

Managing Requirements through User Stories

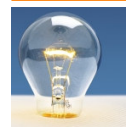


Slide 1
11 March 2009

Keith Braithwaite, Richard
Emerson, Immo Hinkele
© Zühlke 2009

Agenda

- Requirements – Purpose and Approach
- User Story format
- Managing User Stories
- Working out when a User Story is done
- Review



Managing Requirements
through User Stories
Slide 2
11 March 2009

Keith Braithwaite, Richard
Emerson, Immo Hinkele
© Zühlke 2009

What are Requirements For



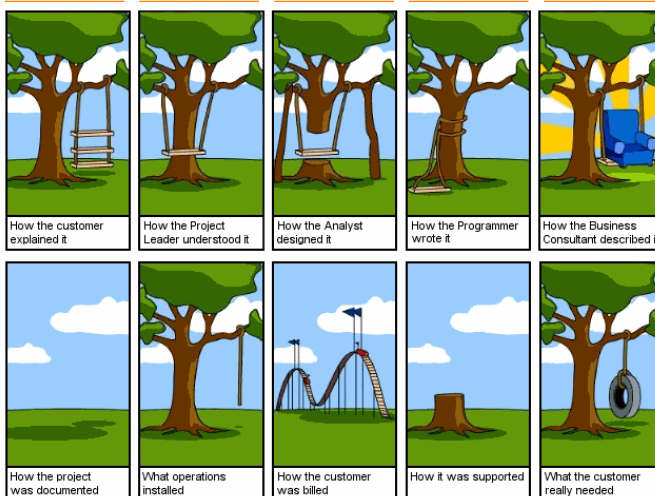
- One of the most significant causes of project failure in study after study (e.g. Standish Group) has been “the lack of clear requirements”
- Requirements specify what the system should be, how it should behave, what it should look like and criteria such as its availability, reliability, performance and security – but not how to meet these demands
- Every requirement should be matched by one or more acceptance tests that prove compliance
- Theoretically, the system can be demonstrated to be “finished” when it passes every acceptance test
- Writing unambiguous and complete requirements is very hard or impossible (think of the Highway Code)



Managing Requirements through User Stories
Slide 3
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinke
© Zühlke 2009

Problems with Requirements



Managing Requirements through User Stories
Slide 4
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinke
© Zühlke 2009

credited to Alex Gorbachev on codinghorror.com

Another Approach - Exemplars



- Human beings instinctively group things into categories based on some representative examples
- Even without explicit recognition rules you can easily categorise the following:



- A *user story* is a way to capture desired behaviour of a system by giving a “typical” instance as a narrative



Managing Requirements through User Stories
Slide 5
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

Format for Capturing a User Story



RAG – Role, Action, Goal

Advertise Availability

As a hotel manager,
I wish to *advertise my hotel* as prominently as possible
so that I can *maximise occupancy*

Search for Hotels

As a traveller,
I wish to *search for hotels* close enough to my destination with vacancies on my dates of travel
so that *I can choose* the one that best suits my needs



Managing Requirements through User Stories
Slide 6
11 March 2009

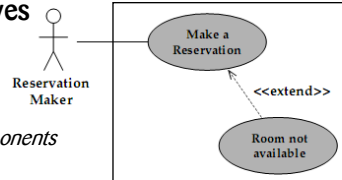
Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

Relationship of User Stories to Use Cases



Use Cases are heavily emphasised in the Rational Unified Process (RUP) and its derivatives

Simple use-case diagram from *UML Components* by John Cheesman & John Daniels



- Presumes you know in advance where the “System Boundary” lies
- Actors can be other systems, rather than real users
- Fully dressed use cases incorporate a complete script to handle every eventuality
- User stories are lightweight – one use case can equate to many user stories (but also the inverse)



Managing Requirements through User Stories
Slide 7
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

Evaluating the Story Quality



The INVEST test was devised by Bill Wake:

- **Independent:** can the user story be built without requiring other stories before we can see and test functionality?
- **Negotiable:** can specific details of the story be resolved through conversation so we can maximize benefit while minimizing development costs?
- **Valuable:** does the story add value to the software for either or both user and business?
- **Estimable:** do we know enough about the story to estimate the time to construct the software?
- **Small:** is the story as small as it can be but still valuable?
- **Testable:** can others easily verify that the story is complete?

Every story should have a meaningful title, so that you can refer to it in discussions.



Managing Requirements through User Stories
Slide 8
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

Fleshing out the Story Card



Annotations to back & front of card:

- Priority
- Status
- Estimates
- Etc.

A story goes through a simple elaboration cycle:

- **Card:** write the initial story text on the card
- **Conversation:** discuss the details with developers, analysts, and testers
- **Confirmation:** record the acceptance tests that will allow us to know when the story is complete



Managing Requirements through User Stories
Slide 9
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

When a Story is Too Big



While prioritising and estimating a story, it may turn out to be at too high a level – it needs breaking down into shorter stories to fit into an iteration.

Such stories are known as *epics*.



Managing Requirements through User Stories
Slide 10
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

Exercise I



Critique user stories

- Look at the stories we presented earlier
- Are they really epics?
- What else could be improved?

Write more user stories

- One or two stories from a customer's perspective
- One or two stories from a hotel's perspective
- One or two stories from the business perspective
- What about generating advertising revenue?

Remember RAG – role, action, goal
(as a <<R>> I want to <<A>> so that <<G>>)



Managing Requirements
through User Stories
Slide 11
11 March 2009

Keith Braithwaite, Richard
Emerson, Immo Hinkele
© Zühlke 2009

When is a Story Done?



Acceptance Criteria

- A way of judging when a requirement has been met
- Each user story gives rise to several scenarios
- Example: Search for Hotels
 - Scenario 1: no hotels found
 - Scenario 2: one hotel found
 - Scenario 3: one pageful of hotels found
 - Scenario 4: multiple pagefuls of hotels found
 - Scenario 5: destination not recognised
- Each scenario must be documented so that a tester can verify that the system handles it correctly
- The acceptance criteria must be written by the time a story is accepted into a sprint for development



Managing Requirements
through User Stories
Slide 12
11 March 2009

Keith Braithwaite, Richard
Emerson, Immo Hinkele
© Zühlke 2009

Acceptance Criteria – Checked Examples



Brian Marick defined checked examples as “the use of tests and examples as tools for thinking about a piece of what the product owner wants with results that help the programmers.” (www.exampler.com)

There are two main forms:

- Table format
- Given / When / Then format

These can be combined.

Advantages of Checked Examples include:

- Intuitively easier to understand (see examples below)
- Acceptance tests can be derived from them directly



Managing Requirements through User Stories
Slide 13
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinke
© Zühlke 2009

Tabular Acceptance Criteria: Example



Basic Employee Compensation

For each week, hourly employees are paid a standard wage per hour for the first 40 hours worked, 1.5 times their wage for each hour after the first 40 hours, and 2 times their wage for each hour worked on Sundays and holidays.

Here are some typical examples of this:

StandardHours	HolidayHours	Wage	Pay()
40	0	20	\$800
45	0	20	\$950
48	8	20	\$1360 <i>expected</i> <i>\$1040 actual</i>

From <http://fit.c2.com/>



Managing Requirements through User Stories
Slide 14
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinke
© Zühlke 2009

Narrative Acceptance Criteria: Example



Search for Hotels – Scenario 1

- Given
 - The only hotel in our database within 3km of the centre of OneHorseTown is The Buckaroo
 - The hotel has at least one vacant room on 18th April
- When
 - User searches for hotels with parameters:
 - Date of arrival = 18th April
 - Destination city = OneHorseTown
 - Acceptable distance = 3km
- Then
 - Result set has length 1
 - Result set contains The Buckaroo



Managing Requirements
through User Stories
Slide 15
11 March 2009

Keith Braithwaite, Richard
Emerson, Immo Hinke
© Zühlke 2009

Exercise II



Write acceptance tests for earlier stories

- Explore different ways in which the user story can succeed
- Explore different ways in which the user story can go wrong (other than internal system failure)
- Each test should have a meaningful name, e.g.
`hotelSearchShouldReturnTopTenResults`

Format: Given <<initial conditions>> When <<event occurs>> Then <<expectations and postconditions>>



Managing Requirements
through User Stories
Slide 16
11 March 2009

Keith Braithwaite, Richard
Emerson, Immo Hinke
© Zühlke 2009

Working with User Stories: Prioritisation



Each user story has a perceived value to the business

- Business value is extremely hard to quantify
- Value may depend to some extent on other stories
- May also depend on time to deliver the feature
- Simpler prioritisation schemes:
 - Relative ranking – shuffle the cards
 - Priority level – low, medium, high, essential
 - Combination: rank within each priority level
- Priority is used to order stories within the product backlog as well as the sprint backlog
 - Aim is to include some lower-priority items within each sprint
 - These can be deferred without much impact if time runs out



Managing Requirements through User Stories
Slide 17
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

User Story Mapping



By arranging activity and task-centric story cards spatially, we can tell bigger stories

Tell a big story (epic) by starting with the activities the product will be used for



Arrange activities left to right in the order you'd explain them to someone when asked the question: "What do people do with this system?"



Managing Requirements through User Stories
Slide 18
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

Add Task-Centric Stories to Refine Activities



Add task-centric stories in under each activity in chronological order left to right.



If you were to explain to someone what a person typically does in this activity, arrange tasks in the order you'd tell the story



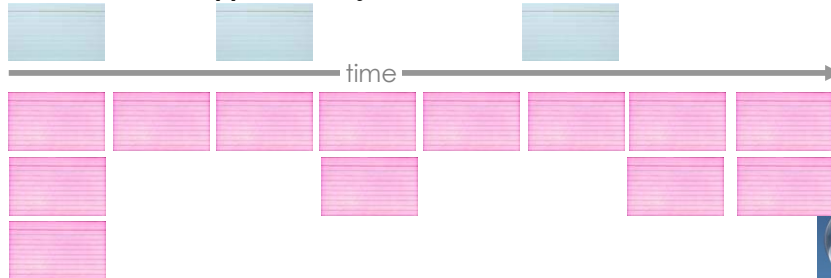
Managing Requirements through User Stories
Slide 19
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

Capture Parallelism and Alternatives



Overlap user tasks vertically if a user may do one of several tasks at approximately the same time.



If in telling the story I say the systems' user typically "does this or this or this, and then does that," "or's" signal a stacking vertically, "and then's" signal stepping horizontally.



Managing Requirements through User Stories
Slide 20
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

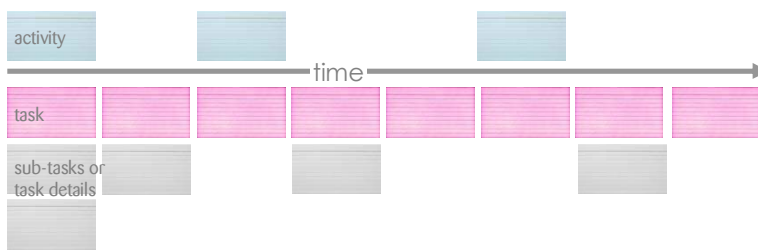
Completing the Conversation



As details emerge in conversation, trap them under their associated task cards

Record details so they're not lost, and so those who you're working with know that you're listening

- Consider tucking them under tasks cards to “hide them” from the discussion



Managing Requirements through User Stories
Slide 21
11 March 2009

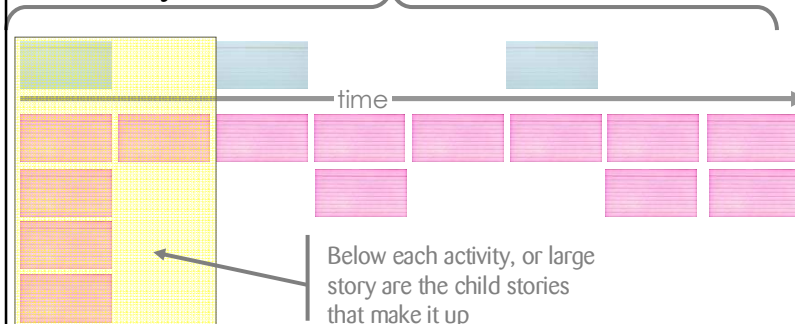
Keith Braithwaite, Richard Emerson, Immo Hinke
© Zühlke 2009

Reading the Story Map



The map of stories shows both story decomposition and a typical flow of use across the entire system

Simply reading the activities, or large stories, across the top of the system helps us understand end-to-end use of the system



Managing Requirements through User Stories
Slide 22
11 March 2009

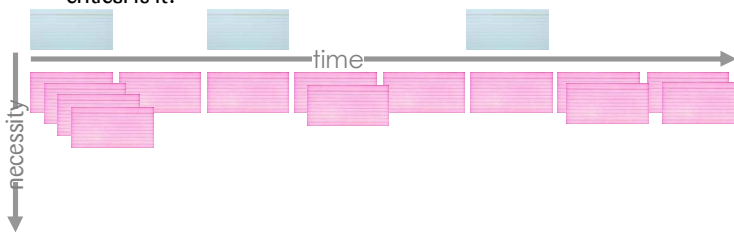
Keith Braithwaite, Richard Emerson, Immo Hinke
© Zühlke 2009

Generating the Release Plan



By arranging activity and task-centric story cards spatially, we can tell bigger stories

- Add a vertical axis to indicate necessity
- Move tasks up and down this axis to indicate how necessary they are to the activity
 - For a user to successfully engage in this activity, is it necessary they perform this task? If it's not absolutely necessary, how critical is it?



Managing Requirements through User Stories
Slide 23
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

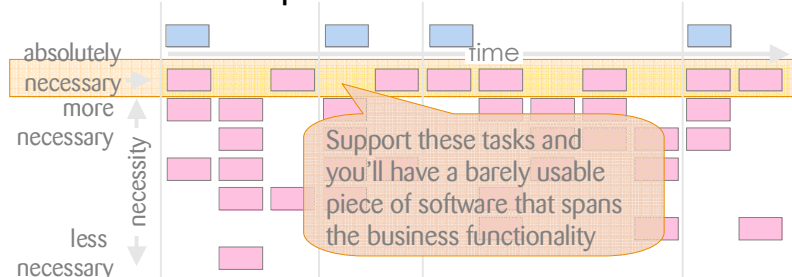
Generating the Release Plan



The top row of our story map identifies the smallest number of tasks the system must support to be usable

The Story Map we've built identifies the major activities and tasks that span the business functionality

A successful software release must support all necessary activities in the business process



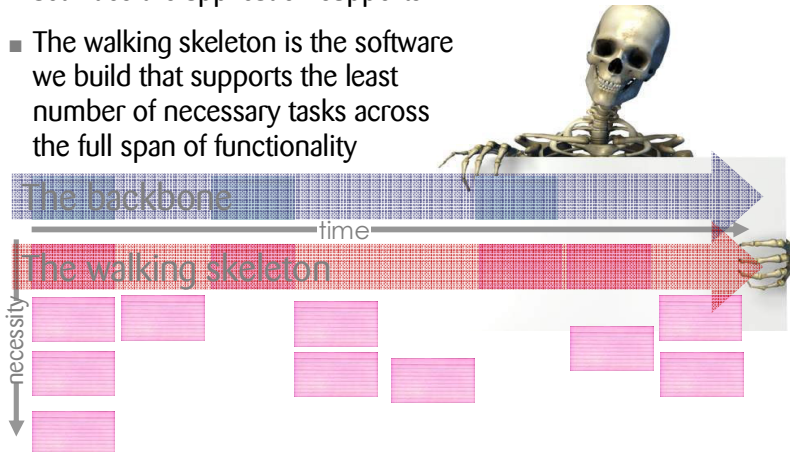
Managing Requirements through User Stories
Slide 24
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

Two important anatomical features



- The backbone of the application is the list of essential activities the application supports
- The walking skeleton is the software we build that supports the least number of necessary tasks across the full span of functionality



Managing Requirements
through User Stories
Slide 25
11 March 2009

Keith Braithwaite, Richard
Emerson, Immo Hinke
© Zühlke 2009

Exercise III



Develop stories to describe the requirements of a Veterinary practice administration system

- Animals are brought in without appointment when they are ill or injured
- Animals are registered
- Animals are examined by a veterinarian
- Treatments may be prescribed
- Appointments for repeat visits may be arranged
- Payment may be taken
- Etc.

Arrange the stories on a story map and refine them

Identify the minimal first release of the system



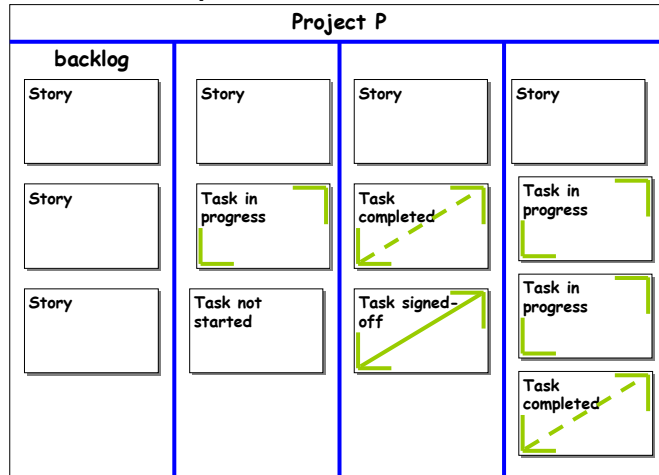
Managing Requirements
through User Stories
Slide 26
11 March 2009

Keith Braithwaite, Richard
Emerson, Immo Hinke
© Zühlke 2009

Working with User Stories: Tracking



The Wall – one possible version



Managing Requirements through User Stories
Slide 27
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

Conclusion



User stories are a useful and intuitive way to capture requirements

User stories feed directly into acceptance testing, prioritisation and planning

- Jeff Patton's "story mapping" technique provides guidance on prioritisation for coherent releases

Suggested story format: RAG (role, action, goal)

Suggested acceptance criteria format: GWT (given, when, then)



Managing Requirements through User Stories
Slide 28
11 March 2009

Keith Braithwaite, Richard Emerson, Immo Hinkele
© Zühlke 2009

Questions



**Managing Requirements
through User Stories**
Slide 29
11 March 2009

Keith Brathwaite, Richard
Emerson, Inero Hinkel
© Zühlke 2009