EE308FZ-Final-Paper-21version

This docement is created by Lance, Laurent, Dupree.

If you have any problem, please feel free to contact with us: MIEC CLUB

---

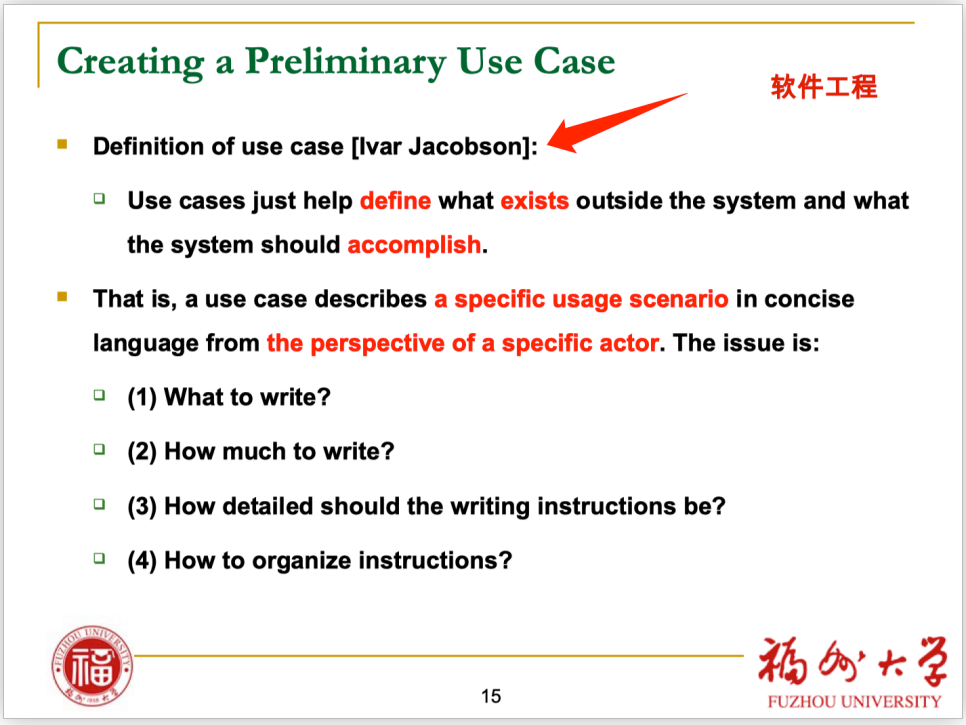
1 Question 1

1.1 Q1-(a) Briefly explain the following terms.

1.1.1 (i) Use Case

Google for use case:

1. A usage scenario for a piece of software;
2. A potential scenario in which a system receives an external request (such as user input) and responds to it.



A use case describes a specific usage scenario in **concise language** from the **perspective of a specific actor.**

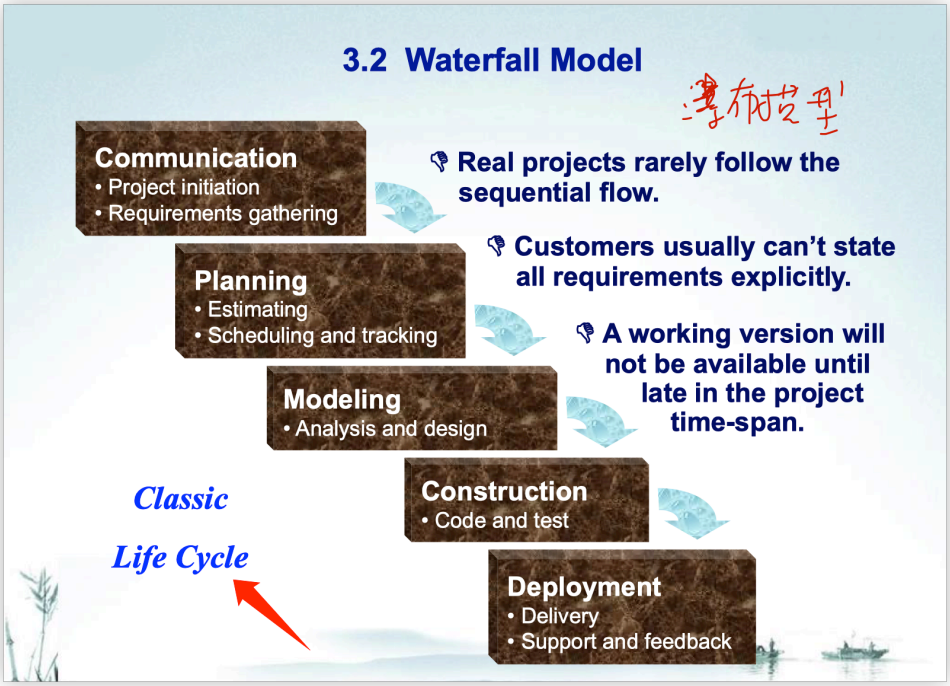
Use cases help define what exists outside the system and what the system should accomplish.

1.1.2 (ii) Software Development Life Cycle

系统发展生命周期

Google for SDLC:

Life cycle is a process for planning, creating, testing, and deploying an information system. There are usually six stages in this cycle: **requirement analysis, design, development and testing, implementation, documentation, and evaluation.**



Classic Life Cycle (Slides, 5 stages):

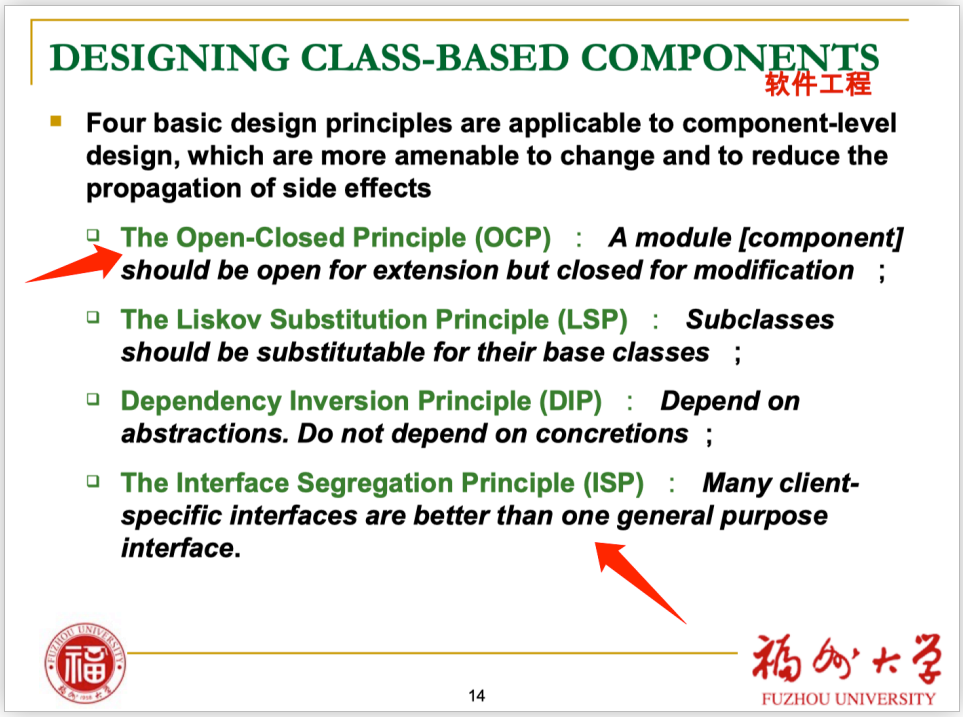
1. Communication
2. Planning
3. Modeling
4. Constrction
5. Deployment

1.1.3 (iii) Open-Closed Principle (OCP)

开闭原则: A module (component) should be open for extension but closed for modification.

[Google for OCP:]

Open–closed principle (OCP) states: software entities (classes, modules, functions, etc.) should be open for extension, but closed for modification.



1.1.4 (iv) Interface Segregation Principle (ISP)

Many clientspecific interfaces are better than one general purpose interface.

[Google for ISP:]

Interface Segregation Principle (ISP) states: no code should be forced to depend on methods it does not use.

补充：[SOLID: The First 5 Principles of Object Oriented Design]

Introduction

SOLID is an acronym for the first five object-oriented design (OOD) principles by Robert C. Martin (also known as Uncle Bob).

Note: While these principles can apply to various programming languages, the sample code contained in this article will use PHP.

These principles establish practices that lend to developing software with considerations for maintaining and extending as the project grows. Adopting these practices can also contribute to avoiding code smells, refactoring code, and Agile or Adaptive software development.

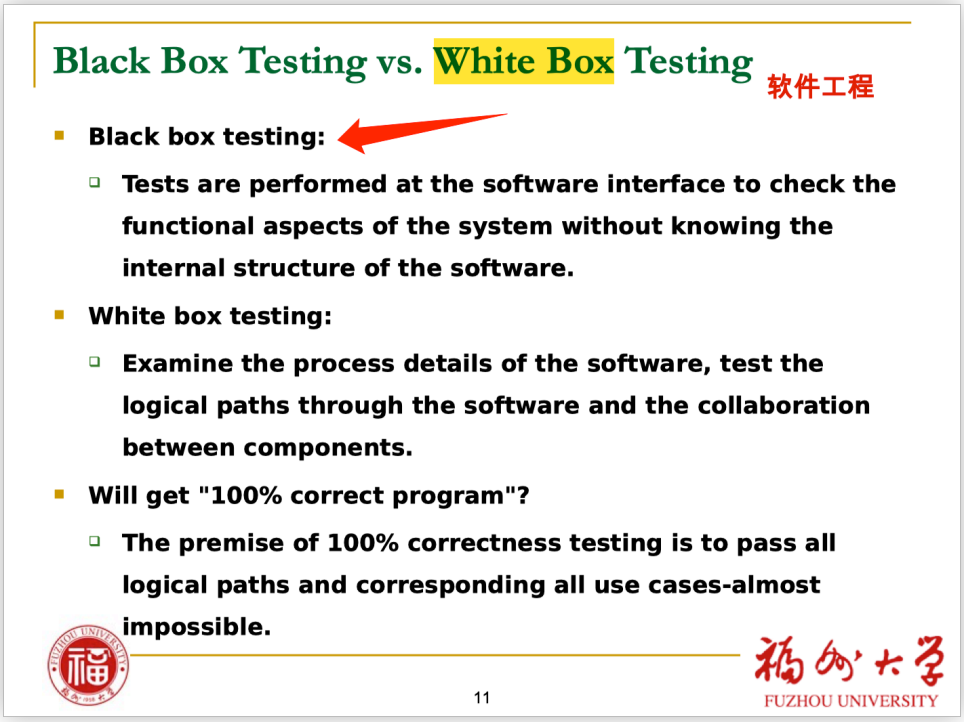
SOLID stands for:

- **S** - Single-responsiblity Principle
- **O** - Open-closed Principle
- **L** - Liskov Substitution Principle
- **I** - Interface Segregation Principle
- **D** - Dependency Inversion Principle

1.1.5 (v) White-Box Testing

白盒测试

[Google for White-Box]



White box testing: Examine the process details of the software, test the logical paths through the software and the collaboration between components.

检查软件的**过程细节**，测试软件的逻辑路径和组件之间的协作。

Black box testing: Tests are performed at the software interface to check the functional aspects of the system without knowing the internal structure of the software.

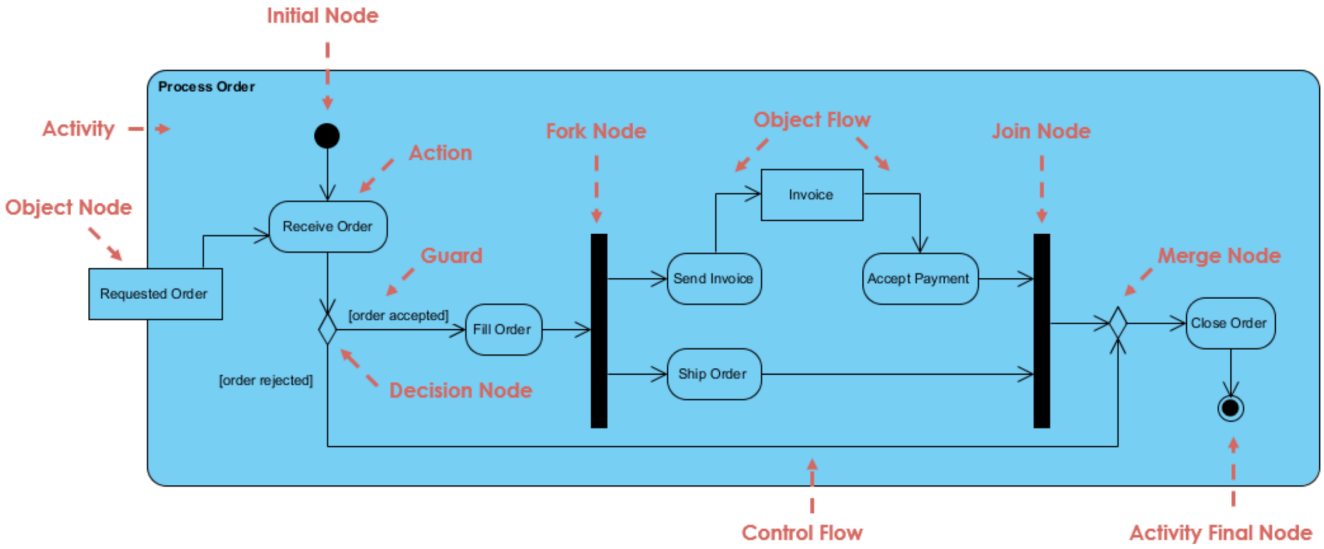
测试在软件界面执行，以检查系统的功能方面，**而不需要知道软件的内部结构**。

补充：[软件测试中文维基]

1.2 Q1-(b) Activity Diagram for order processing

(b) Prepare an activity diagram for order processing. A process of order processing is when we receive an order, we check each line item on the order to see if we have the goods in stock. If we do, we assign the goods to the order. If this assignment sends the quantity of those goods in stock below the reorder level, we reorder the goods. While we are doing this, we check to see if the payment is O.K. If the payment is O.K. and we have the goods in stock, we dispatch the order. If the payment is O.K. but we do not have the goods, we leave the order waiting. If the payment is not O.K., we cancel the order. (13 marks)

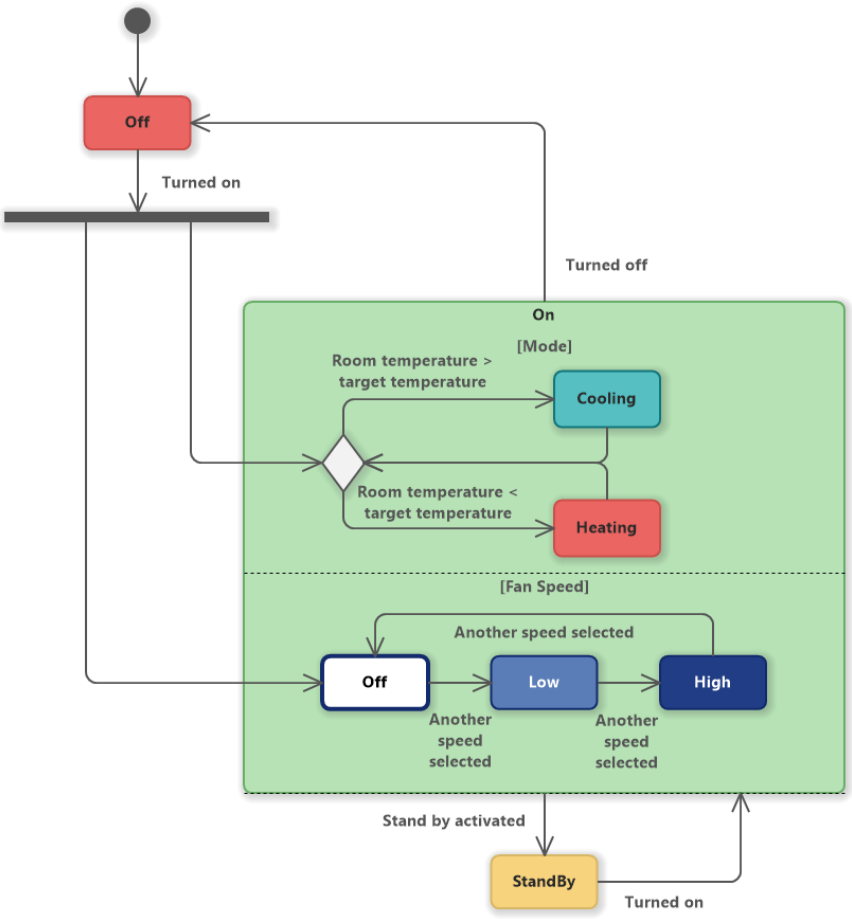
类似的案例（活动图）：



1.3 Q1-(c) State Diagram for an air conditioner

(c) Prepare a state diagram for an air conditioner. An air conditioner is in the mode of idle when not in use. When the temperature is high, the air conditioner is turned on from the mode idle to the mode cooling. When the temperature is very low, the air conditioner is turned on from the mode cooling to the mode heating. Your answer should clearly show how a change of operating mode is implemented by (i) a user request and (ii) a user defined period of inactivity. (10 marks)

更复杂的类似案例（状态图）：



The diagram describes these states:

- Off - activated by turning AC off, the first state
- On - activated by turning AC on
  - [Mode]
  - Cooling - activated when room temperature is higher than target temperature
  - Heating - activated when room temperature is lower than target temperature
  - [Fan Speed]
  - Off - fan speed button pressed
  - Low - fan speed button pressed
  - High - fan speed button pressed
- Stand By - is activated by pressing Stand By button

2 Question 2

QUESTION 2

Consider a course selection system on campus.

(a) List some actors that interact with a course selection system. Explain the relevance of each actor. (2 marks)

选课系统

(b) Prepare a use case diagram for a course selection system. Your diagram must explicitly include a 'select courses' use case. (6 marks)

(c) Prepare a normal scenario for each use case. (i.e. describe each use case in words) (12 marks)

(d) Prepare a sequence diagram for selecting courses in this system. (13 marks)

CSDN: 十四类UML图绘制汇总

选课系统UML及其分析

2.1 Q2-(a) List some actors

There will be teacher, student and system administrator

1. **Teacher**: could upload the course and change the content and delete the course, at the same time, teacher could check the list of student who select the course successfully.
2. **Student**: could look through the course, select the course and check the outcome of course selecting.
3. **System Administrator**: maintain the course and manage the account of students and teachers.

1.1.1学生的用例图描述

在本系统中，学生能够有查询课程信息并选课的功能。在查询课程信息的功能中，扩展了查询已选课程信息及检索课程的功能；在选课功能中，扩展了退课和调课的功能。其中，后两个用例与上一级用例间的关系是extended，即后面的用例用来增强前面的用例的功能。每个功能的具体描述如下表所示：

功能	描述
查询课程信息	包括查询已选课程信息及检索课程的功能。
查询已选课程信息	查看已选课程的详细信息如授课教师、授课时长等。
检索课程	学生通过输入课程名称、课程编号或教师名称查找相应课程。
选课	学生通过检索课程的页面，可以对心仪的课程进行选择操作。
退课	学生通过查询已选课程的页面，对选定的课程退课。
调课	学生通过查询已选课程的页面，输入要重新选择的课程编号，对选定的课程进行调课操作。

1.1.2教师的用例图描述

在本系统中，教师有申报课程及查看学生选课情况的功能。在申报课程的功能中，扩展了两个功能，分别是取消课程和删除课程。取消课程和删除课程是用来增强申报课程的功能。每个功能的具体描述如下表所示：

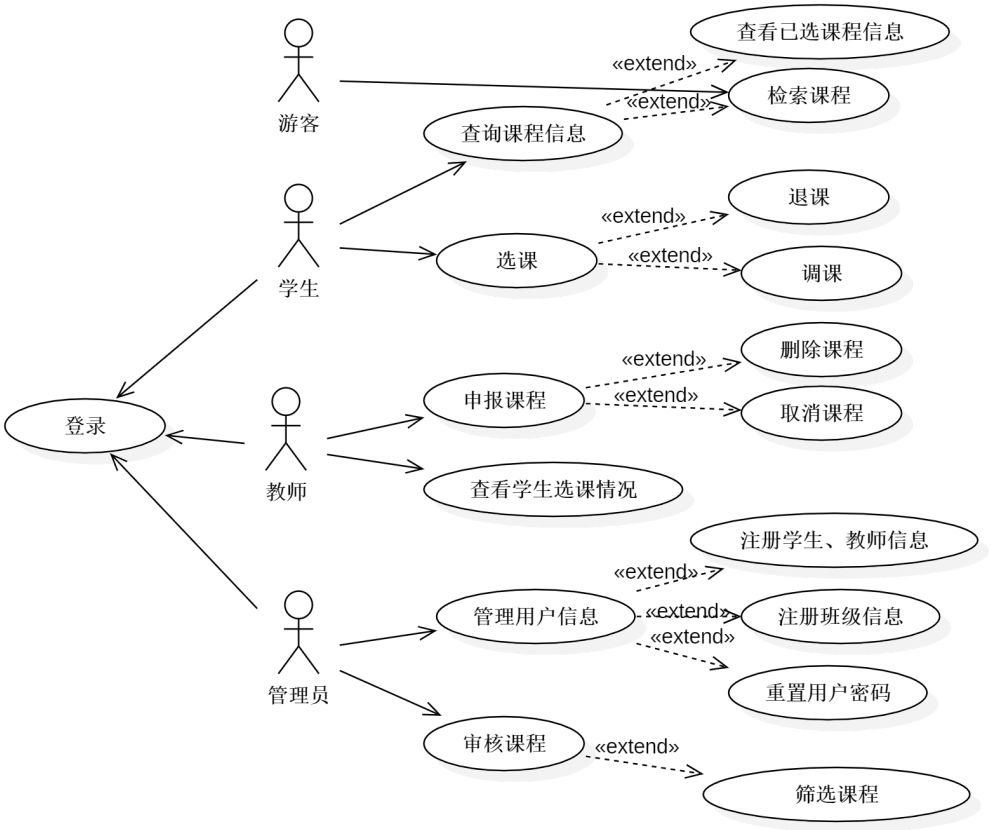
功能	描述
申报课程	教师通过此功能填写课程的详细情况，如课程名称、授课时长和课程简介等。并且，点击提交按钮后，能够 将信息传给管理员。
取消课程	教师可通过此功能将不再需要申报的课程取消申报。可以将此课程放入草稿相中。（是教师不想申报此课程，不是审核失败）
删除课程	教师通过此功能将审核失败的课程删除。
查看学生选课情况	教师通过此功能查看所有被审核过的课程的选课情况，比如选课人数、上课时间等。

1.1.3 管理员的用例图描述

在本系统中，管理员有管理用户信息和审核课程两大功能。在注册信息的功能中，扩展了注册学生、教师和班级的信息的功能和重置用户密码的功能。在审核课程的功能中，扩展了筛选课程的功能。每个功能的具体描述如下表所示：

功能	描述
注册学生、教师信息	管理员通过此功能注册学生和教师信息。
注册班级信息	管理员通过此功能注册班级信息。
重置用户密码	管理员可以对忘记密码的用户重置其密码。
审核课程	管理员通过此功能判断课程是否通过审核。
筛选课程	管理员通过筛选选课人数的功能，将选课人数不足的课程删除。

2.2 Q2-(b) Use Case Diagram

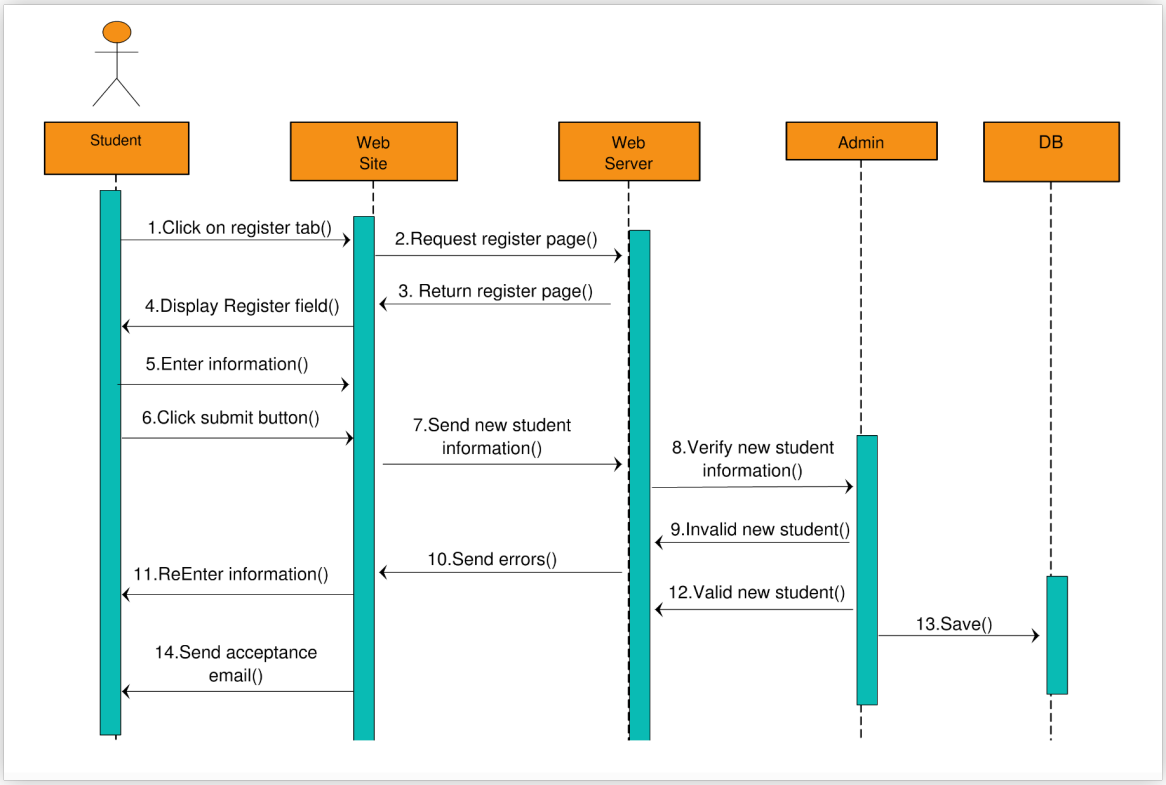


2.3 Q2-(c) Normal scenario for each use case

1. Teacher should be able to upload course, change the content of them and delete the course. They can also check the list of student who select the course successfully.
2. Student can look through the course list and select the course then check whether they successfully select the course
3. System administration can manage the courses to maintain the system.

2.4 Q2-(d) Sequence Diagram

类似的案例 (Student Registration System):



3 Question 3

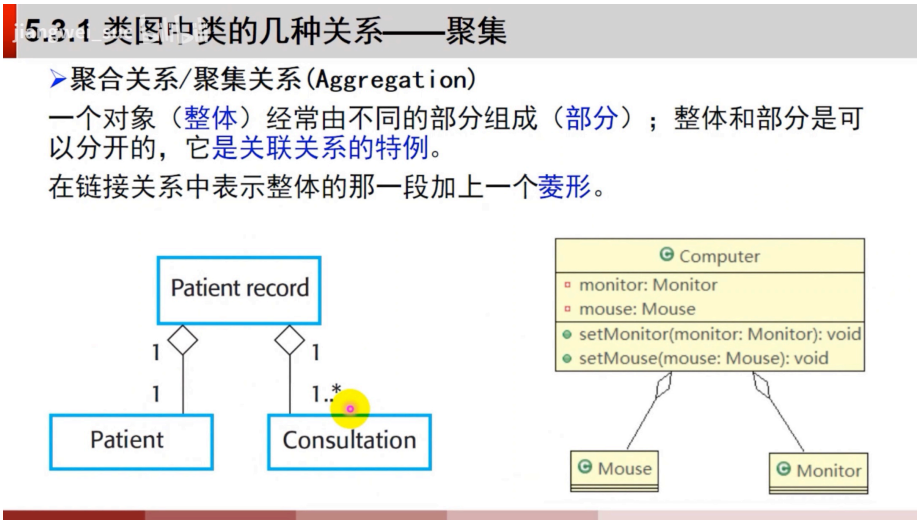
3.1 Q3-(a) Aggregation and Composition

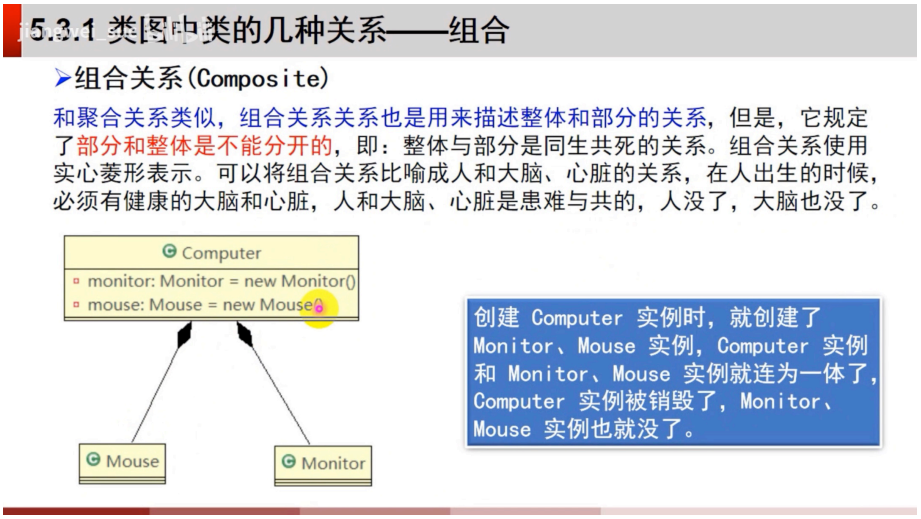
In a class diagram, there are different types of relationships. What is the difference between Aggregation and Composition? (4)

CSDN: 聚合和组合

两者区别:

- **依赖性区别 Dependency**
  - 聚合中的两种类（或实体）是可以单独存在的，不会相互影响；被关联的一方可以独立于关联一方，依赖性不强;
  - 相反，组合中的两个实体（或者类）是高度依赖于彼此的，它们之间会相互影响。
- **关系类型的区别 Relationship type**
  - 聚合代表了has-knowledge-of关系;
  - 组合代表了is-part-of关系。
- **关联强度的不同 Correlation intension** 耦合度(Coupling)
  - 聚合是一种弱关联关系;
  - 组合是一种强关联关系。
- **生命周期的不同 Life Cycle**
  - 在聚合关系中的两个类（或实体）的生命周期是不同步;
  - 但在组合关系中的两个类（或实体）的生命周期是同步的。





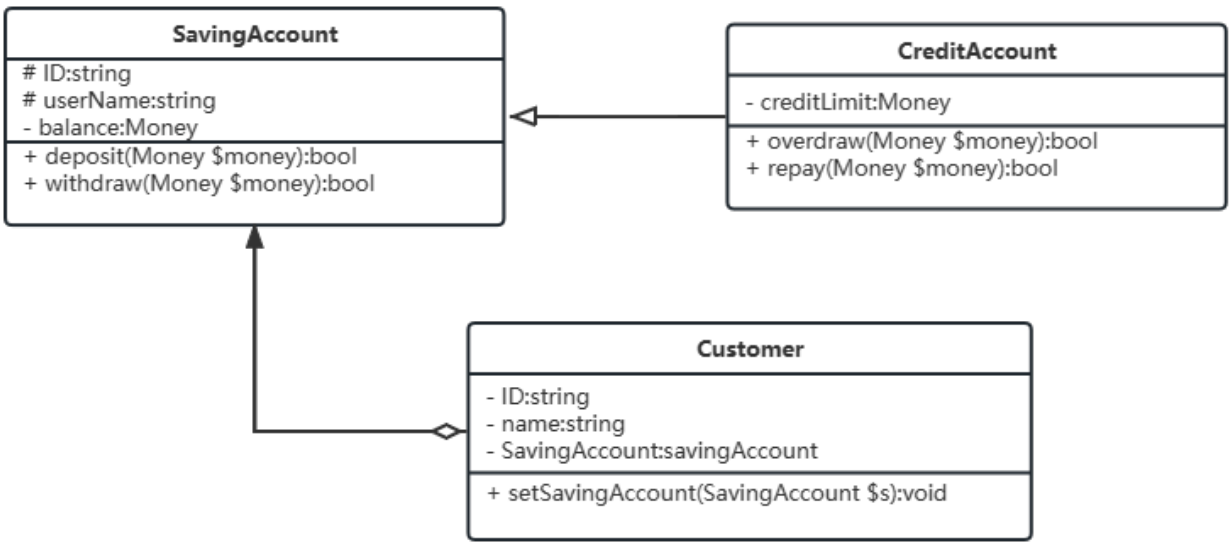
聚合关系和组合关系，都用来描述整体和部分，但是聚合关系中，整体和部分是相互独立的，对象（整体）消亡后，子对象（部分）并不会消亡；而组合则是“同生共死”的，不能分开。

3.2 Q3-(b) Code to Class Diagram

Prepare a class diagram corresponding to the following code. (12)

(b) Prepare a class diagram corresponding to the following code. 类图 (12 marks)

```
class SavingAccount {
protected:
   string ID;
   string userName;
private:
   Money balance;
public:
   bool deposit(Money money) { ... }
   bool withdraw(Money money) { ... }
};
class CreditAccount : public SavingAccount {
private:
   Money creditLimit;
public:
   bool overdraw(Money money) { ... }
   bool repay(Money money) { ... }
};
class Customer {
private:
   string ID;
   string name;
   SavingAccount savingAccount;
public:
   void setSavingAccount(SavingAccount s) {
      this->savingAccount = s;
   }
};
```

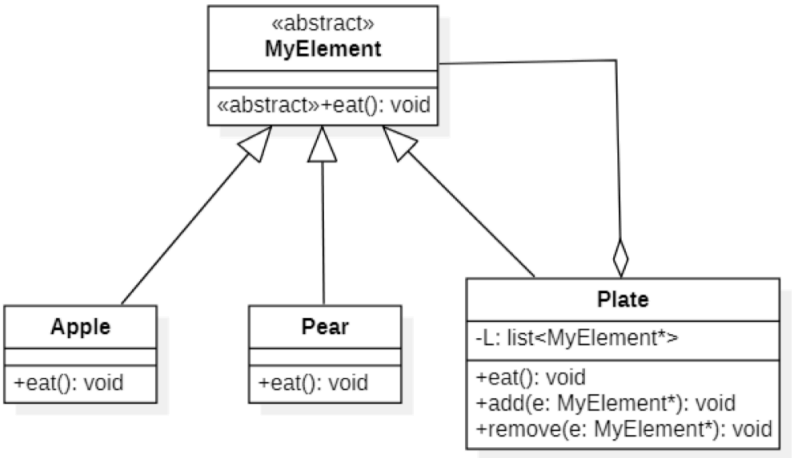


3.3 Q3-(c) Class diagram to Code

Write the skeleton code corresponding to this class diagram in C++, Java, Python, or C#. (17)

提醒：2022年明确不考代码题。

(c) Write the skeleton code corresponding to this class diagram in C++, Java, Python, or C#. (17 marks)




```
abstract class MyElement {
  abstract public void eat();
}

public class Apple extends MyElement{
    @Override
    public void eat() {

    }
}
```

```
public class Pear extends MyElement{
    @Override
    public void eat() {

    }
}

public class Plate extends MyElement{
    List<MyElement> L;
    @Override
    public void eat() {

    }
    public void add(MyElement e){

    }
    public void remove(MyElement e){

    }
}
```

最后的最后，祝你好运!

Lance, Laurent, Dupree于2023/01/02.