

# Are Use Cases Beneficial for Developers Using Agile Requirements?

---

Rosalva E. Gallardo-Valencia

Vivian Olivera

Susan Elliott Sim

*University of California, Irvine*

# Research Question

---

- Can the combination of Use Cases and Agile Requirements help developers to better understand a maintenance task?
  - Agile Requirements are those used in agile software development.
    - Examples: User Stories, On-Site Customer, and Test Cases
  - Are these enough? Or do we need more?
    - Examples: Sketches of graphical user interfaces, Use Cases, Class Diagrams

# Summary

---

- Use Cases and Agile Requirements are complementary. Subjects who had access to both, spent less time understanding requirements. They also spent less time asking questions to the On-Site Customer and asked better questions.
- The requirements format didn't have a great impact on how well the subjects completed the maintenance task. It appeared that the most important factor was the analytical ability of each subject.

# Background

---

- Agile Requirements
  - User Stories: They have 3 aspects
    - Description of a feature or to-do item used for planning.
    - Conversations about the story to flesh out details.
    - Test cases that convey the details.
  - On-Site Customers:
    - Customers (or surrogates), who are co-located with the development team, so they can answer questions.
  - Test Cases
- Use Cases
  - They describe the set of interactions and events between the users or external systems (actors). They are extensively used in processes such as RUP.

# Background cont.

## ■ Agile Requirements

User Stories

As a Buyer, I can modify the quantity of each item in the cart



+

Conversations



+

Test Cases



## ■ Use Case

<b>USE CASE 5</b>	Modify Quantity	
<b>Goal in Context</b>	Modify quantity of the items already in the cart	
<b>Preconditions</b>	Buyer has pressed the "Modify Quantity" button	
<b>Success End Condition</b>	Buyer has successfully changed the quantity of the items in the cart	
<b>Failed End Condition</b>	The user could not change the quantity of the items in the cart	
<b>Primary, secondary Actors</b>	Buyer	
<b>Trigger</b>	"Modify Quantity" button is clicked	
<b>DESCRIPTION</b>	<b>Step</b>	<b>Action</b>
	1	The system shows a table with the following columns: Item Number, Description, Quantity, and Cost. The quantity column should be editable
	2	The user enters the new quantity for the items and press the "Update Cart" button
	3	The system verifies that all quantities are integers greater than zero
	4	Use "Use Case 8. View Cart"
<b>SUB-VARIATIONS</b>		<b>Branching Action</b>
	3	If the quantity is not greater than zero or is not an integer, the system should show an error message "The quantity should be an integer greater than zero."

# Method

---

- We organized our nine subjects into three groups:
  - UC Group:
    - Written Use Cases
  - US&OC Group:
    - Written User Stories
    - Access to an On-Site Customer
  - UC+US&OC Group:
    - Written Use Cases
    - Written User Stories
    - Access to an On-Site Customer

# Procedure

➤ Background Questionnaire (Education, Software Engineering experience, Requirements Engineering experience, Java Web application experience, etc.)	~10 min
➤ Tutorials ➤ Familiarization Task	~10 min
➤ Maintenance Tasks (3 tasks) ➤ Design	~120 min
➤ Debriefing Interview (Comments on experience, opinions, feedback)	~10 min
<b>TOTAL</b>	<b>~150 min</b>

# Maintenance Task

---

- Our subject system was a website through which one could purchase boats over the Internet
  - Java technology (JSP/Servlets)
  - Medium complexity
  - No external database required
  - Example code from 'More Servlets and JavaServer Pages' by Marty Hall, 2001
- Modify an existing feature of the website and add two new features
  - Add quantity of items in the shopping cart
  - Modify quantity of items in the shopping cart
  - Delete items from the shopping cart

# Results

---

## ■ Requirements

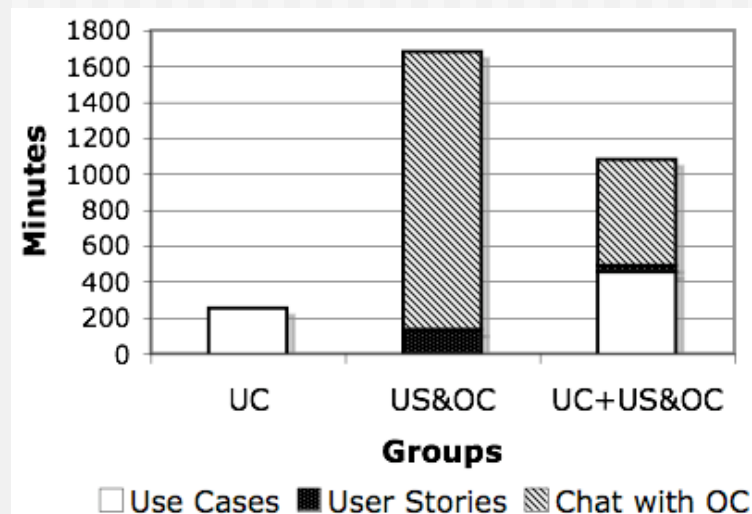
- Subjects using Use Cases and Agile Requirements:
  - Spent less time understanding requirements
  - Spent less time asking questions
- Results were statistically significant ( $p < 0.05$ ) using non-parametric tests

## ■ Implementation

- Requirement format did not have a great impact on how well subjects completed the maintenance task
  - It seems that a key factor was the analytical ability of each subject
- Results were not statistically significant using non-parametric tests

# Results - Requirements

- Subjects using Use Cases and Agile Requirements (UC+US&OC) spent less time understanding requirements than subjects using only Agile Requirements (US&OC). Subjects using only UC spent even less time than the other two groups.
- Subjects using Use Cases and Agile Requirements (UC+US&OC) spent more time reading Use Cases than the group using only UC.
  - May be attributed to the OC. Subjects needed to read and understand UC to frame their questions.



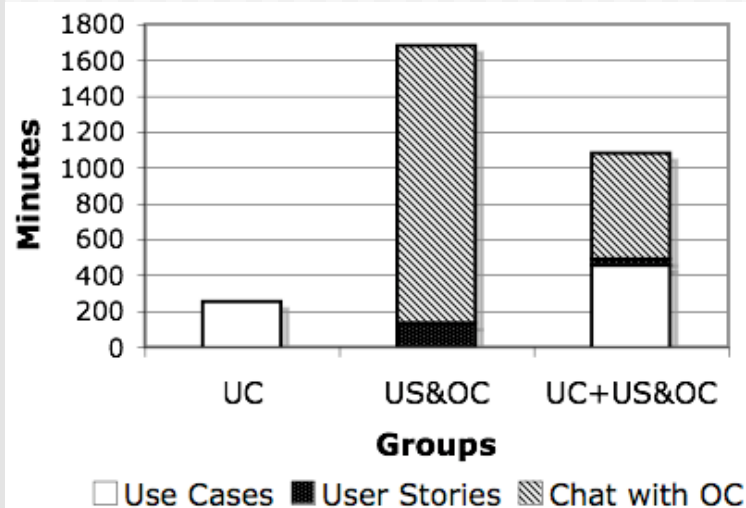
# Results - Requirements cont.

- Subjects using Use Cases and Agile Requirements (UC+US&OC) spent less time reading User Stories than the group using only Agile Requirements.
  - May be attributed to the fact that subjects using Use Cases and Agile Requirements preferred to read the Use Cases which provided more detailed information rather than to use the User Stories.

Average/Group	UC (mm:ss)	US&OC (mm:ss)	UC+US&OC (mm:ss)
Time reading Use Cases	04:13	-	07:34
<b>Time reading User Stories</b>	-	<b>02:12</b>	<b>00:37</b>
Time asking relevant questions to the OC	-	22:46	09:12
Time asking irrelevant questions to the OC	-	03:05	00:37
<b>Total time understanding requirements</b>	<b>04:13</b>	<b>28:03</b>	<b>18:00</b>

# Results - Requirements cont.

- Subjects using Use Cases and Agile Requirements (UC+US&OC) spent less time asking questions to the OC.
  - May be attributed to the fact that subjects using only Agile Requirements also had to elicit the requirements.
- Subjects using Use Cases and Agile Requirements (UC+US&OC) asked a fewer number of relevant and irrelevant\* questions than subjects using only Agile Requirements. The UC+US&OC had a bigger rate of relevant/irrelevant questions.



Average/Group	US&OC	UC+US&OC
Number of relevant questions to the OC	6.00	4.00
Number of irrelevant* questions to the OC	1.67	0.33

\* A question was considered **irrelevant** if it was about a topic that a typical customer could not answer or if it asked about features outside the scope of the tasks.

# Results - Implementation

- We scored the implementation, if completed, otherwise, we scored the design.
- The results were not statistically significant.

Average/Group	UC (score)	US&OC (score)	UC+US&OC (score)
Functionality score	18.17	19.17	17.67
Validations and messages score	5.33	1.33	1.67
<b>Overall score</b>	<b>23.50</b>	<b>20.50</b>	<b>19.34</b>

# Results - Implementation cont.

---

- We expected that subjects using more requirement formats would be able to produce better implementations.
- There was almost no difference in the performance among the groups.
- Based on our observation, it was not the requirements format that made a difference, but rather the analytical ability of each subject.
- This lack of difference brings into question the premise that improvements in tools and methods can result in quality improvements for the software produced.

# Results - Subject Feedback on Requirement Formats

- Strengths and weaknesses of the requirement formats

	<b>Strengths</b>	<b>Weaknesses</b>
<b>Use Cases</b>	Gives context Provides detail Provides business logic	Too much to read Does not have UI
<b>User Stories</b>	Provides good overview Gives the developer flexibility Changes are easy to identify and implement	Not enough detail User stories may overlap No specific exception handling details
<b>On-Site Customer (via chat)</b>	Can get quick answers to questions Fills implementation gaps Can do other work simultaneously	Absence of physical presence may cause communication problems Prefer talking to typing Lack of written skills

# Threats to Validity

---

- The number of subjects (9) was small.
  - Enough for a preliminary study.
  - More subject would improve the external validity.
- Subjects did not have enough experience with the technology used in the experiment.
  - Only one out of nine subjects finished the implementation task.
  - Serious threat. We tried to mitigate it by including a design task.
- Assignment of subjects to each group based on their background and experience before running the experiment.
  - We are aware that our groups might not be perfectly balanced.

# Threats to Validity cont.

---

- Only one researcher scored the source code and the design for all subjects.
  - It could have introduced a bias.
  - It gave a consistent scoring for all subjects.
- We did not include a way to measure or quantify the analytical ability of our subjects.
  - Serious threat that led us to inconclusive results in the implementation portion of the task.

# Summary

---

- Use Cases and Agile Requirements are complementary. Subjects who had access to both, spent less time understanding requirements. They also spent less time asking questions to the On-Site Customer and asked better questions.
- The requirements format didn't have a great impact on how well the subjects completed the maintenance task. It appeared that the most important factor was the analytical ability of each subject.

Thanks!

---