Writing Effective User Stories for Agile Requirements

Mike Cohn
September 26, 2005

Mike Cohn—background

- Programming for 20 years
- Author of
  - User Stories Applied
  - Agile Estimating and Planning
  - Java, C++, database programming books
- Founding member and director of the Agile Alliance and the Scrum Alliance
- Founder of Mountain Goat Software
  - Process and project management consulting and training
Today’s agenda

- What user stories are
- Users and user roles
- Gathering stories
- INVEST in good stories
- Why user stories?

Ron Jeffries’ Three Cs

- Card: Stories are traditionally written on note cards. Cards may be annotated with estimates, notes, etc.
- Conversation: Details behind the story come out during conversation with customer
- Confirmation: Acceptance tests confirm the story was coded correctly
Samples – Travel reservation system

As a user, I can reserve a hotel room.

As a user, I can cancel a reservation.

As a user, I can restrict searches so that I only see hotels with available rooms.

As a vacation planner, I can see photos of the hotels.

Where are the details?

- As a user, I can cancel a reservation.
  - Does the user get a full or partial refund?
    - Is the refund to her credit card or is it site credit?
  - How far ahead must the reservation be cancelled?
    - Is that the same for all hotels?
    - For all site visitors? Can frequent travelers cancel later?
  - Is a confirmation provided to the user?
    - How?
Details added in smaller “sub-stories”

As a user, I can cancel a reservation.

- As a premium site member, I can cancel a reservation up to the last minute.
- As a non-premium member, I can cancel up to 24 hours in advance.
- As a site visitor, I am emailed a confirmation of any cancelled reservation.

Details added as tests

- High level tests are added to the story
  - Can be used to express additional details and expectations

As a user, I can cancel a reservation.

- Verify that a premium member can cancel the same day without a fee.
- Verify that a non-premium member is charged 10% for a same-day cancellation.
- Verify that an email confirmation is sent.
- Verify that the hotel is notified of any cancellation.
Today’s agenda

- What user stories are
- Users and user roles
- Gathering stories
- INVEST in good stories
- Why user stories?

“The User”

- Many projects mistakenly assume there’s only one user:
  - “The user”
- Write all stories from one user’s perspective
- Assume all users have the same goals
- Leads to missing stories
Travel Site—Who’s the user?

- **Mary**: Frequent flier who never knows where she’ll be
- **Laura**: Wants to schedule her family’s annual vacation
- **Jim**: Frequent flier who flies every week but always to the same place
- **Howard**: Mary’s assistant; books her reservations
- **Dominic**: Hotel chain Vice President; wants to monitor reservations

User roles

- Broaden the scope from looking at one user
- Allows users to vary by
  - What they use the software for
  - How they use the software
  - Background
  - Familiarity with the software / computers
- Used extensively in usage-centered design
- Definition
  - A user role is a collection of defining attributes that characterize a population of users and their intended interactions with the system.

Source: Software for Use by Constantine and Lockwood (1999).
User role brainstorming

- Brainstorming meeting
  - Customer, developers, anyone who understands a product’s intended users
- Everyone grabs a stack of cards
- Write role names on cards
  - As fast as possible and with no judgment
    - No turns
  - Place card on table
  - Call out role name as you place it
User role brainstorming

We’ve been hired by fBay to create “the best new web auction site since eBay.”

- Brainstorm the user roles who will interact with this site.

User role modeling steps

- Brainstorm an initial set of user roles
- Organize the initial set
- Consolidate roles
- Refine roles
Organize the initial set

- Arrange cards spatially to indicate overlapping and similar roles
  - Use any arrangement rules you want

Consolidate roles

- Discuss what is meant by each card
- Arrange cards spatially to indicate overlapping and similar roles
  - Use any arrangement rules you want
- Look for cards to
  - Combine
  - Replace with a more generic/different card
- Eliminate cards that are unimportant to the success of the product
Consolidating—an example

Organize and consolidate

1) Organize your initial set of user roles for fBay.
2) Consolidate the user roles.
Advantages of using roles

- Users become tangible
- Start thinking of software as solving needs of real people.
- Avoid saying “the user”
- Instead we talk about “a frequent flier” or “a repeat traveler”
- Incorporate roles into stories
- “As a <role>, I want <story> so that <benefit>.”

Today’s agenda

- What user stories are
- Users and user roles
- Gathering stories
- INVEST in good stories
- Why user stories?
Gathering stories

- Common metaphors for requirements are wrong
  - “Eliciting requirements”
  - “Capturing requirements”
- These metaphors imply
  - Users know the requirements but don’t want to tell us
  - Requirements need to be locked up once “captured”

The proper metaphor

- Trawling† for requirements
  - Trawl: “sift through as part of a search” (OAD)
- Metaphor captures these aspects:
  - Requirements can be captured with different sized nets
  - Requirements change, mature, possibly die
  - Skill is a factor

†Mastering the Requirements Process by Suzanne and James Robertson, 1999.
A little is enough, or is it?

- Agile processes acknowledge that we cannot trawl with such a fine net that we can write all the user stories upfront
- However,
  - This doesn’t mean we shouldn’t write as many as we can

Techniques for trawling for stories

- Questionnaires
- Observation
- User interviews
- Story-writing workshops
Questionnaires

- Good technique for learning more about stories you already have
- If you have a large user base, great way to get information to help prioritize stories
- Not effective as a primary means of trawling for new stories

Observation

- Great way to pick up insights
- Two approaches
  - Just observe, with or without user’s knowledge
  - Have the user demonstrate to a group how she uses the software
Observation example

- Stated need:
  - “We need a large text field to summarize.”
- Observed need:
  - Have the system record the user’s choices

Interviews

- Default approach taken by many teams
- Selection of interviewees is critical
  - Try to interview as many user roles as possible
- Cannot just ask “So whaddaya want?”
  - Most users are not adept at understanding their true needs
Context matters

“My wife and I split up…”

“He’s no longer with us…”

My context isn’t your context

“Dad, make it warmer.”
A horrible question

- This question sucked two years out of my life

  - "Would you like it in a browser?"
  - "Of course, now that you mention it!"

We can do better

  - "What would you think of having this app in a browser rather than as a native Windows application even if it means reduced performance, a poorer overall user experience, and less interactivity?"

- It’s open
  - Full range of answers
- But it has too much context
The best way to ask

“What would you be willing to give up in order to have it in a browser?”

- We want to ask questions that are
  - Open-ended
  - Context-free

Beware of assumptions

- What time did the sun rise in Boston, MA, on October 10, 1582?
- Draw a single line that crosses each line segment in this figure exactly once:
Seeking a solution

- “Dad, do you know the answer?”

It’s my problem, I know the solution

- Having a problem does not uniquely qualify you to solve it
- “It hurts when I go like this…”
We need to stop asking users

- Since users don’t know how to solve their problems, we need to stop *asking*
- We need to *involve* them instead

**Empirical design**
- Designers of the new system make decisions by studying prospective users in typical situations

**Participatory design**
- The users of the system become part of the team designing the behavior of the system

---

**Story-writing workshops**

- Includes developers, users, customer, others
- Goal is to write as many stories as possible
  - No prioritization at this point
- Uses low-fidelity prototyping and brainstorming techniques
Start with epics and iterate

Frequent Flyer

As a frequent flyer, I want to see check my account.

As a frequent flyer, I want to book a trip.

< Epic #3 >

As a frequent flyer, I want to book a trip using miles.

As a frequent flyer, I want to re-book a trip I make often.

As a frequent flyer, I want to request an upgrade.

As a frequent flyer, I want to see if my upgrade cleared.

A low-fidelity prototype

Home Page
News
Hot Deals
Search Fields

Hot Deal Details
Location info
Weather

Hotel Results
List of hotels
Blurb about each

Hotel Details
Info about hotel
Map
Local attractions

News

Weather
Low-fidelity prototyping

- Use paper, note cards, white board, big Post-its
- Prototype is of components or areas within the application, not of actual screens
  - Hotel Results could be on Home Page or be a separate page
- Doesn’t require knowledge of how screens will look
- Throw it away a day or two later
- Works better to go depth-first

Creating the low-fidelity prototype

- Start with an empty box:
  - “Here’s the main screen in the system”
- Ask open-ended, context-free questions as you go:
  - What will the users most likely want to do next?
  - What mistakes could the user make here?
  - What could confuse the user at this point?
  - What additional information could the user need?
- Consider these questions for each user role
A mini story-writing workshop

Write some user stories for fBay based on the roles you identified.

Tip: try this template:
“As a <role>, I want to <story> so that <benefit>.”

Today’s agenda

- What user stories are
- Users and user roles
- Gathering stories
- INVEST in good stories
- Why user stories?
What makes a good story?

- Independent
- Negotiable
- Valuable
- Estimatable
- Small
- Testable

INVEST

Thanks to Bill Wake for the acronym. See www.xp123.com.

Independent

- Avoid introducing dependencies
  - Leads to difficulty prioritizing and planning

A company can pay for a job posting with a Visa card.

A company can pay for a job posting with an AmEx card.

A company can pay for a job posting with a MasterCard.

- The first of these stories will take 3 days to develop
  - It doesn’t matter which is first
  - The others will take 1 day
Making stories independent

- Combine the stories: A customer can pay with a credit card.

- Split across a different dimension:
  - A customer can pay with one type of credit card.
  - A customer can pay with two other types of credit cards.

- Write two estimates and move on: 3 days if first; 1 otherwise

Negotiable

- Stories are not
  - Written contracts
  - Requirements the software must fulfill
- Do not need to include all details
- Too many details give the impressions of
  - False precision or completeness
  - That there’s no need to talk further
- Need some flexibility so that we can adjust how much of the story gets implemented
  - If the card is contract then it needs to be estimated like a contract
Is this story negotiable?

Print dialog allows the user to edit the printer list. The user can add or remove printers from the printer list. The user can add printers either by auto-search or manually specifying the printer DNS name or IP address. An advanced search option also allows the user to restrict his search within specified IP addresses and subnet range.

Valuable

Stories must be valuable to either:

- **Users**
  - As a user, I can search for a job by title and salary range.
- **Purchasers**
  - Throughout the project, the team will produce documentation suitable for an ISO 9001 audit.
  - The development team will produce the software in accordance with CMM level 3.
  - All configuration information is read from a central location.
Stories valued by developers

- Should be rewritten to show the benefit
  - All connections to the database are through a connection pool.
  - Up to 50 users should be able to use the application with a five-user database license.
  - All error handling and logging is done through a set of common classes.
  - All errors are presented to the user and logged in a consistent manner.

Estimatable

- Because stories are used in planning
- A story may not be estimatable if:
  - Developers lack domain knowledge: As a new user, I am given a diabetic screening.
  - Developers lack technical knowledge: As a site visitor, I can elect to see all text in a larger font.
  - The story is too big: As a user, I can find a job.
Small

- Large stories (epics) are
  - hard to estimate
  - hard to plan
    - They don't fit well into single iterations
- Compound story
  - An epic that comprises multiple shorter stories
- Complex story
  - A story that is inherently large and cannot easily be disaggregated into constituent stories

Compound stories

- Often hide a great number of assumptions

As a user, I can post my resume.

- A resume includes separate sections for education, prior jobs, salary history, publications, etc.
- Users can mark resumes as inactive
- Users can have multiple resumes
- Users can edit resumes
- Users can delete resumes
Splitting a compound story

Split along operational boundaries (CRUD)

- As a user, I can create resumes, which include education, prior jobs, salary history, publications, presentations, community service, and an objective.
- As a user, I can edit a resume.
- As a user, I can delete a resume.
- As a user, I can have multiple resumes.
- As a user, I can activate and inactivate resumes.

Splitting a compound story, cont.

Split along data boundaries

- As a user, I can add and edit educational information on a resume.
- As a user, I can add and edit prior jobs on a resume.
- As a user, I can add and edit salary history on a resume.
- As a user, I can delete a resume.
- As a user, I can have multiple resumes.
- As a user, I can activate and inactivate resumes.
Other ways to split large stories

- Remove cross-cutting concerns
- Don’t meet performance targets
- Avoid splitting stories into tasks
- Avoid the temptation of related changes

Testable

- Tests demonstrate that a story meets the customer’s expectations
- Strive for 90+% automation

A user must find the software easy to use.

As a novice user, I am able to complete common workflows without training.

A user must never have to wait long for a screen to appear.

New screens appear within 2 seconds in 95% of all cases.
Fixing stories

1) Assess the stories you’ve written for fBay against the INVEST attributes.
2) Rewrite those that do not meet these criteria.
3) If you can’t figure out how to rewrite a story, save it for class discussion.

- Independent
- Negotiable
- Valuable
- Estimatable
- Small
- Testable

As an OEM procurement agent…

…I want the ability to search for suppliers based on criteria placed in an input interface. The results include the following functionality:

- I am able to select one or more suppliers from the list of suppliers in the database against which to perform the query.
- I input certain criteria into an interface, query the database, and receive results as to which suppliers meet the criteria.
- The results are shown in order from highest match to lowest match with a symbol to show complete match of required criteria and a bar to show overall match. The list should be paginated and have a certain discrete number of returns per page, with next/previous type navigation.
- The query should be performed on the Business Factors: Type of Business, Location, and Supplier Size.
- The query should be performed on technical factors: material, machine type, size, swing, axis.
Today’s agenda

- What user stories are
- Users and user roles
- Gathering stories
- INVEST in good stories
- Why user stories?

So, why user stories?

- Shift emphasis from writing to talking

If requirements are written down then

- The user will get what she wants
- At best, she’ll get what was written

“...You built what I asked for, but it’s not what I need.”
Actual examples

The user can enter a name. It can be 127 characters.

- Must the user enter a name?
- Can it be other than 127 chars?

The system should prominently display a warning message whenever the user enters invalid data.

- What does should mean?
- What does prominently display mean?
- Is invalid data defined elsewhere?

Additional reasons

- Stories are comprehensible
  - Developers and customers understand them
  - People are better able to remember events if they are organized into stories†
- Stories are the right size for planning
- Support and encourage iterative development
  - Can easily start with epics and disaggregate closer to development time

†Bower, Black, and Turner, 1979. Scripts in Memory for Text
Yet more reasons

- Stories support opportunistic development
  - We design solutions by moving opportunistically between top-down and bottom-up approaches†
- Stories support participatory design
  - Participatory design
    - The users of the system become part of the team designing the behavior of the system
  - Empirical design
    - Designers of the new system make decisions by studying prospective users in typical situations


Most importantly…

Don’t forget the purpose

- The story text we write on cards is less important than the conversations we have.
- “Stories represent requirements, they do not document them.”†

Mike Cohn contact information

- mike@mountaingoatsoftware.com
- (303) 810–2190 (mobile)
- (720) 890–6110 (office)
- www.mountaingoatsoftware.com