# CS3342 Software Design

## 2023/24 Semester B

## Assignment 1

**(Online Submission via Canvas)**

| | |
|---|---|
| Student Name: | |
| Student ID: | |

| Section 1 | Section 2 | Section 3 | Total |
|---|---|---|---|
| 20/60 | 20/60 | 20/60 | 60/60 |
| | | | |

- **This is an individual assignment;** students are expected to complete this assignment individually, and to properly acknowledge the use of another person's work if necessary, i.e. references, citations.
- This assignment consists of 3 sections, please complete all sections.

**Caution on Plagiarism**

*Students who commit an act of academic dishonesty which jeopardizes the integrity of the learning and assessment process may be charged and be liable to disciplinary actions. Students are required to act truthfully and honestly in academic pursuits. Students who are found to have violated the principle of academic honesty will be subject to academic disciplinary actions. [CityU Rules on Academic Honesty, Effective from Semester A 2012/13]*

## Section 1 – Software Engineering (20 Marks)

**1. Software Engineering Research study – CILO1 (10 Marks):**

Please carry out an independent study according to your book readings and online research, to address one of the two questions below by writing a short essay. (500 words approx.)

a. What are the main software quality attributes and how can we measure them? Please give an example of a software system that has high or low quality in terms of one or more attributes and explain why. (Please provide your references.)

b. What are the benefits and challenges of the agile software process model? Please compare the agile model with traditional models such as the waterfall model with an example. (Please provide your references.)

*Place your answer in this box. 500 words or one A4 page approx. (excluding references)*

a.

1. Define software quality attributes and explain why they are important for software engineering. Mention some of the common attributes such as functionality, reliability, usability, efficiency, maintainability, and portability.
2. Discuss how to measure each attribute using metrics, standards, or methods. For example, functionality can be measured by testing the software against its requirements, reliability can be measured by calculating the mean time to failure or the number of defects, usability can be measured by conducting user surveys or experiments, etc. Provide references for each measurement technique.
3. Give an example of a software system that has high or low quality in terms of one or more attributes and explain why. For example, you could choose a software system that you have used or developed, or a well-known software system such as Facebook, Microsoft Word, etc. Describe the software system briefly and then analyze its quality attributes using the measurement techniques discussed in the body. Explain how the software system meets or fails to meet the quality criteria and what are the consequences or benefits of its quality level.
4. Correct references and grammar.

b.

1. Benefits of the agile model: Describe at least two benefits of using the agile software process model, such as flexibility, adaptability, customer satisfaction, team collaboration, etc.
2. Challenges of the agile model: Describe at least two challenges of using the agile software process model, such as lack of documentation, difficulty in measuring progress, resistance to change, etc.
3. Comparison with traditional models: Compare the agile model with traditional models such as the waterfall model. Discuss similarities and differences between these models. Explain in detail and support with examples and references.
4. Correct references and grammar.

**2. Roles of Variables – CILO4 (10 Marks)**

Study the following Java codes and identify the role of each variable declared in the code listing by completing the tables below (you may refer to the roles of variables lecture ppt):

```
public class Grades {
    public static void main(String[] args) {
        int student_id[] = { 1031, 1022, 3021, 2023, 2062, 3027, 4022};
        int student_sc[] = { 80, 84, 89, 79, 55, 92, 73};
        int fail = 0;
        int total = 0;
        int max_sc = 0;
        double average = 0;
        Boolean neg = false;
        final int student_num = student_id.length;
        for (int i = 0; i < student_num; i++) {
            int sc = student_sc[i];
            int id = student_id[i];
            total += sc;
            if (sc > max_sc) {
                max_sc = sc;
            }
            if (sc < 60) {
```

```
                fail++;
                neg = true;
            }
        }

        average = total / student_num;
        system.out.println();
        system.out.println("Number of student: " + student_num);
        system.out.println("Average score: " + average);
        system.out.println("Number of fail: " + fail);
        system.out.println("Maximum score: " + max_sc);
    }
}
```

| Variable | Role | (10 Marks) |
|---|---|---|
| *student_sc* | Organizer | (1 Mark) |
| *total* | Gatherer | (1 Mark) |
| *average* | Transformation | (1 Mark) |
| *i* | Stepper | (1 Mark) |
| *student_num* | Constant | (1 Mark) |
| *max_sc* | Most-Recent Holder | (1 Mark) |
| *id* | Temporary | (1 Mark) |
| *neg* | One-way flag | (1 Mark) |
| *fail* | Stepper | (1 Mark) |
| *sc* | Temporary | (1 Mark) |

## Section 2 – Software Requirements Analysis – CILO2 (20 Marks)

***Case study: Hotel Room Booking System***

*When an Adult Customer wants to book a hotel room through the online booking system, the conduct room booking use-case/function usually includes two use-cases/functions: the customer will request the desired type of rooms and pay for those rooms.*

*For the request room use-case, the customer should specify the room type (for example, deluxe room, superior room or standard room). What's more, IF the customer requests special requirements (for example, a kitchen, or a sea view), THEN the system should then book the room that meets the requirements.*

*After all the selections are completed (Note that the customer can choose multiple rooms), he/she may proceed to the CheckOut use-case, which will include (1) Calculate the total amount, and then (2) proceed to the Payment screen, where he/she will be given three options: (hint: use-case inheritance):*

a. ***PayByCreditCard**, the customer will enter the credit card information and pay by credit card, the payment will be processed by a **CardPaymentSystem***
b. ***PayByPayPal,** the customer will log in to PayPal and pay by PayPal, the payment will be validated by a **PayPalValidationSystem***
c. ***PayByWeChat**, the customer will scan a QR code and pay by WeChat, the payment will be validated by a **WeChatValidationSystem***

*In both two use-cases, IF any error occurs, THEN the system must be able to handle it via **HandleException**, there are three different error exceptions (hint: use-case inheritance):*

d. ***RoomUnavailable** (i.e., the room is unavailable for the selected date or time)*
e. ***TransactionAbort** (i.e., the customer chooses to cancel without completing the transaction)*
f. ***TimeOut** (i.e., the customer took too long to complete the transaction)*
g. ***OtherErrors** (i.e., this is to handle any other error not covered above)*

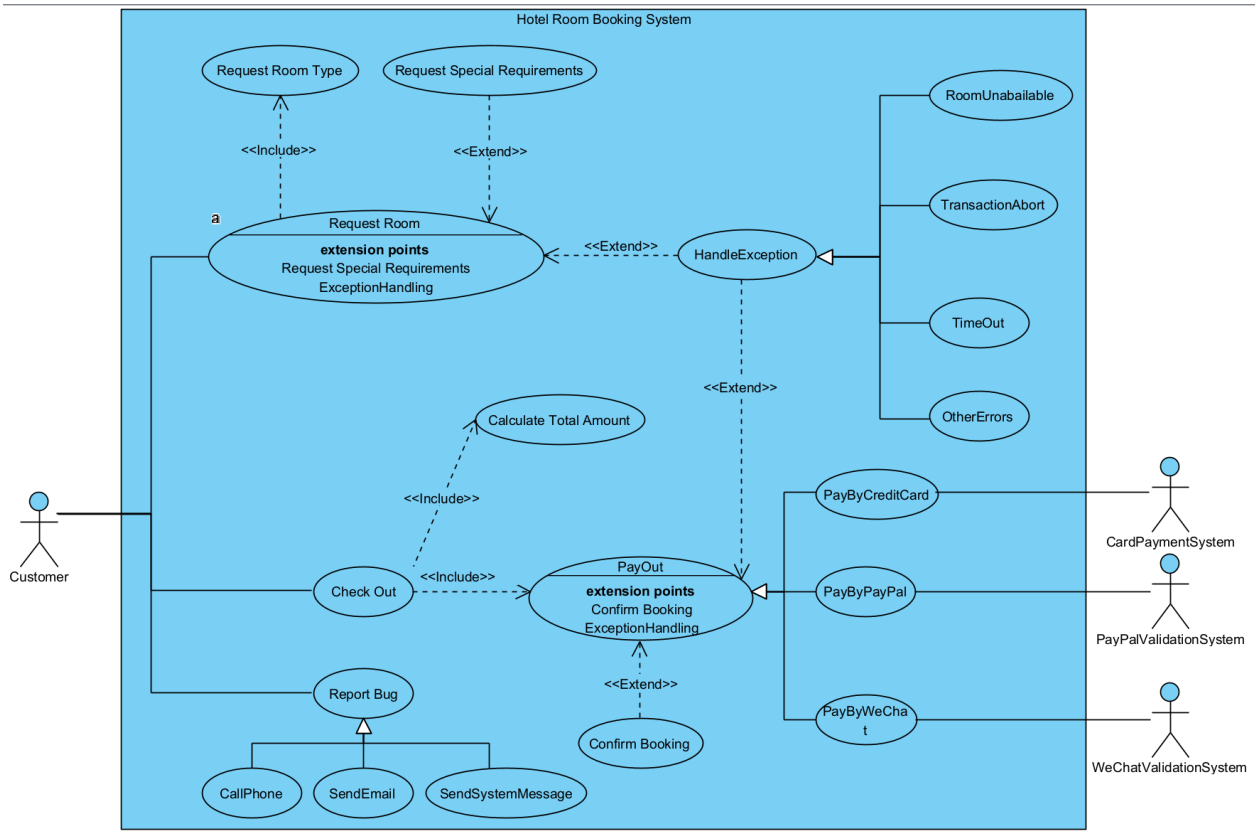
*Furthermore, if any other problems happen, the user should proceed to the ReportBug use-case, she/he can report the encountered errors/bugs via **CallPhone, SendEmail, or SendSystemMessage** method to track the problem. (hint: use-case inheritance)*

***IF the payment is successfully completed**, THEN it will **confirm the booking and send a confirmation email to the customer**. The customer can now check in the hotel with the confirmation number.*

**A. Draw a complete use case diagram (10 Marks)**

Based on the above, draw a use case diagram for Booking System. Whenever possible, your use case diagram **MUST** use **<<Extend>>** or **<<Include>>** to provide a good use case diagram.

*<Screen Capture: Place Use Case Diagram Here >*




**B. Requirements Specifications (10 Marks)**

Based on the same case study described above, complete the following table (from step 1) to describe the **Checkout** use case under typical course of events, and alternative course of events. The situation involves the

**Customer** actor, as well as external payment processing systems such as **Card**, **PayPal, and WeChat** validation systems.

| **Use Case Name:** | **Checkout** | |
|---|---|---|
| **Actor(s):** | **Customer, CardPaymentSystem, PayPalValidationSystem, WeChatValidationSystem** | |
| **Description:** | This use case describes the process of a customer completing the checkout for the room selected. On completion, the system will confirm the booking and send a confirmation email to the customer. | |
| **Reference ID:** | HK-ROOM-BOOKING-1.0 | |
| **Typical course of events:** | **Actor Action** | **System Response** |
| | **Step 1:** The customer successfully selects the room(s) and proceeds to checkout. | |
| | | **Step 2:** The system invokes the use case "Calculate Total Amount". |
| | | **Step 3:** The system invokes the use case "PayOut". |
| | **Step 4:** The customer chooses to pay by PayPal. | |
| | | **Step 5:** The system processes the payment through the PayPalValidationSystem. |
| | | **Step 6:** The system confirms the booking and sends a confirmation email to the customer. |
| **Alternative course of events:** | **Step 3a:** [Extension point: If the customer cancels the transaction, the system invokes the use case "AbortTransaction".]<br>**Step 3b:** [Extension point: If the customer takes too long to finish the transaction for more than a specific period, the system invokes the use case "TimeOut".]<br><br>**Step 4a:** The customer chooses to pay with a credit card instead of PayPal.<br>**Step 4b:** The customer chooses to pay with WeChat instead of PayPal. | |
| **Precondition:** | At least one room is selected to order | |
| **Postcondition:** | The system confirms the booking and sends a confirmation email to the customer | |

## Section 3. OO Modeling and Design Principles – CILO3 (20 Marks)

Study the following code fragments and answer the following questions.

```java
class Client {
  private String name;
  private String tel;
  private Profile profile;

   public Client(String name,  String tel) {
    this.name = name;
    this.tel = tel;
  }

  public void setProfile (String code,
                          String address,
                          String level) {
    profile = new Profile(code, address, level);
  }

  public Profile getProfile() {
    return profile;
  }
}
```

```java
class Membership extends Client {
  private int points;

  public Membership(String name,
                    String tel,
                    int pointNum) {
    super(name, tel);
    points = pointNum;
  }

  public void setPoint(int newPointNum) {
      points = newPointNum;
  }

  public int getPoint() {
    return  points;
  }
}
```

```java
class Profile {
  private String code;
  private String address;
  private String level;

  public Profile(String code, String address,
                 String level) {
    this.code = code;
    this.address = address;
    this.level = level;
  }

  public String toString() {
      return code  + address  + level;
  }
}
```

```java
class VIP extends Client {
  private float discount;

  public VIP(String name,
             String tel,
             float discountNum) {
    super(name,  tel);
    discount =  discountNum ;
  }

public void setDiscount(float newDiscountNum) {
     discount =  newDiscountNum ;
  }


  public float getDiscount() {
    return  discount;
  }
}
```

### A. Draw class diagram (5 Marks)

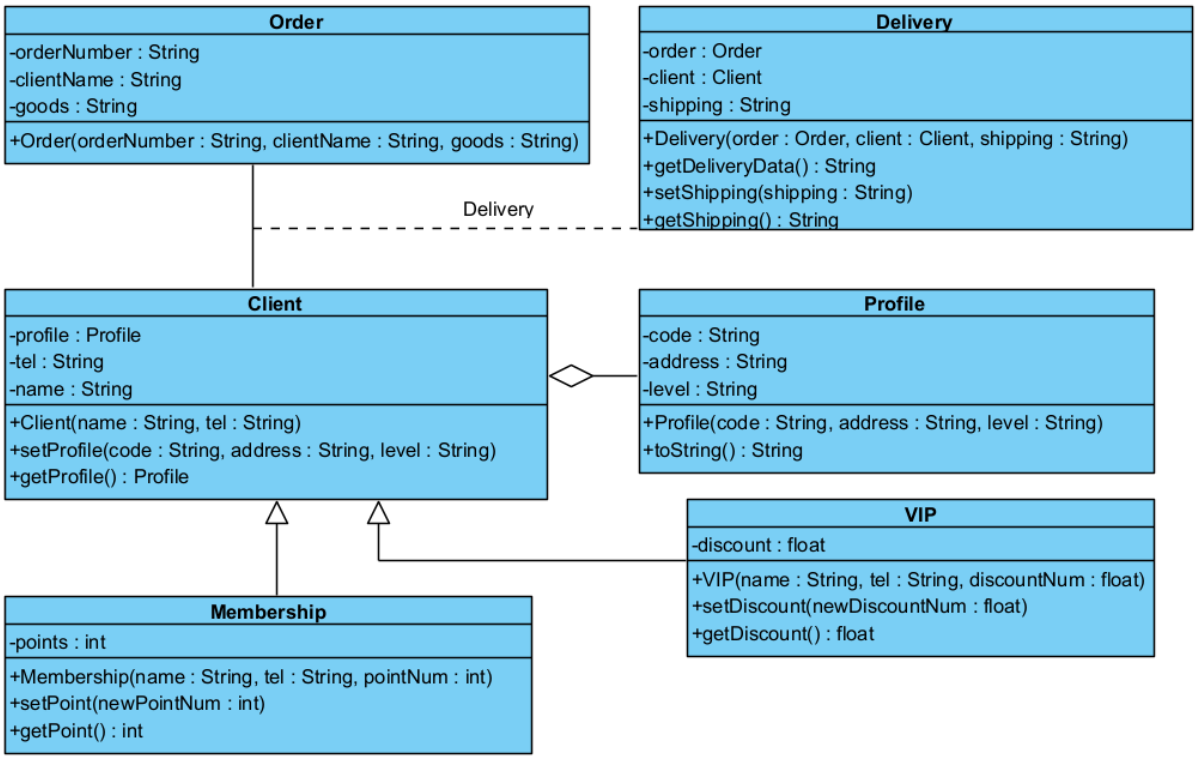
Based on your observation of the above code fragments, draw a complete class diagram to illustrate the class interactions and inheritance (hint: Class diagram captured from BlueJ will **NOT** be accepted, please use Visual Paradigm).

### B. Extend class design (5 Marks)

Assuming that there is a new year promotion event is organized for both membership and VIP, and there are many orders belonging to clients. **Extend** the existing class diagram design to correctly show classes **Order** and **Delivery**. Class **Order** should contain important attributes of order number, client Name and goods. Class **Delivery** should record which way of **Shipping** is adopted for the **Order**. So, it should contain an attribute of the delivery's **Shipping** in the delivery. (hint: you may classify the shipping as String in (B).)

Class Diagram for A+B:

*< Screen Capture: Place your diagram here>*




### C. Design refinement (10 Marks)

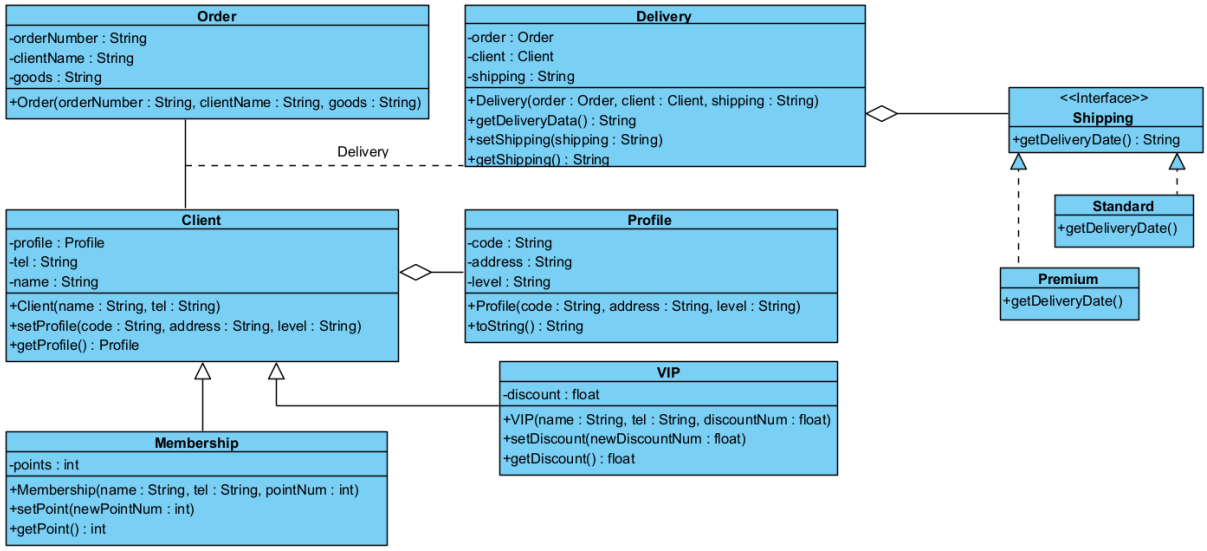
The delivery's Shipping is classified as *String* in (B), however, if we want to classify different Shipping with details, and more importantly each Shipping method will need to be able to perform different operations (e.g., **Standard** and **Premium** shipping method, and the *getDeliveryDate* operation may behave differently on different shipping methods), how would you modify the current design to allow multiple Roles using correct Object-Oriented Design? Please provide a short description on your design and justify your design choice.

Your solution/justification:

We could use a State Pattern to address this design issue, which allows multiple Roles to be used and selected by different event allocations.

Final Class Diagram:

*< Screen Capture: Place your diagram here>*




-- END --