 2001).

[2] "Use Case 3.4: Authority tool for more accurate data entry," Linked Data for Libraries (LD4L), accessed March 1, 2019, https://wiki.duraspace.org/display/ld4l/Use+Case+3.4%3A+Authority+tool+for+more+accurate+data+entry.