



Michael James
Michael James is a software process mentor and Certified Scrum Trainer, focusing on the engineering practices that enable Agile project management. Having worked in the software industry for more than 20 years as a software developer (formerly “architect”), he has experience in automated testing that predates the Extreme Programming movement; formal, phased, high-ceremony processes based on DOD-STD-2167A; chaotic non-processes of the dot-com era; and Agile processes including Scrum and XP.

User Story Examples and Counterexamples

Posted by Michael James on June 21, 2007 at 7:40 p.m.

People who share my background in traditional requirements analysis often have a hard time coming up with Product Backlog Items representing thin vertical slices of potentially shippable product. Here is some material I’ve started using in teaching this.

GUIDELINES

User Stories may assume template form (see [User Stories Applied](#)¹ by Mike Cohn):

- “As a [role], I can [capability] so that [rationale].”

Or expressed as noun phrase:

- “Image clipboard”

Or a one- to two-sentence story:

- “Busy streets are highlighted on the map.”

User stories, like all Product Backlog Items, should contain or clearly imply *acceptance criteria* (definition of “done”). Write these on the back of the index card or in the “Description” field if you’re using ScrumWorks. (By the way, if you’re still using an [information refrigerator](#)² like Microsoft Excel for your product backlog, maybe you haven’t heard ScrumWorks Basic is still free!)

Bill Wake³ has given us the INVEST⁴ mnemonic to help remember the characteristics of a well-formed user story:

- I - Independent
- N - Negotiable
- V - Valuable
- E - Estimable

¹ <http://www.amazon.com/User-Stories-Applied-Development-Addison-Wesley/dp/0321205685>

² <http://c2.com/cgi/wiki?InformationRefrigerator>

³ <http://xp123.com>

⁴ <http://xp123.com/xplor/xp0308/index.shtml>

danube

Danube Technologies, Inc.
520 SW 6th Ave, Suite 940
Portland, OR 97204 USA
1.888.5.danube
T: +1.503.248.0800
F: +1.503.248.0801

- S - Small
- T - Testable

EXAMPLES

What does this look like in practice?

1. A bank customer can change his PIN.
 - Acceptance Criteria:
2. As a student, I can find my grades online so that I don't have to wait until the next day to know whether I passed.
 - Acceptance Criteria:
3. One level of undo.
 - Acceptance Criteria:
4. As a book shopper, I can read reviews of a selected book to help me decide whether to buy it.
 - Acceptance Criteria:
5. As an author, I want the spell checker to ignore words with numbers so that only truly misspelled words are indicated.
 - Acceptance Criteria:

COUNTEREXAMPLES

How do we know when we're missing the mark? Ultimately, "User stories are a promise for a conversation" (Ron Jeffries). If Product Owner and Team both know what they mean, you're off to a good start. But you can probably save some time by avoiding the common pitfalls below:

1. "Design brochure layout."
 - Drawbacks: Not *Independent*, no business *Value*. This is a *task* representing a horizontal architectural layer or phase. The architecture will be done in a vacuum, possibly contributing to analysis paralysis.
 - Better: "As a dog owner, I can find a meal schedule on the brochure so I know whether this doggy day care center is appropriate for my hungry dog."
 - This will lead to only the necessary amount of design to support this Sprint's features. The layout might change the next Sprint, but rework is cheaper than no work.
2. "Write game rules."
 - Drawbacks: Not *Independent*, no business *Value*, not *Small*.
 - Better: "As a newbie game player, I want to know who goes first so we can start the game."
 - Better: "As a competitive gamer, I want a way to leapfrog my opposing players."
3. "I want the brochure to be colorful."
 - Drawbacks: Not *Independent*, not *Estimable* (without knowing other features of brochure), not *Small*.
 - This is an easy trap for those of us who grew up with the habit of writing, "The JFIDM _shall_ comply with the IEEE-488 interface specification."

The logo for Danube Technologies, featuring the word "danube" in a white, lowercase, sans-serif font on a blue rectangular background.

Danube Technologies, Inc.
520 SW 6th Ave, Suite 940
Portland, OR 97204 USA
1.888.5.danube
T: +1.503.248.0800
F: +1.503.248.0801

- Better: Use “colorful” and other cross-cutting requirements as *acceptance criteria* on each of the specific features in the backlog to which they apply.
- 4. “As Product Owner, I want a list of highly rated restaurants on the brochure.”
 - Drawbacks: It’s not only about you!
 - Better: Focus on your end users and stakeholders. “As a gourmet tourist, I want a list of highly rated restaurants on the brochure.”
 - Better: “As the Chicago Public Health Department, I want warnings about restaurants that serve raw ingredients so that tourists don’t get sick on our dime.”
- 5. “Play test the game.”
 - Drawbacks: Not *Independent*. Encourages phase-wise development.
 - Better: Make testing, refactoring, etc. default acceptance criteria on every Product Backlog Item.
 - But: If you failed to fully test and refactor in previous Sprints, you are in technical debt! You are already working on a legacy product⁵. In this case, you may need to make testing and refactoring first class Product Backlog Items to make up for your sins.

User stories are simple things. “Simple” is not always the same as “easy.” Happy unlearning!

--mj
Michael James
Software Process Mentor

The logo for Danube Technologies, featuring the word "danube" in white lowercase letters on a blue rectangular background.

Danube Technologies, Inc.
520 SW 6th Ave, Suite 940
Portland, OR 97204 USA
1.888.5.danube
T: +1.503.248.0800
F: +1.503.248.0801

⁵ http://danube.com/blog/michaeljames/how_to_survive_technical_debt