SWEN30006 Software Modelling and Design
Workshop 1: Use Cases and Java/Eclipse – Partial Solution

School of Computing and Information Systems
University of Melbourne
Semester 2, 2020

Disclaimer: The solution provided in this document is one of many possible solutions. Your solution may also be correct or suitable even though it looks different from the solution in this document. This is because the solution can be developed based on different reasoning and design decisions. In addition, the solution in this document may be *incomplete* for the purpose of learning. We encourage you to discuss with the tutor during the workshop hours to check the correctness of your solution.

Part 1 Use Cases

Q1.1 Write a Main Success Scenario

This example is provided in the form of a fully dressed use case (see Larman 6.8). It should be easy to recast this in the brief or casual format.

Use Case UC1: Update Catalog Item

Scope *LazyFair* software system

Primary Actor Store Manager

Stakeholders and Interests

Store Manager wants to be able to quickly and correctly make changes to an existing catalog item.

Preconditions Store Manager has authenticated. Item exists in catalog.

Postconditions Item in catalog updated with entered and confirmed changes.

Main Success Scenario

1. *Include UC2: Search for Catalog Item [assuming seperate use case for this]*
2. *Display existing attributes of catalog item*
3. *Edit attributes of catalog item*
4. *Confirm and save changes*

Frequency of Occurrence tens of times on a daily basis.

Q1.2 Group Discussion

Below is an example of a use case checklist focussed on ensuring that the use case flows are complete, correct and consistent. You may have found issues corresponding to items on this list, issues with other aspects of the use case (e.g. preconditions), or perhaps you were shown a very well written use case!

- Is it clear how the use case starts?
- Does the flow have a definite ending?
- Does each step in the scenario contain the same level of abstraction?
- Does each step in the scenario describe something that can actually happen and that the system can reasonably detect?
- Does each step make progress toward the goal?
- Are there any missing steps? Is it clear how to go from one step to the next? Does the sequence of communication between the Actors and the use cases conform to the users' expectations?
- Does each step describe how the step helps the Actors achieve their goals?
- Is each step technology-independent? Is it free of technical details and inadvertent design decisions?
- Are the steps correctly numbered?
- For each alternate flow, are the conditions for initiation of the flow clearly defined?
- For each alternate flow, is it clear how the use case ends or where in the basic flow that the use case resumes?

Q1.3 Add Some Alternate Scenarios

Here are some possible alternative flows. The first is an alternative at a specific step, the next two can occur at any time.

2a. Search unsuccessful - end use case

***a.** User requests abort update - end use case [assumes confirmation]

***b.** Catalog not accessible.

1. System notifies user of problem and logs details
2. Initiate periodic reconnection (including status/log updates)

Q1.4 Draw a Use Case Diagram

The example above focussed in Update, and the use case diagram reflects this (and the other options provided for this question). In general, it is worth considering whether the full set of CRUD operations (Create, Retrieve, Update, Delete) are required, if one seems to be necessary.

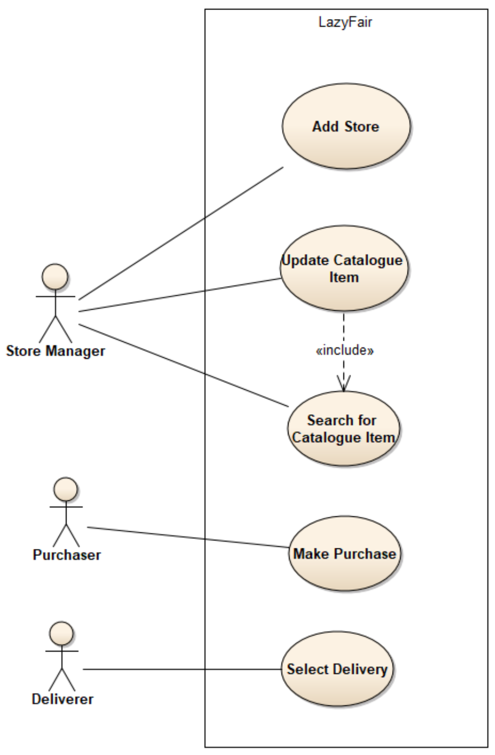

Figure 1: Use Case Diagram for LazyFair

Q1.5 Show Your Understanding

These questions are trying to get you to reflect on your learning, and in doing so help consolidate the concepts. There isn't a single correct answer for each question: following are examples of points you could have written:

1. Clarity and a well-defined scope. A glossary of domain-specific terms. A complete description (i.e., main success scenario + alternate paths).
2. Trick question! You should write as many use cases as you need.
3. The use case text is the main part of the use case, and the diagram shows a summary of what the use cases are and how the actors interact with them. You don't need both, but if you only have one it should be the use case text.
4. Primary: interacts with the system under discussion to achieve a goal.
   Supporting: provides a service to the system under discussion. Is often a computer.
   Offstage: has an interest in the behaviour/outcome of the use case.
5. Different people will find different aspects challenging.

Part 2 The Development Environment

For the last question, there are many ways to complete the task. A simple search will find many options: https://www.google.com/search?q=java+word+frequency

We encourage software reuse, and we know some of you did search for a solution: did you understand fully the solution you presented?