PEC 1

Introduction to Software Engineering and Object Orientation

Alejandro Pérez Bueno

Oct 22, 2023

Table of Contents

Self-Responsibility Declaration

I understand that plagiarism, the use of AI or other generated content will imply that the delivered work will not be reviewed and it will be automatically assigned a grade of D. I certify that I have completed the CAT1 individually and only with the help that the professors of this subject considered appropriate, according to the FAQs about plagiarism.

Module 1 Questions

Question 1. Scrum vs Crystal Clear

a) Scrum or Crystal Clear in software development

My software development idea is creating a floating chat bubble inside a banking app that uses a Large Language Model (LLM) to answer customer's questions in a similar way to how ChatGPT answers.

> **Note**
>
> This model would be fine-tuned (re-trained) using this banking company's knowledge base.

Let's see when it would be best to use a *Crystal Clear* or a *Scrum* methodology.

First and foremost, as regards team size, it makes little difference using one methodology or the other, since both are meant for smaller teams of up to 8-9 members. However, there are other

On the one hand, using *Crystal Clear* would be useful at the beginning of the development of this project, as more frequent releases will be needed to set up the various services required to deploy the necessary architecture, namely a **cloud service** that hosts the new Web Application, a **specific LLM service** (that being OpenAI or some other alternative) with all its required prompting and configuration. All this setup will require small and incremental deliveries that will help reach a *Minimum Viable Product* (MVP) as quickly as possible.

Furthermore, a *Crystal Clear* approach will not require defining roles, which will ensure that the team will devote their time strictly to development and communication, which is essential to meeting deadlines.

On the other hand, it would be a better idea to use *Scrum* once the project has been deployed and is sufficiently tested. At this stage the team will need to focus on **maintaining the product**, making sure there are no bugs in the code, and **develop new features** based on the needs of customers and the company. At this stage, it will be required to define roles to organize the team according to the *Scrum* methodology, and at the same time there will be fewer deliveries required.

b) Real Example of Scrum or Crystal Clear

A real example of a company using *Crystal Clear* is the case of IBM (Lowe 2016), which as it turns out is the origin of this methodology. According to this article, Alistair Cockburn (who was working in the company) was asked to create a methodology that helped with a project in object-oriented programming. He interviewed several successful teams in the industry and found that, though they had no formal agile methodology in place, they all shared similar views. With all that information, Alistair developed and started promoting this new *Crystal* methodology.

Question 2. Project Types and Development Methods

Project 1. Aerospace agency

Because the objectives in this project are very clear and have to be carefully followed, I think that this is a type of project of **group 1**. As for the methodology, I would choose a **basic or waterfall life cycle** method, since it is stated that the project requires a high level of precision, meticulous planning and extensive documentation, which are typical of waterfall life cycle projects

Project 2. Image Editor

This is a clear example of **group 2**, as the company wants to change their initial ideas while developing something that satisfies all users with *empirical* data. A good approach for this project would be the **iterative and incremental life cycle** since it allows for testing smaller changes in the project. This is ideal for A/B testing, which is what the project wanted to achieve.

Project 3. WordPress Plugin

In this case, as before, I would consider this a project from **group 2** and would again choose an **iterative and incremental life cycle**, because this Engineer wants to develop something quickly, adding features progressively. He might change his initial plans over time, but wants to start small. He also does not want to spend too much time developing this project.

Question 3. Roles and Tasks

Plan and prioritize tasks for the next iteration of the project

- **Role**: Project Manager (PM)
- **Task**: Planning Tasks

The PM is responsible for distributing work and ordering each of the tasks in a project.

Create a system use case diagram

- **Role**: Functional Analyst
- **Task**: Execution tasks

The Functional Analyst is in charge of unifying the different views of the domain in a single model that is clear, concise and consistent. Even though it is not very technological, his duty of creating a system use case diagram requires a least some technical expertise.

Define the general structure of classes

- **Role**: Architect
- **Task**: Execution tasks

The Architect is the one who will have to design the actual code, ensuring it is adaptable to future needs, so he is key in the definition of the groundwork of defining the initial structure of the classes.

Define and perform extensive testing of a critical system component

- **Role**: Quality Expert
- **Task**: Monitoring and control tasks

The Quality Expert is the person in charge of testing the software to ensure that everything works as expected and as defined initially. Testing provides an external point of view that is crucial in the development of any piece of software.

Review code to ensure it is readable and complies with good practices

- **Role**: Programmer
- **Task**: Monitoring and control tasks

The Programmer not only develops the code to make the project work. This person is also responsible for making their own code readable and ensure it follows the necessary standards and good practices. The next person in the chain is the tester, who will just ensure the code works as expected, but won't ensure that the code is written properly.

Module 2 Questions

Question 4. Inheritance and Associations

Furniture **Class**

a) Subclasses

- *Couch*
- *Rug*

b) Association

The *Furniture* class would have an association with a class *Material*.

> Multiplicity
>
> A piece of furniture is made of one or more materials, and a material can make up multiple pieces of furniture

c) Attributes

- *Furniture*: *weight*
- *Couch*: *maxPeopleSitting*
- *Rug*: *shape*

d) Instances

Here's a sample `python` code to instantiate both classes:

```
couch1 = Couch(wegiht = 65, maxPeopleSitting = 4)
rug1 = Rug(weight = 25, shape = "round")
```

e) Explanation

In this case, we have defined a class for furniture and two pieces of furniture as classes for a couch and a rug. Every piece of furniture is made of one or more materials, and every piece has a specific weight. In the example, a couch has a maximum for the number of people that can fit at the same time on the couch, and the rug can have a specific shape, namely rectangle or round.

RealEstateProperty **Class**

a) Subclasses

- *residentialProperty*
- *rawLandProperty*

b) Association

The *RealEstateProperty* is associated to a class called *Neighborhood*

> Multiplicity
>
> A property may belong to a single neighborhood, but a given neighborhood may or may not have a real estate property.

c) Attributes

- *RealEstateProperty*: *landSquareFeet*
- *residentialProperty*: *yearBuilt*
- *rawLandProperty*: *isBuildable*

d) Instances

Here's a sample `python` code to instantiate both classes:

```
residential1 = residentialProperty(landSquareFeet = 15000, yearBuilt = "1979")
rawLand1 = rawLandProperty(landSquareFeet = 26000, isBuildable = True)
```

e) Explanation

In this example, we have created a real estate property class and two subclasses for residential property and raw land. A real estate property may belong to a neighborhood. A residential property will always have a year indicating when said property was initially built, and raw land may or may not be buildable.

Question 5. Visibility

- What visibility (`private`, `protected` or `public`) should have the attribute *name* of the class *Person* if we want it to be accessible by the `Person` class itself, but not visible by the instances of *City* or *Worker*?

The visibility should be **private** so that the attribute can only be accessed from itself, and no other class (not even subclasses) can see the attribute.

- What visibility (`private`, `protected` or `public`) should have the attribute *age* of the class *Person* so that it is accessible from the instances of its subclasses *Student* and *Worker*, but not from the instances of the class City?

The visibility should be **protected**. This way the attribute is hidden from outside that class or outside its subclasses.

- What visibility (`private`, `protected` or `public`) should have the attribute *population* of the class *City* if we want both instances of the class *City* as the *Student* and *Worker* have access to said attribute?

The attribute should be **public** in order to be accessed from outside the class or from other classes including *Student* or *Worker*.

- What visibility (`private`, `protected` or `public`) should have the attribute *levelStudies* of the class *Student* if we want both instances of the class *Student* like class *Person* have access to said attribute?

I am not sure I understood the question. If we want *levelStudies* to be accessible from both *Student* and *Person*, there is no direct answer. Here is a sample code in `java` that would allow both classes to access the attribute with some constraints explained below:

```
abstract class Person {                                                    ①
  public String name;
  public int age;
}

class Student extends Person {                                             ②
  public String levelStudies;
}

class Main {
  public static void main(String[] args) {
    Student alice = new Student();                                         ③
    Person bob = new Student();

    alice.levelStudies;                                                    ④
    ((Student)bob).levelStudies;
  }
}
```

① Create abstract class *Person*.
② Create subclass *Student* that extends *Person*.
③ Instantiate `alice` normally and `bob` as a polymorphic instance.
④ `alice` can access `levelStudies` from the *Main* since the attribute is **public**. `bob` cannot directly access `levelStudies` since it is a *Person*, not a *Student*. In order for `bob` to access `levelStudies`, a cast to *Student* must be performed. Only then can `bob` access the **public** attribute.

> **Note**
>
> That's how you would be able to access the `levelStudies` attribute from both classes.

- Propose an association between the classes *Worker* and *City*. Please also indicate the multiplicity of the association between both classes. Is it a polymorphic association?

A *Worker* can have an attribute named `belongsTo` that is actually of class *City*. A worker can belong to one city at a time, but a city can hold multiple workers at a time (association is **one to many**). This is not a polymorphic association.

- List the classes that are instantiable (that is, that can be created specific instances). If there is one that is not, could you explain why?

Student, *Worker* and *City* can be instantiated, as they are normal classes. *Person* cannot be instantiated since it is an **abstract class**, i.e. it is a template class for its subclasses.

Question 6. Online Bookstore App

a) List of classes and attributes

- **Book**: `concrete`
  - **Attributes**: `title`, `authors[]` (array), `isbn`, `stockAmount`
- **Customer**: `abstract`
  - **Attributes**: `name`, `email`, `password`, `address`, `receiveNewsletter` (bool)
- **Champion**: `concrete` (subclass of `Customer`)
  - **Attributes**: `lastPaymentDate`, `registrationDate`, `bankNumber`
- **Standard**: `concrete` (subclass of `Customer`)
  - **Attributes**: `N/A`
- **Order**: `concrete`
  - **Attributes**: `customer` (Customer), `books[]` (array), `fullDate`, `orderId`, `paymentMethod` (`"creditcard"/"cash"/"bizum"`)

b) List of associations

> Template
>
> (associationname): Associates each instance of (Class 1) with (none or one / one and only one / one or more than one / n or more than n / any number of) instances of (Class 2) and each instance of (Class 2) with (none or one / one and only one / one or more than one / n or more than n / any number of) instances of (Class1).

Here are the associations I found:

- **Book**:
  - Associates each instance of **Book** with any number of instances of **Order**.
- **Customer** (same applies for its subclasses **Champion** and **Standard**):
  - Associates each instance of **Customer** with any number of instances of **Order**.
- **Order**:
  - Associates each instance of **Order** with one instance of **Customer**
  - Associates each instance of **Order** with one or more than one instances of **Book**

References

Lowe, David. 2016. "What Is Crystal Clear?" Scrum & Kanban. April 25, 2016. https://scrumandkanban.co.uk/what-is-crystal-clear/.