CSc 322 - Software Engineering

Lab – Fall 2017

Teaching Assistant: Isatou Sanneh
Email: isanneh00@citymail.cuny.edu

Use-Case Modeling

February 16th, 2017

Introduction

Use case diagrams, which provide a graphical overview of a system's functionality, help end-users, developers, and domain experts get a common understanding of a system. It is used to capture the basic functionality i.e. use cases, and the users of those available functionality, i.e. actors, from a given problem statement.

Use case diagrams belong to the category of UML behavioral diagrams. Use cases are usually identified during the early stages of the project. Use case diagrams aim to present a graphical overview of the functionality provided by the system. Thus it is highly useful for exposing requirements and planning almost every project. It consists of a set of actions (referred to as use cases) that the concerned system can perform, one or more actors, and dependencies among them.

Actor

An actor can be defined as an object or set of objects, external to the system, which interacts with the system to get some meaningful work done. Actors could be humans, devices, or even other systems. For example, consider the case where a customer *withdraws* cash from an ATM. Here, customer is a human actor.

2 types of actor classification:

- *Primary actor:* They are principal users of the system, who fulfill their goal by availing some service from the system. For example, a customer uses an ATM to withdraw cash when he needs it. A customer is the primary actor here.
- *Supporting actor:* They render some kind of service to the system. "Bank representatives", who replenish the stock of cash, is such an example. It may be noted that replenishing stock of cash in an ATM is not the prime functionality of an ATM.

Actor Designing Consideration:

- Who/what will be interested in the system
- Who/what will want to change the data in the system
- Who/what will want to interact with the system
- Who/what will want information from the system

Use Case

A use case is simply a functionality provided by a system. Use case should ideally begin with a verb. Should not be open ended. Register, wrong. Register New User, right. Once the primary and secondary actors have been identified, we have to find out their goals i.e. what are the functionality they can obtain from the system.

Continuing with the example of the ATM, withdraw cash is a functionality that the ATM provides. Therefore, this is a use case. Other possible use cases include check balance, change PIN, and so on.

Use cases include both successful and unsuccessful scenarios of user interactions with the system. For example, authentication of a customer by the ATM would fail if he/she enters a wrong PIN. In such a case, an error message is displayed on the screen of the ATM.

Subject

Subject is simply the system under consideration. Use cases apply to a subject. For example, an ATM is a subject, having multiple use cases, and multiple actors interact with it. However, one should be careful of external systems interacting with the subject as actors.

Graphical Representation

An actor is represented by a stick figure and name of the actor is written below it. A use case is depicted by an ellipse and name of the use case is written inside it. The subject is shown by drawing a rectangle. Label for the system could be put inside it. Use cases are drawn inside the rectangle, and actors are drawn outside the rectangle, as shown in fig. 1.

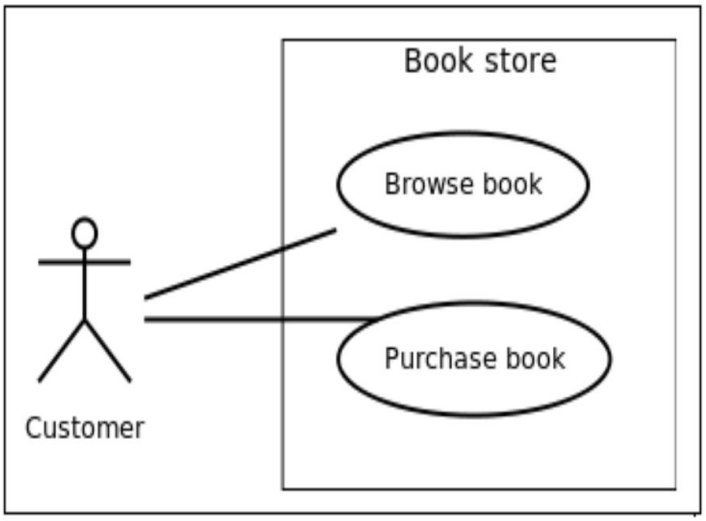

Figure 1: A use case diagram for a book store

Association between Actors and Use Cases

A use case is triggered by an actor. Actors and use cases are connected through binary associations indicating that the two communicate through message passing. An actor must be associated with at least one use case. Similarly, a given use case must be associated with at least one actor. Association among the actors are usually not shown. However, one can depict the class hierarchy among actors.

Use Case Relationships

Three types of relationships exist among use cases:

1. Include relationship
2. Extend relationship
3. Use case generalization

Include relationship

Include relationships are used to depict common behaviour that are shared by multiple use cases. This could be considered analogous to writing functions in a program in order to avoid repetition of writing the same code. Such a function would be called from different points within the program. Include relationship is depicted by a dashed arrow with an «include» stereotype from use case A to use case B, with B being the use case that is included.

Example

For example, consider an email application. A user can send a new mail, reply to an email he has received, or forward an email. However, in each of these three cases, the user must be logged in to perform those actions. Thus, we could have a login use case, which is included by compose mail, reply, and forward email use cases. The relationship is
shown in fig. 2.

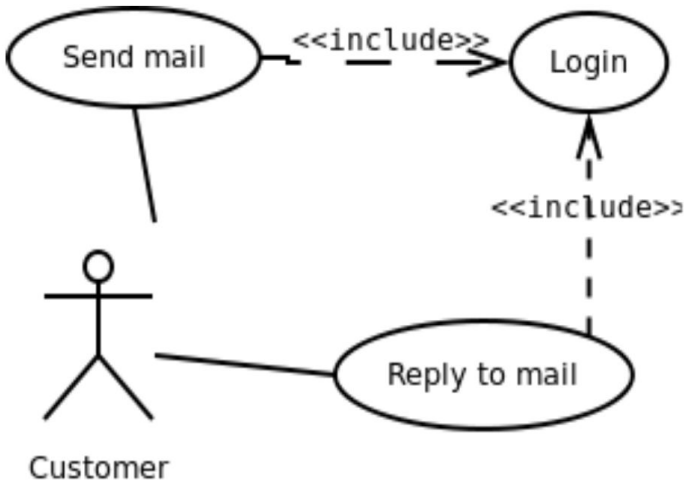

Figure 2: Include relationship between use cases

Extend Relationship

Use case extensions are used to depict any variation to an existing use case. They are used to specify the changes required when any assumption made by the existing use case becomes false. Extend relationship is depicted by a dashed arrow with an «extend» stereotype from use case A to use case B, with B being the use case that is extended.

Example

Let's consider an online bookstore. The system allows an authenticated user to buy selected book(s). While the order is being placed, the system also allows to specify any special shipping instructions, for example, call the customer before delivery. The Shipping Instructions step is optional, and not part of the main Place Order use case. fig. 3 depicts such relationship.

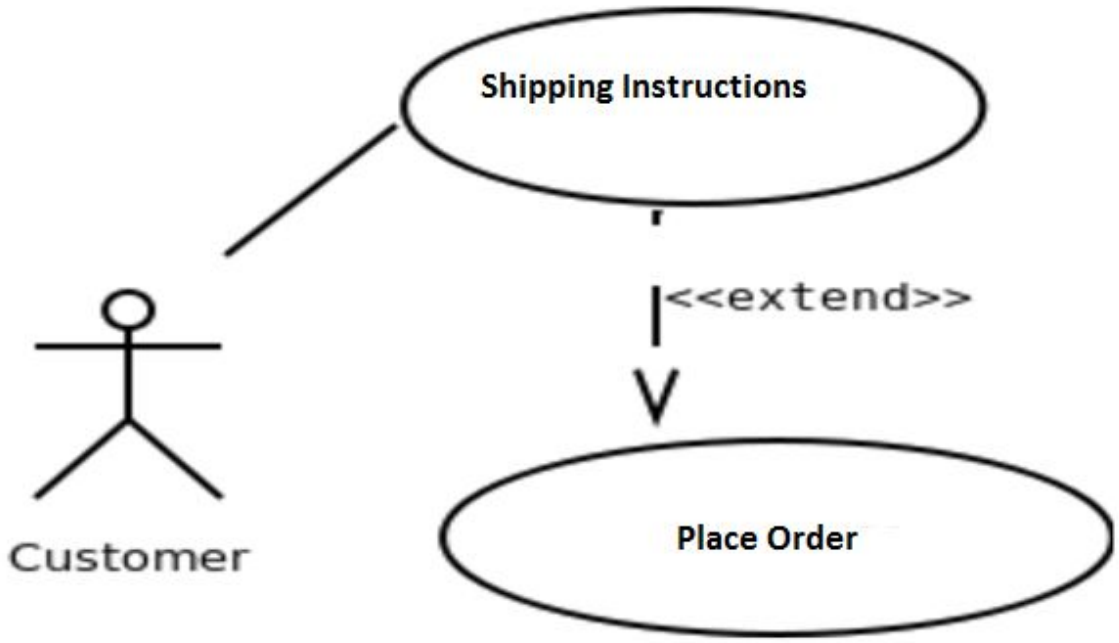

Figure 3: Extend relationship between use cases

Generalization Relationship

Generalization relationships are used to represent the inheritance between use cases. A derived use case specializes some functionality it has already inherited from the base use case. Generalization relationship is depicted by a solid arrow from the specialized (derived) use case to the more generalized (base) use case.

Example

To illustrate this, consider a graphical application that allows users to draw polygons. We could have a use case, "draw polygon." Now, rectangle is a particular instance of polygon having four sides at right angles to each other.

So, the use case “draw rectangle” inherits the properties of the use case “draw polygon” and overrides its drawing method. This is an example of a generalization relationship. Similarly, a generalization relationship exists between “draw rectangle” and “draw square” use cases. The relationship has been illustrated in fig. 4.

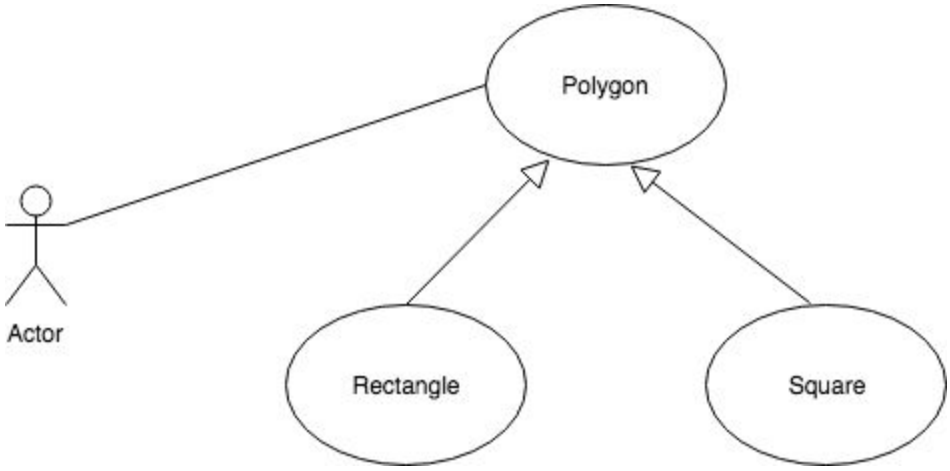

Figure 4: Generalization relationship among use cases

TASK 1

Part 1 Identify the actors and use cases for the following problem statement

An automated teller machine (ATM) lets customers to withdraw cash anytime from anywhere without requiring involvement of any banking clerk or representative. A customer must insert his/her ATM card into the machine and authenticate himself by typing in his/her personal identification number (PIN). He cannot avail any of the facilities if the PIN entered is wrong. Authenticated customers can also change their PIN. They can deposit cash to their account with the bank. They can also transfer funds to any other account. The ATM also provides options to the user to pay electricity or phone bill. Every morning, the stock of cash in the ATM machine is replenished by a representative from the bank. Also, if the machine stops working, then it is fixed by a maintenance guy.

Part 2 Exercise on Use Case Relationships

1. Consider a Shop Sales System. A sales officer uses the system to process/record requests of both customer order and faulty goods return. Both processes will need to identify the customer first.
2. Consider an online selling/buying website. The website requires the seller and buyer to register their bank account information. A separate process will be called if the bank account is suspended/unavailable/illegitimate.
3. An order management system accepts ordering of products by phone or internet. Write the generalization for this, where that base use case will be used by the order registry clerk.

Part 3 Mail Order System

Identify Actors and Use Cases for the following problem and draw a use case diagram using ArgoUML tool with appropriate relationships.

In order to improve the operational efficiency of a mail order company, the chief executive officer is interested in computerizing the company's business process. The major business activities of the company can be briefly described as follows:

A customer registers as a member by filling in the membership form and mailing it to the company. A member who has not been active (no transactions made) for a period of one year will be removed from the membership list and he/she needs to re-apply for the reinstatement of the lapsed membership.

A member should inform the company of any changes of personal details such as home address, telephone numbers, etc. A member can place an order by filling out a sales order form and faxing it to the company or by phoning the Customer Service Assistant with the order details.

The Customer Service Assistant first checks for the validity of membership and enters the sales order information into the system.

The Order Processing Clerk checks the availability of the ordered items and holds them for the order. When all the ordered items are available, he/she will schedule their delivery.

The Inventory Control Clerk controls and maintains an appropriate level of stock and is also responsible for acquiring new items.

If there is a problem with an order, members will call the Customer Service Assistant. The Customer Service Assistant will take appropriate action to follow up the sales order.

Members may return defective goods within 30 days and get their money back.

The system will record the name of the staff member who initialized an updated transaction to the system.