SWEN30006 Software Modelling and Design
Workshop 5: Fundamental Design Patterns

School of Computing and Information Systems
University of Melbourne
Semester 2, 2019

Completion: You *should* complete the exercises ***as a team***. To receive a workshop mark, you **need** to demonstrate your active participation during the workshop. Your tutor will observe your participation for you to be eligible to receive a mark for this workshop. The active participation include (but not limit to) present your solutions, help your teammates, and actively engage with the team. Attending the workshop with only passive participation will *not* be given a mark. See the Workshop Overview under Subject Information on the LMS for more details.

Requisite Knowledge and Tools

It is expected by this point that you are familiar with the following terms and concepts:

- Static Domain and Design Modelling with UML Class Diagram Notation
- GRASP
- Converting a Domain Model to a Design Model
- Designing for Visibility
- Dynamic modelling with UML Sequence Diagram Notation

It is suggested that you have watched the lectures on GRASP and read chapters 9, 17, 19 and 25 of Applying UML and Patterns by Larman (the prescribed text).

Introduction

This week we will be looking at a POS system, one which is different to the textbook case study. For this system, we have provided a description below and set of rules as a client might. We have also detailed one use case for making a sale where a credit card is used for payment. We have not described the other use cases or formally distilled the requirements; this will be a process that *you will have to complete before beginning the exercises*.

Our goal is still to control the assignment of responsibilities, and use the GRASP patterns (and later the Gang of Four patterns and other patterns) as a framework to improve our design decision making. We will also continue to make use of the GRASP patterns this week and use UML for modelling. We have now covered all the GRASP patterns; these patterns are discussed in depth in Chapters 17 and 25 of the subject textbook:

- Information Expert
- Creator
- Low Coupling
- High Cohesion
- Controller
- Polymorphism
- Pure Fabrication
- Indirection
- Protected Variations

By the end of the workshop you should be more familiar with the GRASP Patterns and their application in modelling software systems. You should be comfortable applying this knowledge to design software systems and represent these in UML.

This week we will perform a series of analysis and design processes for a single example. We will be designing the software for a mobile Point of Sale system (POS System) from a series of rules and use cases. Note that this is a different POS System from the textbook case study (NextGen POS), though your understanding of the NextGen POS example will better prepare you to understand and model the mobile POS System outlined below. We will start by performing static modelling techniques, followed by dynamic modelling.

We structure the workshops in this manner to reflect the software design process. In application, it is likely that you will start by developing static models before moving to dynamic models. Once you have created dynamic models, you will likely find that you have discovered discrepancies within your static models, and therefore need to revise these models as a result. This process continues until an appropriate design has been found, one that meets the explicit requirements and is maintainable and extendible, with static modelling informing dynamic modelling and dynamic informing static. These exercises will attempt to demonstrate this process to you.

The Point of Sale (POS) System

A timber and hardware store sells different kinds of timber, paints and painting products, lights and household electrical products, plumbing and bathroom products, and gardening products.

A project is underway with the aim of equipping staff with mobile service terminals that they can use to advise customers, get the latest prices if not displayed and inform customers about the latest discounts and sales in the store. Once sales are made the aim is to automatically adjust the level of stock through a stock monitoring system and to release the items sold so that they can pass through the exits monitored by a security system without raising an alarm.

Your task is to design a Mobile Service Terminal (MST) and inventory monitoring but not the security subsystem. You need to assume that you have an interface to the security subsystem that registers the barcodes of released items.

Rules of POS Operation

1. Store staff can walk around the store with an MST. A staff member can complete sales of items without the need to go to a register or sales station provided that customers have credit cards or store cards.
2. To complete a sale with a *credit card* the following sequence of steps must occur:
    i. The item's barcode is scanned.
    ii. The credit card is inserted into the machine and the card is validated against the credit agency for the card.
    iii. If the card is valid then the price of the item is charged to the card if enough credit remains in the card. If not the sale will be *declined* and no further action is possible for this sale.
    iv. If the amount is accepted then the item is 'released' from the store by sending its barcode to the central monitoring system. Released items do not raise alarms when they pass through an exit monitored by the security system.
3. Store cards are discounted debit cards, that is, the card acts as a debit card but applies a 10% discount to all non-sale and non-discounted items. The following sequence of steps must occur for a sale with a store card:
    i. The item's barcode is scanned.
    ii. The store card is inserted into the machine and the card is validated against the customer's debit balance.
    iii. If the card balance is sufficient for the item then the sale is made and the customer's balance is adjusted according to the price of the item. If the customer's balance is not sufficient then the sale will be declined and they can be directed to a cash register for combined cash/card sales.
    iv. If the sale proceeds then item is 'released' from the store.
4. The staff can also handle stock arrivals using their MSTs. They do this by scanning barcodes or stock numbers of new items. Once an item is scanned the inventory is updated automatically.
5. If no barcode or stock number exists on the item then they can look up the item's stock number by searching based on a description of the item. The staff member must select the correct item in this case and the selected item's stock number is used as in 4. above.
6. Supervisors can use the mobile data terminals to check sales for a specific sales consultant, or to view the available margin in an item's price.
7. Card sales must be done reliably, that is, that there is a 99.9999% chance that the sale goes through without a failure.

Use Case for the POS System

Below we present a single use case for the point of sale system.

Use Case POS-1 Credit Card Sale

Use Case	POS-1 Credit Card Sale
Actors	Customer, Staff Member, Credit Card Agency
External Trigger	Staff member initiates a sale using the Mobile Service Terminal (MST).
Internal Trigger	N/A
Brief Description	The use-case describes how the customer, staff member, credit card agency and MST system interact to achieve a credit card sale.

Flow of Events

1. The staff member holding the MST initiates a sale.
2. The staff member scans the barcodes of all the items in the sale.
3. The MST calculates a total sale price applying any customer discounts or sale policies that are in place.
4. The credit card details are read by the MST .
5. The MST system validates the card details against the credit agency. The validation procedure is as follows:
   i. The card details are read by the MST.
   ii. The customer types in their PIN.
   iii. The card number, the sale amount and the pin are sent to the credit agency.
   iv. The credit agency sends a message saying a card is valid and that the sale is approved.
6. **If** the card is valid **and** enough credit remains in the account linked to the card then the total price of the items is charged to the card account.
7. The MST system records the transaction and stores the transaction for accounting and later data analysis.
8. All of the purchased items are *released* from the store by sending a list of barcodes to the security system:
   - Released items do not raise alarms when they pass through an exit monitored by the security system.

Alternative Flows

- **6a)** If the card is not valid or the account does not have enough credit for the item then the sale will be declined and no further action is possible for this sale. The use-case ends.

Exercise One - Core Working with your team, your aim is to create a domain model of the MST store sales and inventory monitoring system. Think of a single staff member making a sale. Create a model (domain classes and the relationships between them) of the objects involved in each of the following scenarios:

1. making a credit card sale;
2. making a store card sale;
3. looking up the inventory.

You should create three different models initially.

Now develop a domain model from your three models. Step through each of the requirements in the rules above and add any classes that are missing from your domain model. Look for where you can perform generalisation and use inheritance and aggregations, and look for where you can simplify. You may well need to iterate and adjust the models as you go adding classes and adding/deleting attributes - but this is all part of the *fun of modelling!*

Exercise Two - Core Transform your domain model into an initial design class diagram, assigning responsibilities to your design classes. Again, you should expect to iterate, adjusting classes, attributes and methods.

You will revisit your static design model based on dynamic modelling so don't expect to get everything right in the initial version.

We will be performing dynamic modelling techniques to analyse a single use case, and use the information from this analysis to update our static models.

Exercise Three - Core Using the design class diagram that you created earlier, along with the Credit Card Sale use case presented above, your task is to develop a sequence diagram that shows the flow of interaction throughout the above use case.

To develop the sequence diagram you will need to think carefully about the *methods* in each of the classes. The trick to modelling is to think of the *methods* as something that an object (of a class) must do to meet a responsibility as part of fulfilling a use case. Each action by an actor, and each response by the system must be mapped into methods in the appropriate classes.

Of course this could also mean that you will need to add new attributes to store the state of the new methods that you have added.

Each step in the flow of events in the use case should contain objects and actions that can be modelled objects and methods. Consider each step of the flow in turn and add its effects to the sequence diagram.

> Take care to ensure you are using proper UML syntax to describe loops and control flow as these are techniques that will be useful in your Project design submission.

Exercise Four - Core We now have a much better idea about how information will flow through the system, thanks to our sequence diagram. We also have a much better idea of which methods will be required on which classes.

Using this information, complete your design class diagram so that it contains all methods needed to implement the system. Be sure to include arguments along with return types.