Introducing User Stories

User stories...

- are not just a sentence ("*As a <type of user> I want...*")!
- describe NEW or CHANGED functionality that is valuable to the stakeholders
- provide (at least on small piece of) visible functionality to the end user
- are sliced small enough to fit into 2-5 days max (but still with visible functionality!)
- specify the behaviour of a system not technical implementation
- consist of 3 aspects:
  - written description of the story (used for planning and as reminder) → "*As a <type of user> I want <some goal> so that <some reason>*"
  - conversations about the story → details of the story
  - tests to convey and document details (used to determine when a story is accomplished)

beware of:

- when writing user stories focus more on discussing acceptance tests for the story than on title or description-template ("*As a <user> I want...*")
- acceptance tests should be well understood by all team members. common techniques for formulation: "*Test that <some expected behaviour>*", Given/When/then, specification by example
- scrum meetings and dev items as "tasks" are no user story and should be tracked separately from the Product Backlog (as they pollute velocity otherwise)
- in case of non co-located and difficultly reachable PO regular Backlog grooming meetings help to ensure adaption to PO, recommended 1-2h per week
- changing the mindset of the team – learn to communication and optimize

useful strategies for splitting user stories

http://www.christiaanverwijs.nl/post/2013/05/17/8-useful-strategies-for-splitting-large-user-stories-(and-a-cheatsheet).aspx

splitting improves understanding, increases accuracy of estimation and facilitates prioritization for the PO

horizontal splitting

split by:

- by kind of work that needs to be done or
- the application layer

problem! limits the ability of the team to deliver business value because

- no business value delivered by individual items
- bottlenecks increased (silo thinking)
- prioritization not possible, fosters misunderstanding

vertical splitting
benefit: smaller item still deliver business value

split by:

- workflow steps
- business rules
- happy/ unhappy flow
- input options/platform
- datatypes or parameters
- operations
- test scenarios / test cases
- roles
- acceptance criteria
- parts hard / easy to implement
- external dependencies
- usability requirements
- ...