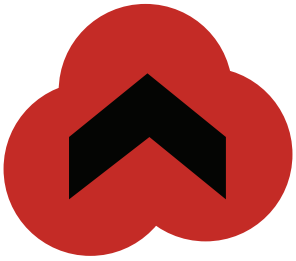

TREEHOUSE
TECHNOLOGY GROUP

CREATING EFFECTIVE USER STORIES

THE PRODUCT OWNER'S PERSPECTIVE

By: Philip Wess

Overview of a User Story

Epics vs User Stories

User Stories

- Derived from the practice of XP (Extreme Programming)
- (http://en.wikipedia.org/wiki/Extreme_programming)

- Stories are put on notecards or stickies to help visualize the process and development

- Meant to encourage conversation (hence the tone of a user story) through a succinct message

Acceptance tests are put on the back of the user stories, for example:

" As a user I need to be able to login to the website using FB Connect"

– Acceptance must be:

1. User is logged in
2. Picture for the user is captured
3. User can logout
4. Tracking is enabled
5. Etc…

   Product Owner works closely with a tester/developer and other stakeholders to define the acceptance criteria as they come closer to development

User Story description from Ron Jefferies [Creator of Xtreme Programming].

"A User Story is a story, told by the user, specifying how the system is supposed to work, written on a card and of a complexity permitting estimation of how long it will take to implement (very important). The User Story promises as much subsequent conversation as necessary to fill in the details of what is wanted [by the development team and stakeholders]. The cards themselves are used as tokens in the planning process after assessment of a business value and [possibly] risk. The customer or product owner prioritizes the stories and schedules them for implementation. "

Some examples of how a story should be constructed are the following:

1. As a <role> I would like to be able to <action> to achieve <business value>
2. As a <user role> I can <story> so that <benefit>
3. As <person name> I can <story> so that <benefit>
4. As a <user> I want to <goal> so that <value to attain>

The “So that” portion can be optional, but helps define the reason why to the stakeholders.

Once the stories are created, they are prioritized, related and grouped into epics. Sometimes the reverse happens when Epics are broken down into smaller user stories if needed require, and that is addressed in the subsequent section of “How to break down user stories.”

Epics

An Epic is a group of related users stories, when looking at them visually,

- Epics are at the top, Stories underneath
- Can be large features that need further breakdown

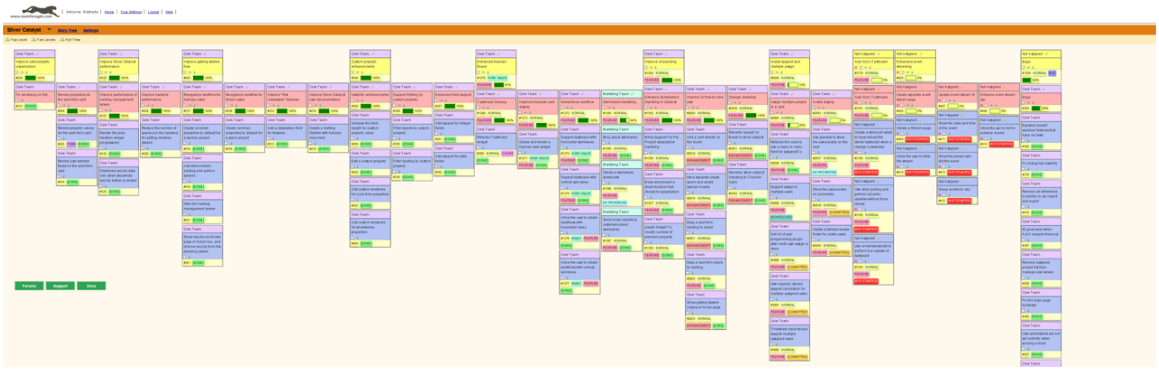

Figure 1

Example: “As a user I need to be able to login in to the website and view my list of available scholarships”

A further breakdown is required such as,

1. “As a user I need the ability to login on the homepage through Facebook connect”
2. “As a user I need the ability to provide filter criteria in order to narrow/expand my scholarship list”,
3. “As an administrator I need the ability to enter scholarships into the system so that a user can select one”,
4. “As a manager I need the ability to report on users who have selected a specific scholarship in order to market better products to them” etc…)

The Visual Display of Data

Epics and Users Stories should be organized into the workflow of building an application, once complete, this breakdown will display the information in a manner which will help the development staff and product owner determine the MVP or

minimum viable product (or that which we can release at first and receive critical user base testing)

When displaying stories and epics, you can view them in a two dimensional view as noted above. This allows the team to visualize the big picture, get on-board faster and understand dependencies

User Stories should be Organized by ➔Workflow ➓Richness

They can be:

1. A tool for identifying MVP or Minimal viable product
2. Drawing the Story Map (The path a user takes through the stories while navigating to the site)

In determining which stories to build first, create a Value Map

1. High Value/Low Effort
2. High Value/High Effort
3. Low Value/Low Effort
4. Low Value/High Effort

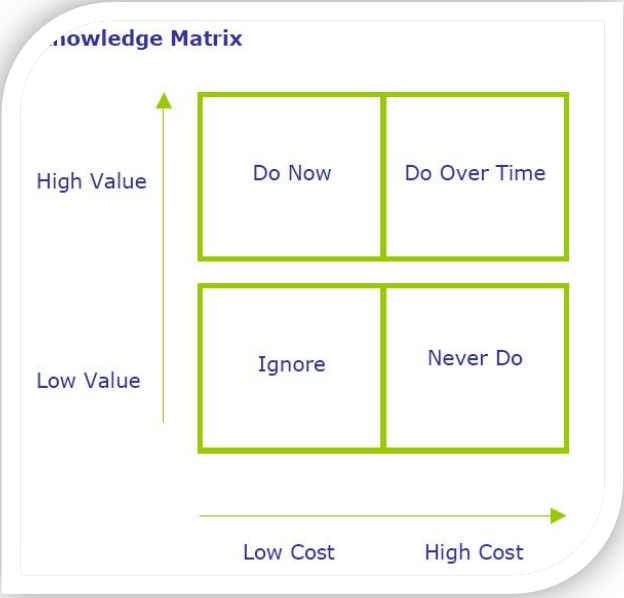

Figure 2

To help further, think -

- **MoSCoW**
  - **M**ust have this, **S**hould have, **C**ould be nice to have , **W**on't have this – maybe later
- NPV – Estimate feature value of the feature based on revenue, cost and time

Product Backlog

Product Owner

The Product Owner is the sole person responsible for managing the Product Backlog and the user stories it contains. To do so the PO must:

1. Clearly express backlog items
2. Ensure they are groomed and ready before development (in conjunction with the Scrum Master)
3. Order and prioritize the items to best achieve goals and mission
4. Ensure that the backlog is visible, transparent, and clear to all
5. Ensure that the development team understands the items in the backlog to the level needed
6. PO must be ready to present the items that are 'ready ready' at the sprint planning session and clarify the stories for all of the developers. In addition, the PO must be available for follow up questions and clarification that cannot be read from the enabling specification (see below).
7. 'Ready Ready' is defined as fully groomed and thought out stories, some documentation and review by the development staff has already begun
8. Stories will be further split down into tasks during sprint planning and technical breakout sessions, hours assigned and points estimated

Readiness

As User Stories move to the top of the backlog and are up next for development, they must go through a progression in order to get ready.

1. Level 1 Stories
   a. All inputs and ideas accepted, Product Owner determines that the story matches the product goals and puts them in the backlog
2. Level 2 Stories
   a. Story is decomposed and shelled out (Analyst, P/O, Developer)
3. Level 3 Stories
   a. User Story details take shape
   b. Acceptance criteria defined
   c. UI Pre-work (Wireframes, visual, content prototypes)
   d. Promotion of story with other stake holders to get feedback and buyin if necessary
4. Level 4 Stories
   a. Ready for Sprint
   b. Candidates for release planning and sprint planning

c. Minimal refinement needed or core user experience, well groomed stories = less sprint planning

5. User Stories at the Level 4 stage should follow the INVEST principles:
    a. Immediately Actionable
    b. Negotiable (take talk about it and trade off functionality)
    c. Valuable
    d. Estimable
    e. Sized to Fit (for the sprint)
    f. Testable (if you can't test it, more work needs to be done)

How to break down Epics/User Stories

User stories need to be broken down when they cannot be designed/developed & tested within one iteration. The intent of a user story is that it is fully completed within one sprint, i.e. developed and tested (some design may happen before the sprint). If stories are going over the length of one sprint, they need to be further broken down.

1. One reason to split a user story is if it can fit into an iteration, but will not be completed in that iteration because there is not enough time left. For example, We have assigned 25 story points in one sprint and typically complete 30 points. We have an 8 point user story and want to get started on it, so we break that story in half and assign half to the current sprint and half the next sprint if applicable.
2. When a better estimate is needed – When a user story is too big it is an epic and is hard to estimate accurately (at the sprint iteration level). If this is the case, then you need to break down the epic is better *estimable* parts.
3. Splitting the story by data elements
    a. Stories grouped by logical data mapping may be broken up and grouped, for example Epics that contain stories which are grouped by the following data points may be broken down:
        i. User data
        ii. Financial data
        iii. Scholarship data
        iv. Etc
4. Splitting the story on operational boundaries [if it is too large]

    E.G. "As a user I need to be able to search for scholarships and save my results"

This may cross many different operational boundaries and include the following elements:

    a. A Form to input criteria – May be too complex if the form involves customer interaction, drag/drop

    b. A grid to display results - "Show more" functionality, additional filtering, etc.
    c. Saving results into the users' profile
5. Cross Cutting Features
    a. There many be many cross-cutting features of the application and they may be broken down or culled out of user stories
        i. **For example:** If we need to send an email to a member at different times during their user experience, then this might be culled out into its own user story
            1. Some examples of cross cutting functionality may be:
                a. Ability to email members,
                b. Track correspondence
                c. Track user traffic across the site etc
6. Stories of mixed priority

Lastly, Stories should not be split down into tasks at this level

For example – don't break the story down into technical requirements, i.e. create SQL to fetch results, integrate with API, fetch results from API, Store results in a database, etc.

These are really stories that come from a user and do not contain a solution.

Once the User Stories have been developed, what's next?

Once the stories have been created, they will be organized, grouped and/or broken down during a planning session with development staff. A Story map will be defined which will closely following the path of the user through the system (across the stories). The stories that are touched through the initial story path are considered the minimum viable product for release.

Release planning can begin and dependencies are identified

Resources assigned for upcoming stories and number of iterations to be determined.

Enabling [Agile] Specifications

The enabling specification will help drive the acceptance criteria and will be used to:

- Check progress on the feature to date.
- Allow the development team to operate with autonomy
- Allow for less dropped requirements
- Ensure the feature works as intended

The following is an except from Jeff Sutherland, once of the founders of Scrum:

"A user story needs to be a mini-enabling specification for agile teams to operate at peak performance. If it is not, there will be the need for continued dialogue with the Product Owner during the sprint to figure out what the story means. This will reduce story process efficiency and cripple velocity.

A user story contains a template, notes, acceptance tests, and implies a conversation with the Product Owner. So the conversation may be part of the enabling specification if the conversation is clear before the beginning of a sprint. This can be on a notecard for a simple application and will be on the order of no more than 3-5 pages even for a complicated mission-critical application[1]"

[1] *Jeff Sutherland "http://scrum.jeffsutherland.com/2009/11/enabling-specifications-key-to-building.html"