MGMT70043 Software Testing

Lesson 5: Use Case Testing and Testable Requirements

MGMT70043 LESSON 5

Agenda

1. Introduction and Overview
    - Quiz 4 based on Lesson 4 (and on textbook chapter 4)
    - Introduce group project
    - Lesson 5: Use case testing and other topics

2. Lesson 5: Use case testing and other topics
    - Testable requirements
    - Acceptance criteria
    - Use case testing
    - Agile testing

MGMT70043 LESSON 5

Testable requirements

- In Lesson 3, we discussed static testing as a means to ensure that requirements are testable. But how do you know that a requirement is testable? Ideally, the lack of testability should be identified as part of the review process.
- In this lesson, we will discuss what to look for when checking for testability. In particular, we will discuss how to specify **testable non-functional requirements**.

When checking for non-testable requirements, look for the following:

- **Ambiguous** requirements
- Requirements that are **open-ended**
- Requirements that are **not measurable**

MGMT70043 LESSON 5

Ambiguous requirements

- The problem with the natural language spoken by humans, as opposed to computer languages, is that natural language is ambiguous.
- For example, a common non-functional requirement is **usability**. But usability can refer either to **ease of learning** (how quickly users are able to learn the system) or to **ease of use** (how easy the system is to use).
    - The following is an example of an "ease of learning" usability requirement that is specified in a testable way: "95% of customer service agents will be able to update the customer's credit card expiry date after attending one training session".
    - The following is an example of an "ease of use" usability requirement that is specified in a testable way: "Users will be able to update their credit card expiry date by navigating to three screens or less."

MGMT70043 LESSON 5

Ambiguous requirements

When checking for **ambiguity**, look for the following:

- Words ending in "ly" (quickly, user friendly, clearly, easily)
- Words containing "ize" (minimize, maximize, optimize)
- Other key words (flexible, accommodate, safe, fast, easy, sufficient, small, large, improve, reduce, etc.)
- Passive wording
- Pronouns (he, she, it, they, them)

When checking for **open-ended requirements**, look for words and phrases such as "all", "most", "every", "sometimes" and "including but not limited to".

- For example, the following requirement is not testable: "The admin screen must be displayed to all applicable user roles, including but not limited to...."

MGMT70043 LESSON 5

Measurable requirements

- In order for requirements to be testable, they have to be **measurable**.
- **Non-functional requirements** are often defined in a way that is not measurable.
- For example, "The system must have high availability."
- A better way to define this requirement would be: "The system must be available 99.9 percent of the time."

MGMT70043 LESSON 5

Non-functional requirements

The following examples of measurable non-functional requirements are taken from Linda Westfall's presentation "Writing Testable Requirements".

Reliability: [mean time] for a defined [product or component] to experience defined [failure type] under defined [conditions]

Availability: [percentage] of defined [time period] a defined [product] is [available] for its defined [tasks]

Security: [probability] for a defined [product] to [handle (e.g., detect, prevent, recover from)] a defined [attacks] under defined [conditions]

MGMT70043 LESSON 5

Non-functional requirements

Performance:

- **Throughput**: [quantity] of defined [work units] which can be successfully handled per [time unit]
- **Capacity:** [quantity] of defined [entities or components] that can be [task, action or state] under defined [conditions] under defined [conditions]
- **Response time**: [mean time or maximum time] of a defined [response] to a defined [event]

MGMT70043 LESSON 5

Non-functional requirements

Usability:

- **Productivity**: [quantity] of defined [tasks or actions] that can be performed using the product per [time unit] under defined [conditions] per [time unit] under defined [conditions]
- **Response time**: [mean time or maximum time] of a defined [response] to a defined [user stimulus]
- **Error rate**: [quantity] of user [error type] mistakes experienced per [time unit] under defined [conditions]
- **Likeability**: [quantity or percentage] of users that report liking the product

MGMT70043 LESSON 5

UAT acceptance criteria

As you are reviewing requirements documents for testability, and generating test cases from those requirements, you should also be looking forward to the User Acceptance Testing (UAT) phase of the project. This includes:

- **Defining acceptance criteria for key requirements, from a business stakeholder perspective**. Make sure that these acceptance criteria are identified as early as possible and included in the UAT test cases.
- **Identifying test cases that would be good candidates for UAT**. "Business as usual" test cases, which reflect the way the system will actually be used, should be the focus of UAT.

MGMT70043 LESSON 5

Group activity: UAT

- Do you have any "war stories" to share about stakeholders that did not accept a system after development was complete?
- What are the top five challenges when planning and facilitating UAT?
- What are strategies for overcoming each of these challenges?

MGMT70043 LESSON 5

Use cases

https://www.youtube.com/watch?v=nN7lTDWKP6g

MGMT70043 LESSON 5

Use Cases in BABOK v3

According to BABOK v3:

- Use cases describe the interactions between the primary actor, the solution (i.e. the system), and any secondary actors needed to achieve the primary actor's goal.
- A use case describes the possible outcomes of an attempt to accomplish a particular goal that the solution will support.
- Use cases are usually triggered by the primary actor, but may also be triggered by another system or by an external event or timer.
- A scenario describes just one way that an actor can accomplish a particular goal.

MGMT70043 LESSON 5

Advantages of use cases

Use cases are a common way of defining requirements because of their many advantages:

- Business stakeholders love use cases because they define the system from the end user's perspective. Like user manuals, use cases describe the steps that users must perform in order to achieve their goals.
- Project managers love use cases because they can be used to define scope. Because each use case describes a given functionality, a list of use cases can be used to limit the functionality that will be delivered.
- Testers love use cases because they make the tester's job much easier. Use cases already contain many of the elements that must be included in test cases and test scripts, so it is relatively easy to write test cases based on use cases — provided that the use cases are well-written.

MGMT70043 LESSON 5

Disadvantages of use cases

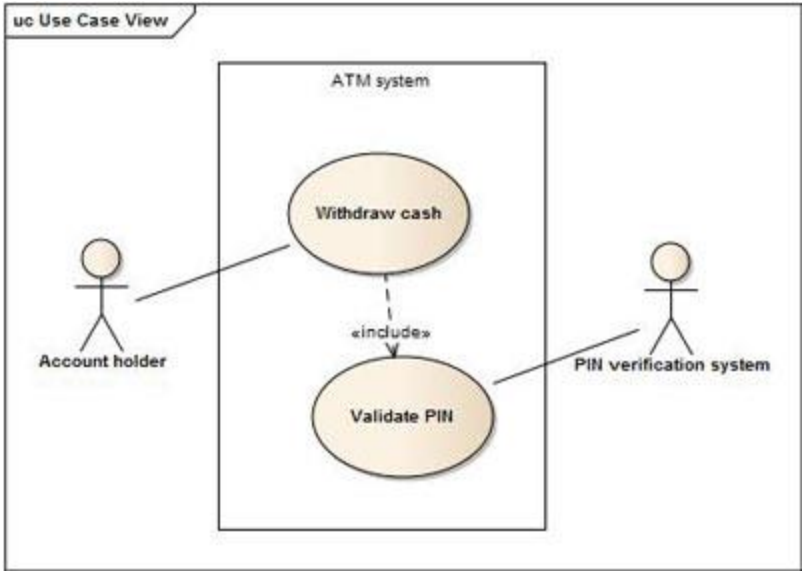
Disadvantages of use cases:

- Use cases can be overutilized, and are sometimes used to define requirements that don't lend themselves well to use case specification.
- Because use cases emphasize the most common "business as usual" scenarios, they emphasize how the system should work, as opposed to the ways in which the system can fail.
- Another problem with use cases is that they can be challenging to write correctly. Although use case templates are relatively straightforward, they are often filled out incorrectly, which makes it hard for the audience of the document, particularly developers and testers, to do their jobs correctly.

MGMT70043 LESSON 5

Use case models

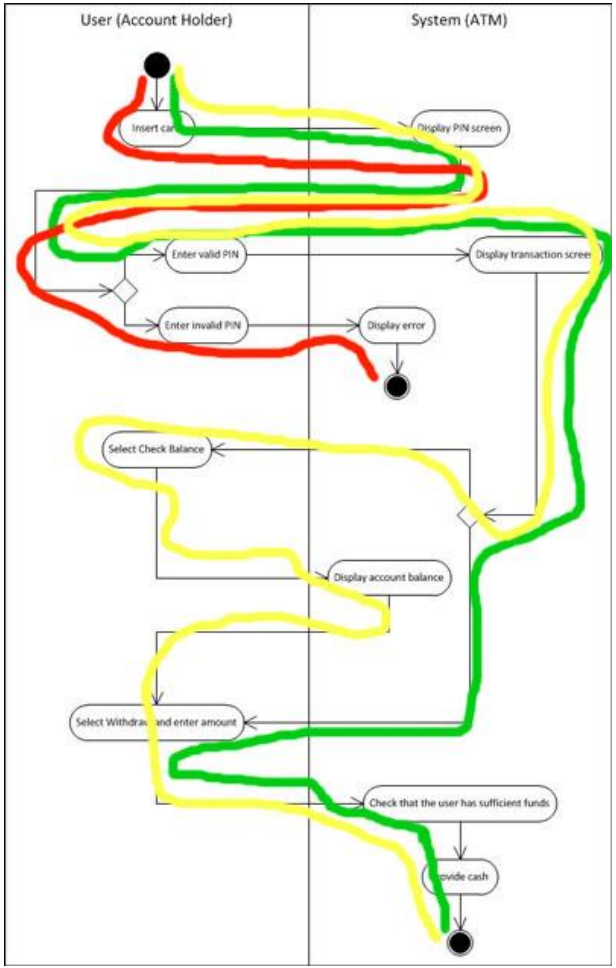
- In use case models, the system boundary is represented by a rectangle, each actor is represented by a stickman, and each use case is represented by an oval.
- Use case models are great for defining the system boundary and for understanding the relationships between use cases.
- For testing purposes, however, use case models don't contain enough information to create a test case specification.



MGMT70043 LESSON 5

UML activity diagrams

- **Activity diagrams** are a way to illustrate individual use cases so that they can be used to generate test cases.
- Activity diagrams with "swimlanes" are useful for showing the interaction between a system and various actors.
- They are also useful for showing use cases with multiple flows.
- The green line indicates the basic flow, the yellow line indicates the alternative flow, and the red line shows the exception flow.



MGMT70043 LESSON 5

Use case specifications

- Writing use cases well takes skill and practice, but it will have payoffs in terms of reducing the effort required to create test cases.
- Written use case specifications contain much of the information required to create test case and test scripts, although they do not contain specific inputs or outputs, nor do they contain setup instructions.

Here is the information that must be included in a use case specification:

- **Use case ID**: A unique identifier for the use case. For example, UC-01.
- **Use case name**: The name of the use case in terms of the user goal, in Verb-Noun format. For example, Withdraw Cash.
- **Use case description**: A brief description of the use case. For example, "This use case describes the steps taken by the account holder to withdraw cash."

MGMT70043 LESSON 5

Use case specifications

Here is more information that must be included in a use case specification:

- **System**: The name of the system that is being specified. For example, the ATM system.
- **Primary actor**: The actor that initiates the action and that has a goal requiring the assistance of the system. For example, account holder.
- **Secondary actor(s)**: Any actor that is called upon by the system because it is needed to satisfy the primary actor's goal. For example, PIN verification sub-system.

MGMT70043 LESSON 5

Use case specifications

The following should also be included in a use case specification:

- **Preconditions**: Conditions that must be in place prior to the start of the use case. For example, the user has an active account and a valid bank card.
- **Postconditions**: Conditions that will be in place after the use case if the user has achieved his or her goals. For example, the ATM has provided cash to the user.
- **Trigger**: The action or event that triggers the use case. For example, the account holder inserts the bank card into the ATM. Note that some use case are triggered by events, such as "the first day of the month arrives" or "the balance falls below zero".

MGMT70043 LESSON 5

Use case specifications

The following should also be included in a use case specification:

- **Normal Flow**, also known as **Basic Path**: The simplest steps required for the primary actor to achieve his or her goal. By convention, the steps are often numbered. Ideally, the steps should be written in alternating "He Says, She Says" format, like a dialogue or a screenplay.
- **Alternative Flows**: Alternative ways in which the primary actor can achieve his or her goal.
- **Exceptions**: Errors that result in the user not achieving his or her goal.

MGMT70043 LESSON 5

Basic path example

Here is an example of a Basic Path for a use case specification:

1. The user inserts the card.
2. The system displays the PIN screen.
3. The user enters the PIN.
4. The system verifies the PIN.
5. The system displays the transaction selection screen. The options are: Withdraw, Deposit, Check Balance.
6. The user selects Withdraw and enters an amount.
7. The system checks that the user has sufficient funds and provides the cash.

MGMT70043 LESSON 5

Alternate and exception flows

Here is an example of an alternate flow:

- 6a. The user selects Check Balance.
- 6b. The system displays the account balance.
- 6c. The user selects Withdraw and enters an amount.

Here is an example of an exception flow:

- 3b. The user enters an invalid PIN.
- 3c. The system displays a "try again" error message.

MGMT70043 LESSON 5

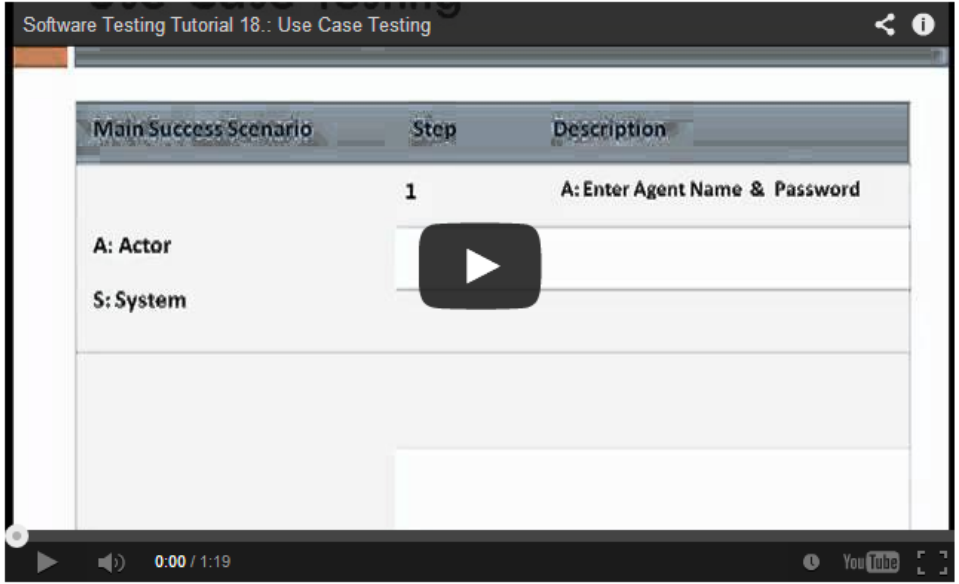
Tips for writing use cases

To write clear use cases that can be readily transformed into test cases, do the following:

- Use the **active voice**, so your reader can tell who is doing what. An example of the active voice: "The system validates the PIN". An example of the passive voice with a subject: "The PIN is validated by the system". An example of the passive voice without a subject: "The PIN is validated".
- Use the **present tense**, as it is more vivid. An example of the present tense: "The system validates the PIN". An example of the future tense: "The system will validate the PIN".
- Put the **right information into the right slots**, so key details doesn't get lost. For example, don't put a business rule in the use case description.
- Write **short sentences**, as they are easier to digest.

MGMT70043 LESSON 5

Use case testing



https://www.youtube.com/watch?v=ijtvAvapsP0

MGMT70043 LESSON 5

Use case testing

In order to generate test cases and test scripts from a use cases, do the following:

- Create one test case per flow. The ATM use case above, for example, would generate three test cases.
- For each test case, fill in the preconditions and postconditions from the use case. Note: if the test case is for an exception flow, then the postconditions will be different.
- Fill in the Input Values using the actions performed by the user, but with specific values. For example, instead of "User enters PIN", state "John Smith enters 1234".
- Fill in the Expected Results using the actions performed by the system, but with specific values.
- In the traceability/cross-reference section, list the use case ID.

MGMT70043 LESSON 5

Use case testing

In order to generate test procedures (test scripts) from a use cases, do the following:

- Use the “steps” section of the use case to generate the steps required for the test procedure.
- Add any necessary setup steps. In the ATM example, this might involve creating a user named "John Smith" with a given account balance.

MGMT70043 LESSON 5

Group activity: use cases

- Create a use case specification for an application that allows users to order pizza online.
- Then create a test case and a test script based on the use case specification.

MGMT70043 LESSON 5

Agile testing

https://www.youtube.com/watch?v=yj8E4fhB1F8

MGMT70043 LESSON 5

Agile testing

- On projects that use the agile software development methodology, the requirements are written in the form of **user stories**.
- User stories consist of single sentences that take the form:

```
As a [user role], I want to [desired
functionality] in order to [business value].
```

- An example of a user story for an ATM system would be: "As an account holder, I want to be able to view my account balance in order to see how much cash is available".

MGMT70043 LESSON 5

Agile testing

- Test cases in agile projects take the form of **acceptance criteria**, which are sometimes called **conditions of satisfaction** (CoS).
- Acceptance criteria are written at the same time as the user story and enable the development team to provide an estimate for the user story.
- The acceptance criterion for the previous user story would be: `"I can see my account balance".`

MGMT70043 LESSON 5

Group activity: Agile test

- You are the Product Owner on an Agile development project at a telecommunications company.
- The goal of the project is to migrate prepaid cell phone customers to the more profitable post-pay platform.
- You represent multiple stakeholder groups, including Marketing, Call Centre, Finance and Credit.
    1. For each stakeholder group, define one user story.
    2. For each user story, define the corresponding condition of satisfaction (CoS).
    3. Place the user stories into a Product Backlog, prioritizing them in terms of business value.

References

- International Institute of Business Analysis (2015). 10.47 Use Cases and Scenarios. In *BABOK v3: A Guide to the Business Analysis Body of Knowledge* (pp. 356-359). Toronto: IIBA.
- Berger, B. (n.d.). The Dangers of Use Cases Employed as Test Cases. *Test Assured*. Retrieved from testassured.com.
- Donaldson, N. (2012, June 26). User stories part 2: Acceptance criteria. *Boost Agile Blog*. Retrieved from boostagile.com.
- Wiegers, Karl. (2012, July 26). The Use Case Technique: An Overview. *Modern Analyst*. Retrieved from modernanalyst.com.
- Westfall, L. (2010, December 10). Writing Testable Requirements. *CQAA Webinar*. Retrieved from all cqaa.org.