Agile requirements through user stories and scenarios
Agenda
• Key principles of agile requirements
• User stories
• INVEST
• Prioritizing stories
• User story mapping
• Challenges
• Conclusion
Key Principles for Agile Requirements

• Active user involvement is imperative
• Agile teams must be empowered to make decisions
• Requirements emerge and evolve as software is developed
• Agile requirements are ‘barely sufficient’
• Requirements are developed in small, bite-sized pieces
• Enough’s enough – apply the 80/20 rule
• Cooperation, collaboration and communication between all team members is essential
Requirements are a Communication Problem

• **Written requirements**
  – can be well thought through, reviewed and edited
  – provide a permanent record
  – are more easily shared with groups of people
  – time consuming to produce
  – may be less relevant or superseded over time
  – can be easily misinterpreted

• **Verbal requirements**
  – instantaneous feedback and clarification
  – information-packed exchange
  – easier to clarify and gain common understanding
  – more easily adapted to any new information known at the time
  – can spark ideas about problems and opportunities
User Stories

seek to combine the strengths of written and verbal communication, where possible supported by a picture.

* Kent Beck coined the term user stories in Extreme Programming Explained 1st Edition, 1999
What is a User Story?

• A concise, written description of a piece of functionality that will be valuable to a user (or owner) of the software.

• Stories are:
  – User’s needs
  – Product description
  – Planning item
  – Token for a conversation
  – Mechanism for deferring conversation
User Story Cards have 3 parts

1. **Card** - A written description of the user story for planning purposes and as a reminder

2. **Conversation** - A section for capturing further information about the user story and details of any conversations

3. **Confirmation** - A section to convey what tests will be carried out to confirm the user story is complete and working as expected
User Story Description

- As a [user role] I want to [goal] so I can [reason]
- As a [type of user] I want to [perform some task] so that I can [reach some goal]

For example:

• As a registered user I want to log in so I can access subscriber-only content
User Story Description

• **Who** (user role)

• **What** (goal)

• **Why** (reason)
  – gives clarity as to why a feature is useful
  – can influence how a feature should function
  – can give you ideas for other useful features that support the user's goals
User Story Description

Steps:

• Start with a title.

• Add a concise description often using this useful templates.

• Add other relevant notes, specifications, or sketches

• Before building software write acceptance criteria (how do we know when we’re done?)
User Story Example: Front of Card

As a [registered user], I want to [log in], so I can [access subscriber content].

User Login
Username: [input field]
Password: [input field]

Remember me

Login

User’s email address. Validate format.
Authenticate against SRS using new web service.
Go to forgotten password page.

Display message here if not successful. (see confirmation scenarios over)

Further information is attached to this story on VSTS Product Backlog.
User Story Example: Back of Card

Confirmation

1. Success – valid user logged in and referred to home page.
   a. ‘Remember me’ ticked – store cookie / automatic login next time.
   b. ‘Remember me’ not ticked – force login next time.

2. Failure – display message:
   a) “Email address in wrong format”
   b) “Unrecognised user name, please try again”
   c) “Incorrect password, please try again”
   d) “Service unavailable, please try again”
   e) Account has expired – refer to account renewal sales page.
How detailed should a User Story be?

Detailed enough for the team to start work from, and further details to be established and clarified at the time of development.
INVEST in Good User Stories

- **Independent** – User Stories should be as independent as possible.
- **Negotiable** – User Stories are not a contract. They are not detailed specifications. They are reminders of features for the team to discuss and collaborate to clarify the details near the time of development.
- **Valuable** – User Stories should be valuable to the user (or owner) of the solution. They should be written in user language. They should be features, not tasks.
- **Estimatable** – User Stories need to be possible to estimate. They need to provide enough information to estimate, without being too detailed.
- **Small** – User Stories should be small. Not too small. But not too big.
- **Testable** – User Stories need to be worded in a way that is testable, i.e. not too subjective and to provide clear details of how the User Story will be tested.
Prioritizing stories into a backlog

- Agile customers or product owner prioritize stories into a backlog

- A collection of stories for a software product is referred to as the product backlog

- The backlog is prioritized such that the most valuable items are highest
User Story Mapping

• User Story Mapping is an approach to Organizing and Prioritizing user stories

• Unlike typical user story backlogs, Story Maps:
  – make visible the workflow or value chain
  – show the relationships of larger stories to their child stories
  – help confirm the completeness of your backlog
  – provide a useful context for prioritization
  – Plan releases in complete and valuable slices of functionality.
User Story Mapping

• Spatial arrangement:
  – By arranging activity and task-centric story cards spatially, we can tell bigger stories
  – 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?”
User Story Mapping

• 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.
User Story Mapping

• The map shows decomposition and typical flow across the entire system.

Below each activity, or large story are the child stories that make it up

• Reading the activities across the top of the system helps us understand end-to-end use of the system.
User Story Mapping - prioritizing

- Prioritizing based on product goal
  - Product goals describe what outcome or benefit received by the organization after the product is in use
  - Use product goals to identify candidate incremental releases, where each release delivers benefit
User Story Mapping - prioritizing

– Create horizontal swim-lanes to group features into releases

– Arrange features vertically by necessity from the user’s perspective

– Split tasks into parts that can be deferred till later releases

– Use the product goals from your handouts to identify slices that incrementally realize product goals
Agile- Challenges

• Active user involvement can be very demanding on the user representative's time and require a big commitment for the duration of the project.

• Iterations can be a substantial overhead if the deployment cost are large

• Agile requirements are barely sufficient:
  – This can mean less information available to new starters in the team about features and how they should work.

• Usually not suitable for projects with high developer turnover with long-term maintenance contract

• Arguably not suitable for safety critical systems.
User Stories Summary

- User Stories combine written and verbal communications, supported with a picture where possible.
- User Stories should describe features that are of value to the user, written in a user’s language.
- User Stories detail just enough information and no more.
- Details are deferred and captured through collaboration just in time for development.
- Test cases should be written before development, when the User Story is written.
- User Stories should be Independent, Negotiable, Valuable, Estimatable, Small and Testable.