in real life

User Stories and Use Cases

Mikael Svahnberg[1]

2017-03-23

[1] Mikael.Svahnberg@bth.se

User Stories

- User Stories are the currently preferred way in agile of writing requirements.
- Simpler structure than the “regular” requirements
- Not usually worked with in lenghty documents
  - Rather, just enough to fill a sprint
- Template: *As a <type of user> I want <some goal> so that <some reason>.*

User Stories Examples

- *As a software designer I want to document what the customer is telling me so that I can discuss my understanding of their requirements with them.*
- *As a software designer I want to document what the customer is telling me so that I can easily continue designing based on a joint understanding.*
- *As a software designer I want to be able to reuse common procedures between my use cases so that I only have to maintain them in one place.*
- *As a project manager I need to decide what to focus on building right now so that I best satisfy all customers' expectations.*
- *As a product manager I want to make sure that we are building software of high quality.*

Discuss: Good and Bad Requirements I

Users shall be able to view a personal calendar and recent notifications in the system.

> *Use Case: View Calendar and Notifications*
> *Actors: System Users*
> *Description:*
> > *A user requests to view their personal calendar.*
> > > *The system displays the users' personal calendar.*
> >
> > *A user requests to view their recent notifications.*
> > > *The system displays the users' recent notifications.*

Discuss: Good and Bad Requirements I

Users shall be able to view a personal calendar and recent notifications in the system.

> *Use Case: View Calendar and Notifications*
> *Actors: System Users*
> *Description:*
> > *A user requests to view their personal calendar.*
> > > *The system displays the users' personal calendar.*
> >
> > *A user requests to view their recent notifications.*
> > > *The system displays the users' recent notifications.*

As a regular user I want to be able to view my personal calendar and recent notifications *so that I can make sure that a new notification does not double-book me on an event*.

User Stories and UML Use Cases

- User Stories are *not* UML Use Cases.
  - . . . But you can use them as a starting point (for the Description)
  - Discuss: Why are they not the same? Why do we need both?
- User Stories have “Conditions of Satisfaction” – Acceptance criteria.
  - Where would you fit these in a Use Case?
  - *Would* you fit these in a Use Case?

Value

- Both User Stories and UML Use Cases focus on *value*.
- They describe something that adds value to the system.
  - e.g. A Business Process

UML Use Cases

- A concrete scenario
- Normally describes a process consisting of some back-and-forth interaction with the system.
- A collection of requirements, working together to solve a particular scenario.
- Focus primarily on the *positive* scenario
  - everything goes as expected.
- Describes a particular instance of the scenario
  - not all possible cases.

UML Use Cases

- Name
- Actors
- Brief Description

Use Case: Return Book
Actors: Customer, Librarian
Description: A customer returns a book. The librarian scans the barcode and the system registers the book as being back in the library.

Discuss

- Why can't the customer scan the barcode themselves?
- Why barcode?
- Why does the system need to know that the book is back in the library?
- How do we test this?
  - How do we regression test this?
- From a Requirements Engineering perspective, how can we find more information if we need it?

Expanded Use Cases

Use Case: name
Actors: primary actors
Description: brief description
Main Course of Events:

Actor	System
1. action	2. response
...	...
n-1. action	n. response

Alternative Flow of Events:

p. a description of what might happen instead, and how the system reacts to this.

Discussion: What’s in a Name

What are good names for use cases?

Example: Discussion Forum

What are the primary use cases for a discussion forum?

Planning with the help of Use Cases

- Use Cases describe high-level usage scenarios
  - features?
- Which use case should be implemented first?
- Which use cases contribute to a Minimum Viable Product?

Discussion on Use Case Ranking

Increase ranking of a use case if it

- has direct impact on architectural design
  - example: adds classes to domain layer, require persistent services
- includes risky, time-critical, complex functions
- involves new research or technology
- represents primary business processes
- directly supports revenue or decreased costs

Discuss

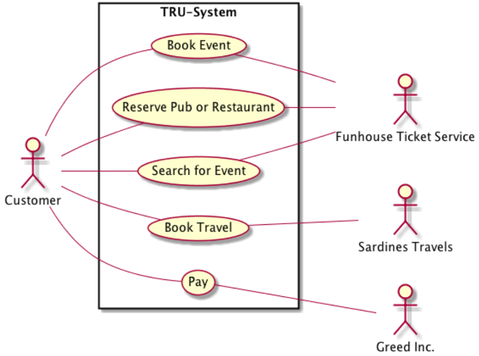
For each of these cases, why does it increase the rank of a use case?

Use Case Ranking Techniques

- Scored (Numerical Weights)
- Discrete (High, Medium, Low)
- Simple Ordering (bubble sort?)
- MoSCoW (Must have, Should have, Could have, Won’t have)
- Cumulative Voting

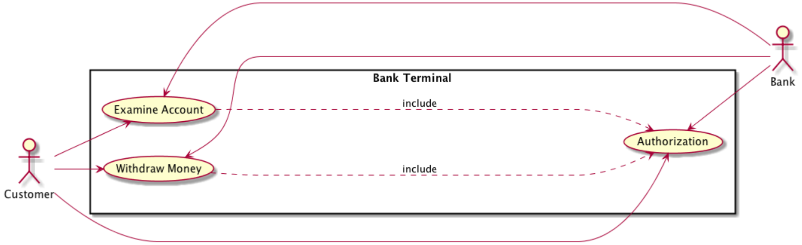
Use Case Diagrams

- Overview of Use Cases, System Boundaries, and Actors



Use Case Reuse

- Extend and Include
- Do this later! Your main job is still to understand *what* the customer wants. Relating use cases is mostly a modelling exercise.



Use Case `WithdrawMoney`

Use Case: Withdraw Money
Primary Actor: Customer
Actors: Bank

Description: A customer authenticates themselves against the bank, and selects how much money they want to withdraw and in what types of banknotes. The system updates the account (if there is enough money) and gives the money to the customer.

Main Flow of Events:

Actor	System
1. Customer *Authorises* themself to the machine	2. Initiate use case Authorisation
3. Customer *Checks* how much money is available	4. System displays the amount of money on the account.
5. Customer decides how much money they want	
6. Customer decides what banknotes they want	7. System verifies that there are sufficient funds in the account.
	8. System subtracts desired amount from account.
	9. System dispenses money (according to the preferred banknotes).
	10. System returns card.

Use Case `WithdrawMoney`

Use Case: Withdraw Money
Primary Actor: Customer
Actors: Bank

Preconditions:
- Customer is a customer at the bank
- There is money in the bank terminal

Postconditions:
- Customer walks away with cash in hand
- The customers account is accordingly updated

Description: A customer authenticates themselves against the bank, and selects how much money they want to withdraw and in what types of banknotes. The system updates the account (if there is enough money) and gives the money to the customer.

Main Flow of Events:

Actor	System
1. Customer *Authorises* themself to the machine	2. Initiate use case Authorisation
3. Customer *Checks* how much money is available	4. System displays the amount of money on the account.
5. Customer decides how much money they want	
6. Customer decides what banknotes they want	7. System verifies that there are sufficient funds in the account.
	8. System subtracts desired amount from account.
	9. System dispenses money (according to the preferred banknotes).
	10. System returns card.

Alternative Flows:
- *2. Customer fails to authorise themselves correctly. The transaction is aborted.*
- *7. There are not enough money in the account. The transaction is aborted and an error message is shown.*
- *9. The desired banknotes are not available. The system dispenses other banknotes.*
- *9. There is not enough money in the machine. The transaction is aborted and the system initiates the out of money use case.*

Extends vs Includes

- Extends $\approx$ inheritance: Your new use case is the same as the old, *except* for any modifications you introduce.
- Includes “borrows” a separate use case and inserts it into the flow of the current use case.
- In fact, you may use the *include* relationship to compose smaller scenarios into a bigger one.

Example

- Example the use case “Pay for Items” can *include* “Pay by cash”, “Pay by card”, and “Pay by cheque”.
- The use case “Pay with Gift Certificate” *extends* “Pay by cash”.

How to write Extends Use Cases

- Trigger: What starts the extension rather than the original use case?
- Extension Point: Where in the original use case do you add your extensions
- Main Course of Events: the changes to the flow.

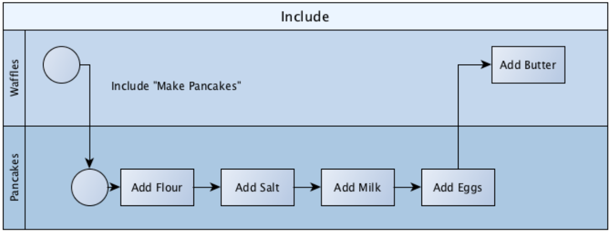
Trigger: Customer wants to pay with a gift certificate
Extension Points: Payment in the 'Process Sale' use case
Main Success Scenario:

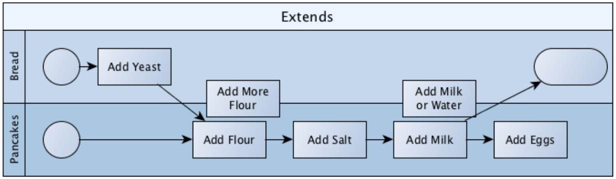
Actor	System
1. ...	2. ...

Larman and the Extend Relationship

See more in Larman, page 497.
The bottom line is that not even Larman knows what to use the extends relationship for, so he writes something fluffy in the hope that someone might glean some meaning out of it.
So in your case I think the best advice is: don't use the extends relationship.

Extends vs Includes


Include
Waffles
Pancakes
Include "Make Pancakes"
Add Butter
Add Flour
Add Salt
Add Milk
Add Eggs


Extends
Bread
Pancakes
Add Yeast
Add More Flour
Add Milk or Water
Add Flour
Add Salt
Add Milk
Add Eggs