

Requirements Modeling - User Stories

CSC 326

© NC State Software Engineering Faculty

1

Three Forms Of Requirements

Traditional/Plan-Driven.
Use Case.
User Story.

© NC State Software Engineering Faculty

2

Types of Requirements Statements

- Traditional
  - "The system shall…"
  - Very formal
- Use case based
  - Description of system from user perspective
- User story – includes acceptance test to supply the details
  - Agile requirements

©NCSU NC State Software Engineering Faculty

3

Spectrum of Appropriate Requirements Processes and Tools

Requirements Process

more formal — *less formal*



Traditional Requirements Use Cases

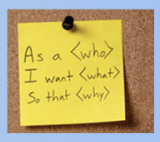
- Stable
- Larger
- Safety Critical



Examples:
- Surgery Robots
- Health Care
- Nuclear Power Plant

- Plan-driven, well-documented

User Stories



- Changing
- Smaller
- Fast-to-market
  - Examples:
    - Games
    - Most Apps
    - Social media



TikTok

- Agile, Scrum, XP

4

Requirements Process → *less formal*

User Stories

© NC State Software Engineering Faculty

5

User Story Structure Template

•Title

–Used for a reference, 3-10 words

–Long enough to differentiate it from another story

–Short enough to fit on a 3”x5” card in marker

•Description

–Main body with template

–Includes user characteristic and rationale for functionality

•Acceptance Criteria

–When has the story met the goal of the user

6

User Story Description

- Template: Who What Why
  - "As a [***user role***], I want to [***goal***] so I can [***rationale***]"
  - If it's too long
    - It might be multiple stories
    - It may have too much detail

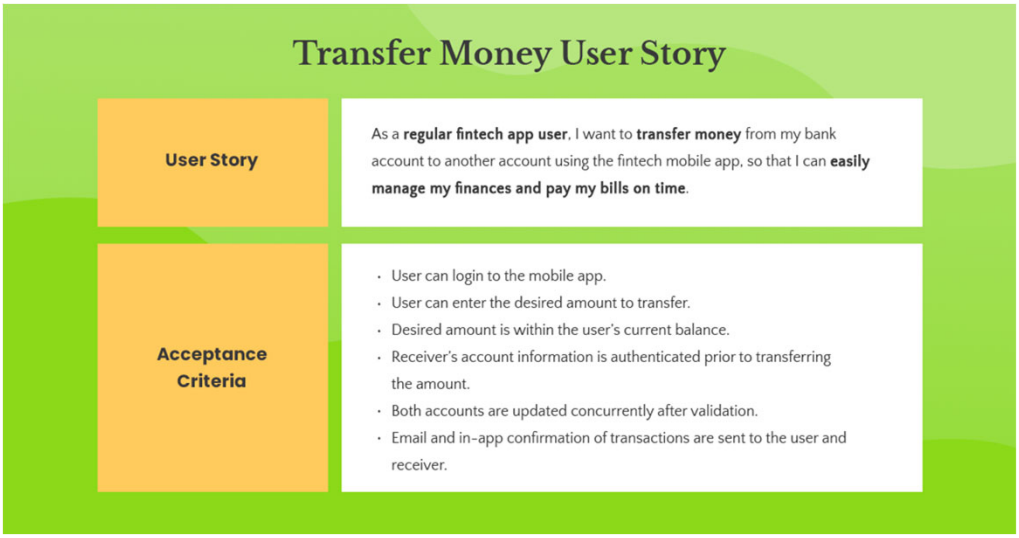
Acceptance Criteria

- Describes "how will I know when I'm done with the story?"
- Customer helps to determine criteria
  - May change as the functionality for the story matures

7

7

Sample User Story



8

8

Sample User Story

•**Title:** Search for coffee recipe by ingredient

•**Description:**

–As a coffee machine user, I need the ability to search for coffee recipes by ingredient so that I can more efficiently find seasonal drinks.

•**Acceptance criteria:**

–The coffee recipe search mechanism has the ability to enter an ingredient.

–The search will have a list of recipes indexed by ingredient

–Searching for recipes via ingredient will return a list of matching recipes or a message indicating that there are no results.

– The results will be ordered based on the proportion of the recipe that includes the ingredient. For example, searching by "oat milk" would rank "oat milk latte" above "oat milk cappuccino" because the latte includes more oat milk per ounce.

–If there are more results than can fit on one page, the system will provide the capability to view the list in pages or sections.

9

9

Structure comparison between **US** and **UC**

User Story	Use Case
•Title	UC-X Use Case Name
	X.1 Preconditions
•Description	X.2 Main Flow
	X.3 Subflows
•Acceptance Criteria	X.4 Alternative Flows

10

10

Structure comparison between **US** and **UC**

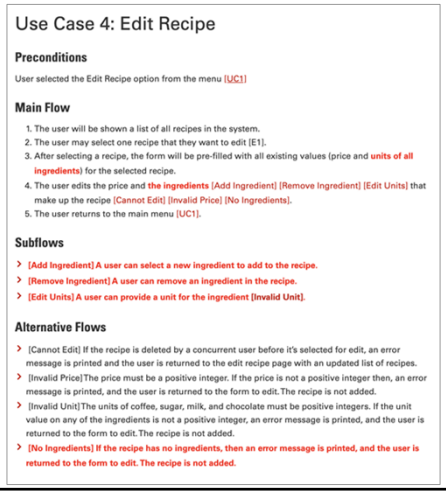
User Story

Create Recipes

As a ***customer***, I want to ***add new ingredients to create more custom drink recipes***, so I ***can enjoy a yummy beverage during work***.

Acceptance: Abi adds apple cider as an ingredient in the CoffeeMaker system.

Use Case



Use Case 4: Edit Recipe

Preconditions

User selected the Edit Recipe option from the menu [UC1]

Main Flow

1. The user will be shown a list of all recipes in the system.
2. The user may select one recipe that they want to edit [E1].
3. After selecting a recipe, the form will be pre-filled with all existing values (price and units of all ingredients) for the selected recipe.
4. The user edits the price and the ingredients [Add Ingredient] [Remove Ingredient] [Edit Units] that make up the recipe [Cannot Edit] [Invalid Price] [No Ingredients].
5. The user returns to the main menu [UC1].

Subflows

- [Add Ingredient] A user can select a new ingredient to add to the recipe.
- [Remove Ingredient] A user can remove an ingredient in the recipe.
- [Edit Units] A user can provide a unit for the ingredient [Invalid Unit].

Alternative Flows

- [Cannot Edit] If the recipe is deleted by a concurrent user before it's selected for edit, an error message is printed and the user is returned to the edit recipe page with an updated list of recipes.
- [Invalid Price] The price must be a positive integer. If the price is not a positive integer then, an error message is printed, and the user is returned to the form to edit. The recipe is not added.
- [Invalid Unit] The units of coffee, sugar, milk, and chocolate must be positive integers. If the unit value on any of the ingredients is not a positive integer, an error message is printed, and the user is returned to the form to edit. The recipe is not added.
- [No Ingredients] If the recipe has no ingredients, then an error message is printed, and the user is returned to the form to edit. The recipe is not added.

11

Assumption:
In the beginning, neither the customer nor the developer has full knowledge of requirements.

© NC State Software Engineering Faculty

12

12

Process:
Frequent, personal interactions with customers and/or stakeholders

© NC State Software Engineering Faculty

13

Goal:
Frequent Delivery of software

© NC State Software Engineering Faculty

14

Result:
Faster delivery of software that meets customer needs

© NC State Software Engineering Faculty

15

Drivers of Requirements

- Domain Expert
- Study the Market
- Talk to Customers
- Understand the Business
- Run Focus Groups
- Competitive Analysis
- Go to Trade Shows



- …. Also take suggestions from the development team, technical support, professional services, etc. and understand feasibility/limitations

© NC State Software Engineering Faculty

16

In other words: Agile processes are highly collaborative

© NC State Software Engineering Faculty

17

Features ≡ User Stories

© NC State Software Engineering Faculty

18

Writing a User Story

1 As an [X], I want to [Y] so I can [Z].

X = user role (who)
Y = desired functionality (what)
Z = benefit (why)

micro-story

2 Acceptance Criterion:

- ❑ A
- ❑ B
- ❑ C

} checklist ✓

19

They put it on a coffee mug...

1. micro-story

2. checklist ✓



20

Exercise 14.06: User Story

Use Case 4: Edit Recipe

Preconditions

User selected the Edit Recipe option from the menu [UC1]

Main Flow

1. The user will be shown a list of all recipes in the system.
2. The user may select one recipe that they want to edit [E1].
3. After selecting a recipe, the form will be pre-filled with all existing values (price and **units of all ingredients**) for the selected recipe.
4. The user edits the price and **the ingredients** [Add Ingredient] [Remove Ingredient] [Edit Units] that make up the recipe [Cannot Edit] [Invalid Price] [No Ingredients].
5. The user returns to the main menu [UC1].

21

21

I am your customer







Coffee at WakeUp
As a user, when I wake up in the morning, I want my coffee to be ready.

Acceptance: alarm goes off, coffee is 190F

Clean the coffee machine
As a machine, I want to be cleaned within 30 minutes of brewing.

Acceptance: after brewing, ping the user to clean me until I've been cleaned

Coffee after Lunch
As a user, after I eat lunch, I want my coffee to be ready.

Acceptance: after lunch appt on calendar, coffee is 190F

Coffee after Morning Workout
As a user, after I work out in the morning, I want my coffee to be ready.

Acceptance: after my fitbit is done charging, coffee begins

© NC State Software Engineering Faculty

22

22

What makes a good story?

23

Attributes of a user story (INVEST)

- Independent (of all others)
- Negotiable (not a specific contract for features)
- Valuable to users
- Estimate-able (to a good approximation)
- Small* (so as to fit within an iteration)
- Testable (in principle, even if there isn't a test for it yet)

* Only applies to user stories; epics are big

© NC State Software Engineering Faculty

24

Attributes of a user story (INVEST)

- Independent
- Negotiable
- Valuable to users
- Estimate-able
- Small*
- Testable

* Only applies to user stories

Avoid dependencies on other stories

Write stories to establish the system foundation

Combine stories for a single iteration (where appropriate)

Ref: Craig Brown

© NC State Software Engineering Faculty

Attributes of a user story (INVEST)

- Independent
- Negotiable
- Valuable to users
- Estimate-able
- Small*
- Testable

* Only applies to user stories

Stories are not a contract

Too much written detail - it suggests that there is no more to explore

Know when you can't negotiate - some constraints are fixed

Ref: Craig Brown

© NC State Software Engineering Faculty

Attributes of a user story (INVEST)

- Independent
- Negotiable
- Valuable to users
- Estimate-able
- Small*
- Testable

* Only applies to user stories

Show what the value of the story is for the customers and other stakeholders

Ref: Craig Brown

© NC State Software Engineering Faculty

Attributes of a user story (INVEST)

- Independent
- Negotiable
- Valuable to users
- Estimate-able
- Small*
- Testable

* Only applies to user stories

Sufficient detail needs to be present to **estimate** the work effort

Stories should be small enough to estimate, (but not too small)

Ref: Craig Brown

© NC State Software Engineering Faculty

Attributes of a user story (INVEST)

- Independent
- Negotiable
- Valuable to users
- Estimate-able
- Small*
- Testable

* Only applies to user stories

Stories should be small enough to complete in a sprint (like 2 weeks)

The closer a story is to being worked on the more specific it should be

Stories can start high level (epic) but they'll need to be brokendown later

Ref: Craig Brown

© NC State Software Engineering Faculty

Attributes of a user story (INVEST)

- Independent
- Negotiable
- Valuable to users
- Estimate-able
- Small*
- Testable

* Only applies to user stories

Acceptance criteria should be apparent in the user story

Tests should be automated wherever possible

User stories should not be commenced until they have clear acceptance criteria

Ref: Craig Brown

© NC State Software Engineering Faculty

Non-functional requirements and constraints

- Create a story for these as a reminder **they may apply to all stories**. If applicable to a story, the estimate for a story must consider the non-functional requirement and/or constraint.
  - Often placed in "Iteration 0"

© NC State Software Engineering Faculty

Non functional story examples

- As a customer, I want to be able to run the product on all versions of Windows from Windows 10 on so that I don't have to upgrade all my systems.

- As the CTO, I want the system to use our existing orders database rather than create a new one so that we don't have one more database to maintain.

© NC State Software Engineering Faculty
Adapted from http://blog.mountaingoatsoftware.com/non-functional-requirements-as-user-stories

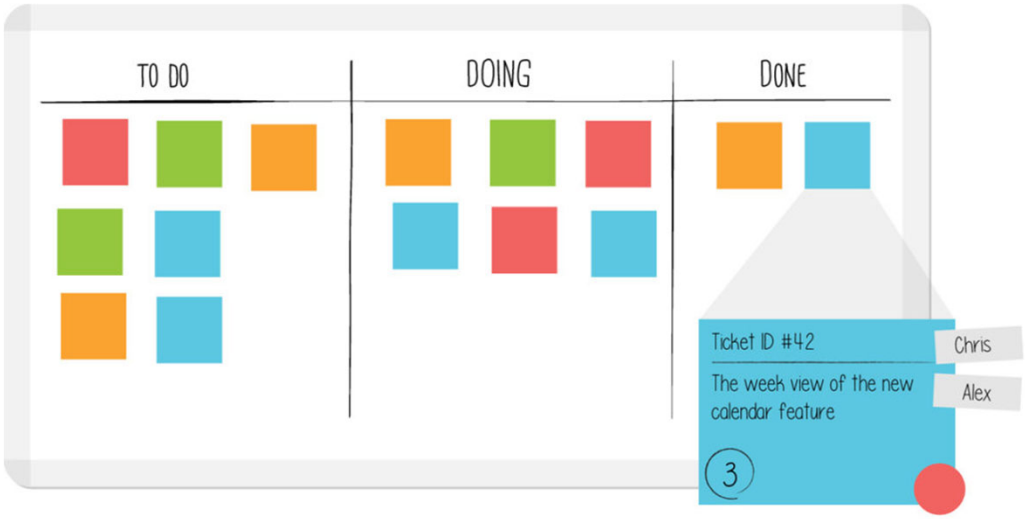
Kanban Boards (Agile Planning)



© NC State Software Engineering Faculty

33

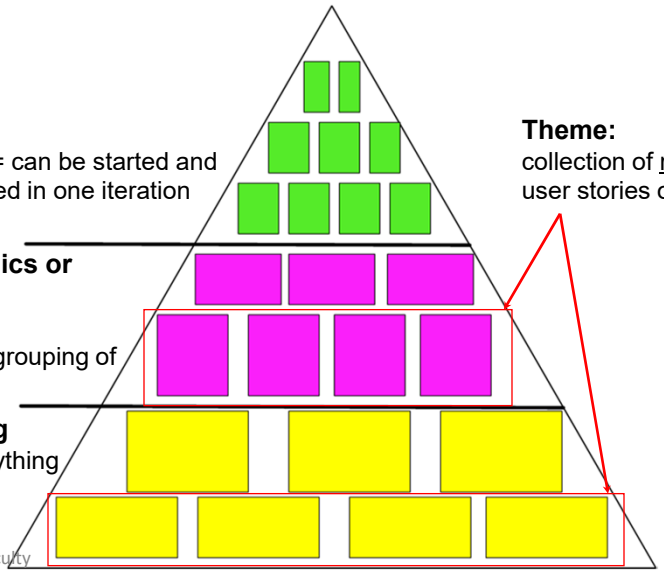
Kanban Boards (Agile Planning)



© NC State Software Engineering Faculty https://leankit.com/learn/kanban/kanban-board/

34

Agile Requirements Granularity



© NC State Software Engineering Faculty

Summary – User Stories

Requirements Process → *less formal*

- Customer plays prime role in requirements
- Epics and stories are not "fully" documented requirements, but a token for further conversation between the development team and the customer.
  - The details of epics and stories emerge on a "just-in-time" basis during iteration planning.
- Epics are too large to finish within an iteration.
- Stories can be completed in one iteration but must still be useful to a user or purchaser (i.e. are not technical tasks)

© NC State Software Engineering Faculty

Use Case vs. User Story

- Use cases written by developers
  - Fully documented, archived
  - Read by developers
  - Read by stakeholders
  - Read by customers
- User stories are written by customers (maybe with developer assistance)
  - Low ceremony
  - Read by developers
  - Read by stakeholders
  - Read by customers

© NC State Software Engineering Faculty

37

Spectrum of Appropriate Requirements Processes and Tools

more formal ← *Requirements Process* → *less formal*



Traditional Requirements Use Cases

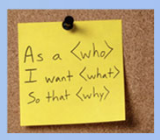
- Stable
- Larger
- Safety Critical



Examples:
- Surgery Robots
- Health Care
- Nuclear Power Plant

- Plan-driven, well-documented

User Stories



- Changing
- Smaller
- Fast-to-market
  - Examples:
    - Games
    - Most Apps
    - Social media

- Agile, Scrum, XP

38