

0422

14:10-16:00

System Analysis & Design

软件系统分析与设计

System Analysis & Design

M210007B [02]

Jingxin Su

Sunday, April 21, 2024

面向对象

Object-Oriented

- 建模系统功能
- 发现并确定业务对象
- 组织对象并确定其关系

对象建模的历史

History of Object Modeling

- 1989年到1994年，面向对象建模语言从不到10种增加到了50多种
- 不同的建模语言具有不同的建模符号体系，妨碍了软件设计人员、开发人员和用户之间的交流。有必要建立一个标准的、统一的建模语言
- 统一建模语言UML的诞生结束了符号方面的“方法大战”
- OOA (*Object-Oriented Analysis*) / OOD (*Object-Oriented Design*)方法
- OMT(*Object Modeling Technique*)方法
- Booch (*Booch Method*)方法
- OOSE (*Object-Oriented Software Engineering*)方法



对象建模

Concept

对象

对象、属性、方法和封装

类

类、泛化和特化

关系

对象类关系、多重性

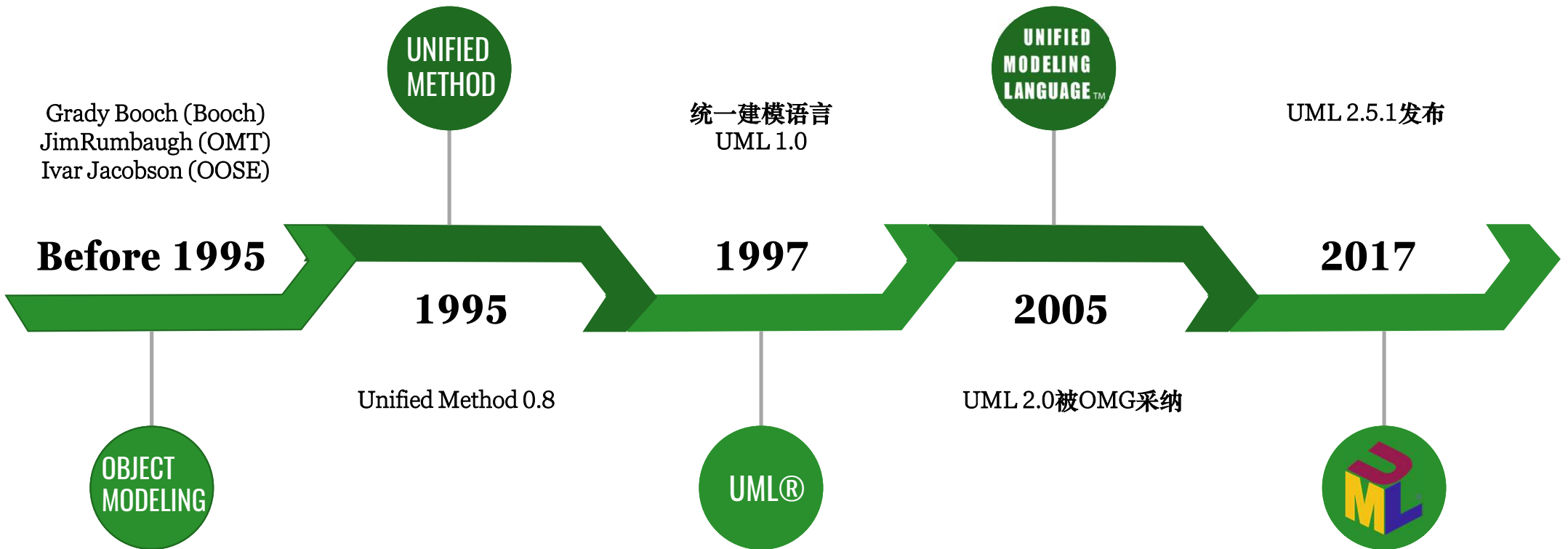
消息

消息和消息通信

识别系统环境中的对象并识别这些对象之间的关系。

UML: 起源

The Origin and History of UML



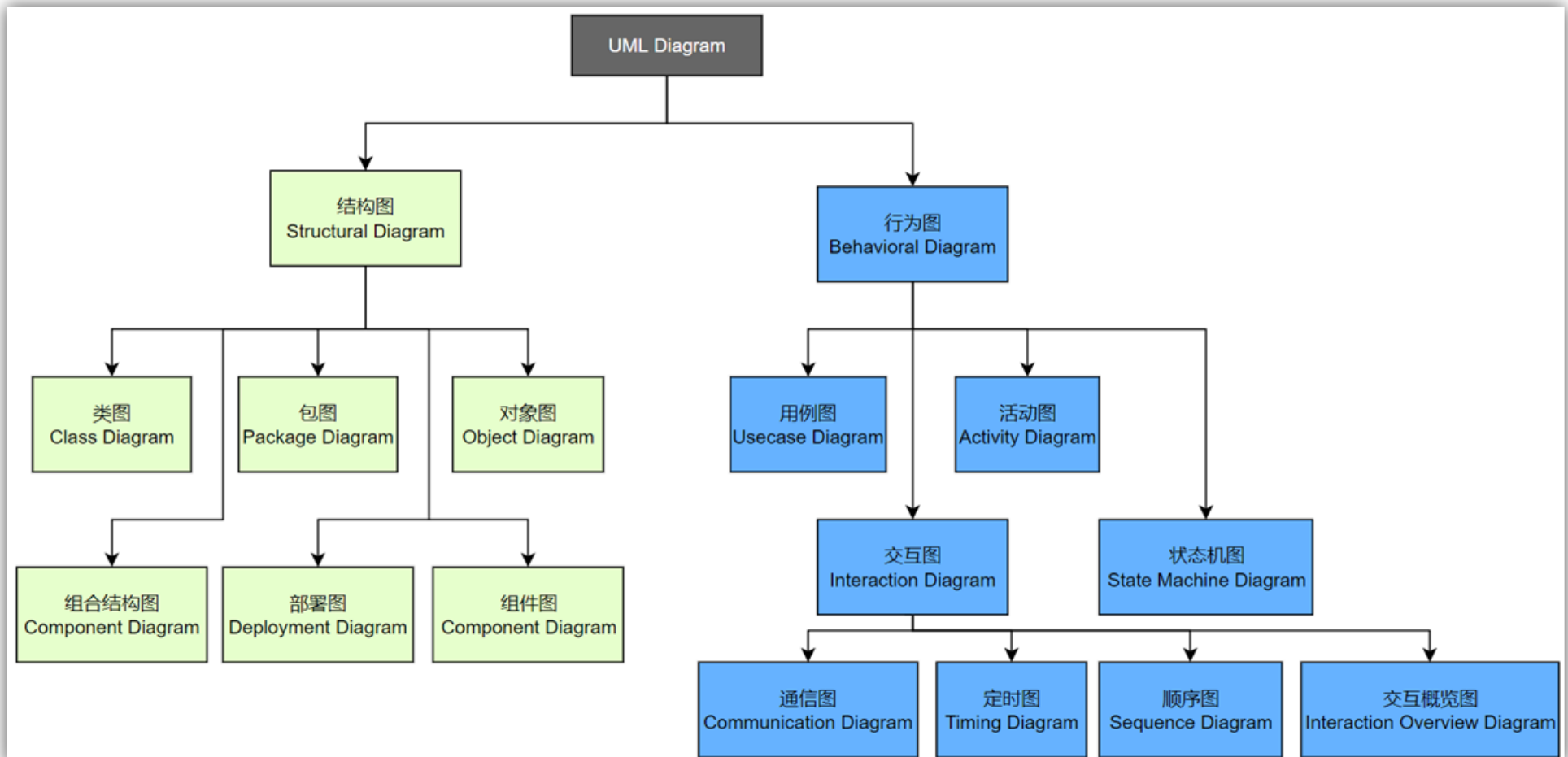
统一建模语言概念

UML Concepts

- 提供业务问题或整个系统的可视化模型
 - 通过使用UML图，有利于交流与协同工作
- 规约软件系统的产物
 - 规约 (Specifying) 意味着建立的模型是准确的、无歧义的、完整的
- 构造软件系统的产物
 - 前向工程:从UML模型生成编程语言代码的过程
 - 逆向工程:从代码实现生成UML模型的过程
- 建立软件系统的文档
 - UML可以为系统的体系结构及其所有细节建立文档

UML图

UML Diagrams



用例建模

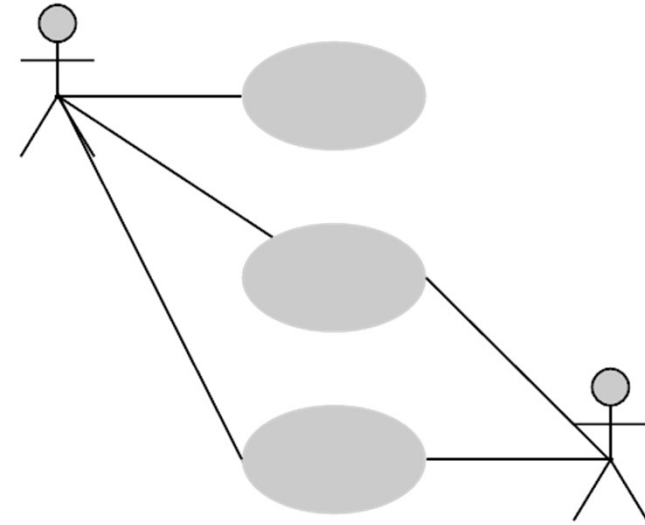
Use Case Modeling

- **以用户为中心的开发：**

- 基于理解利益相关者的需求以及系统开发原因的开发过程

- **用例建模：**

- 使用业务事件、发起业务事件的人，以及系统如何响应这些事件来建模系统功能的过程
- 促进并鼓励了用户参与，是确保项目成功的关键因素之一
- 产物：**用例图与用例描述**



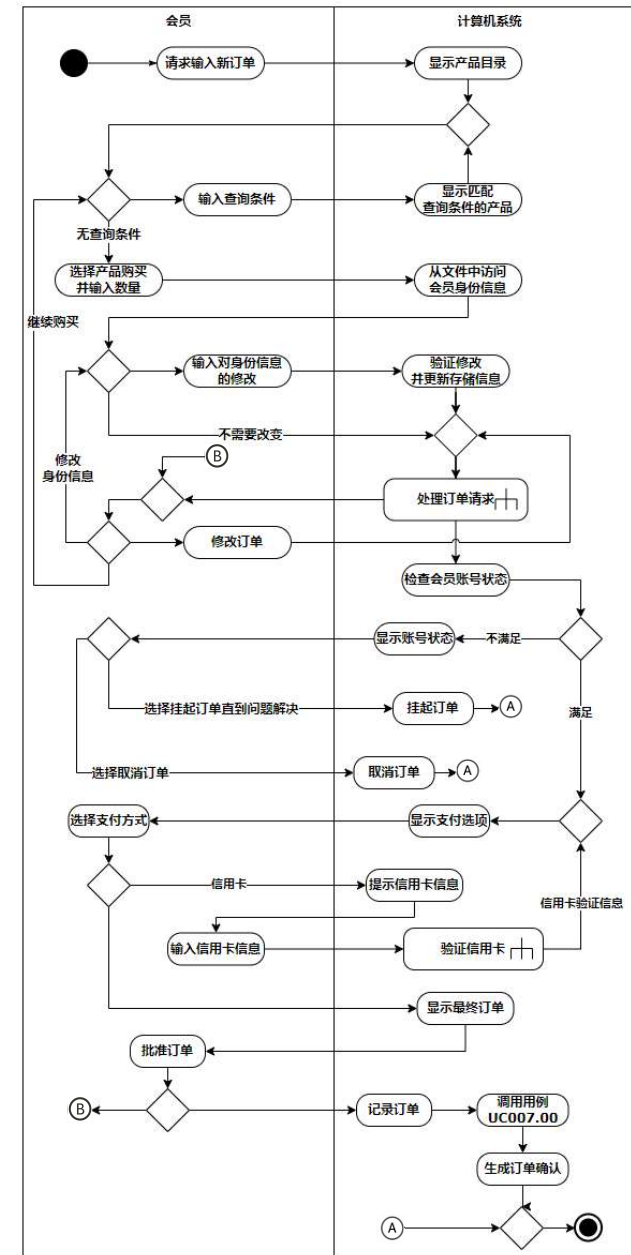
建模用例活动

Modeling the Use-Case Activities

● 活动图(Activity Diagram):

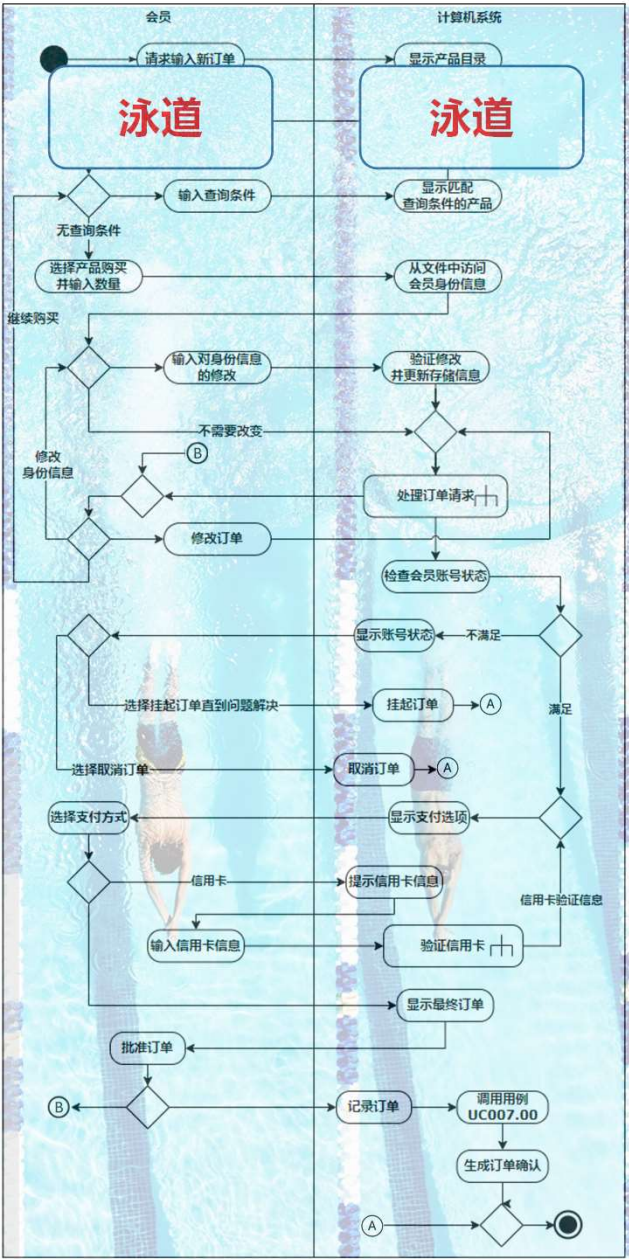
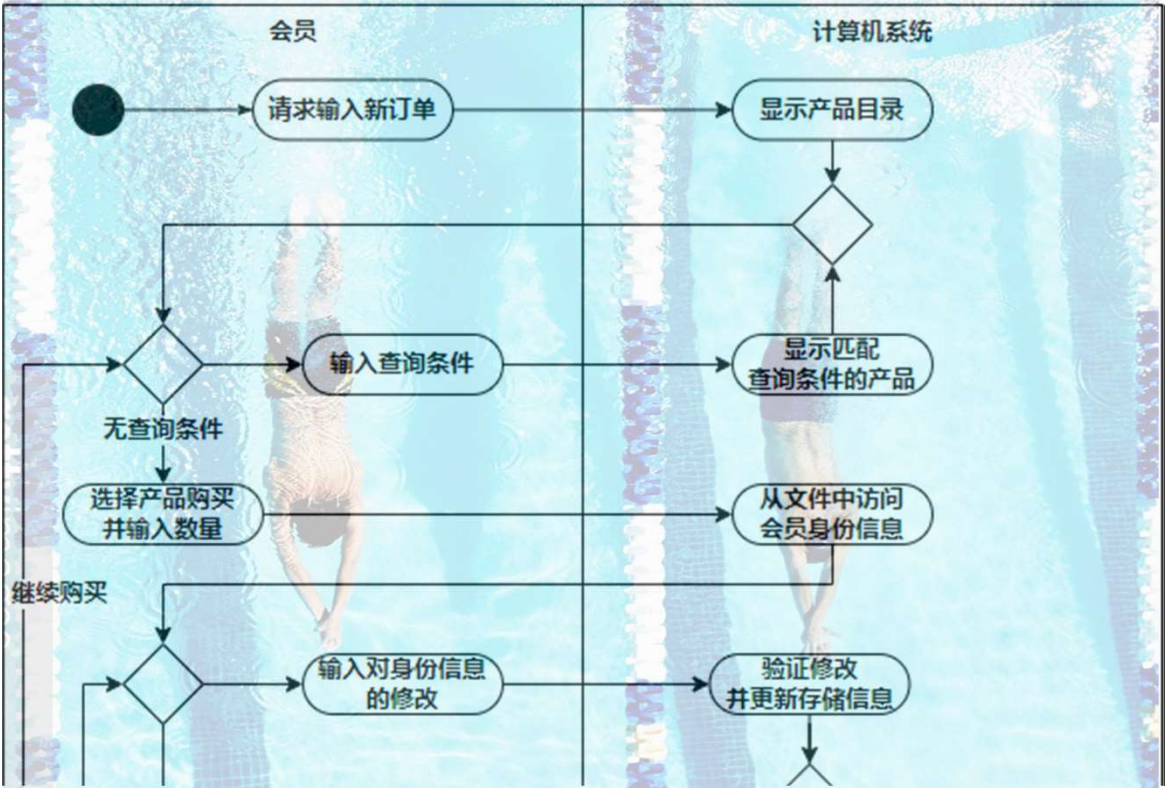
用于以图形方式描述业务过程或用例的活动的顺序流程。

- 提供了描述并行活动的机制
- 每个用例至少可以构造一个活动图
- **只对**具有复杂逻辑的用例(或用例的一个片段)绘制活动图
- 更好地理解用例步骤的流程和顺序



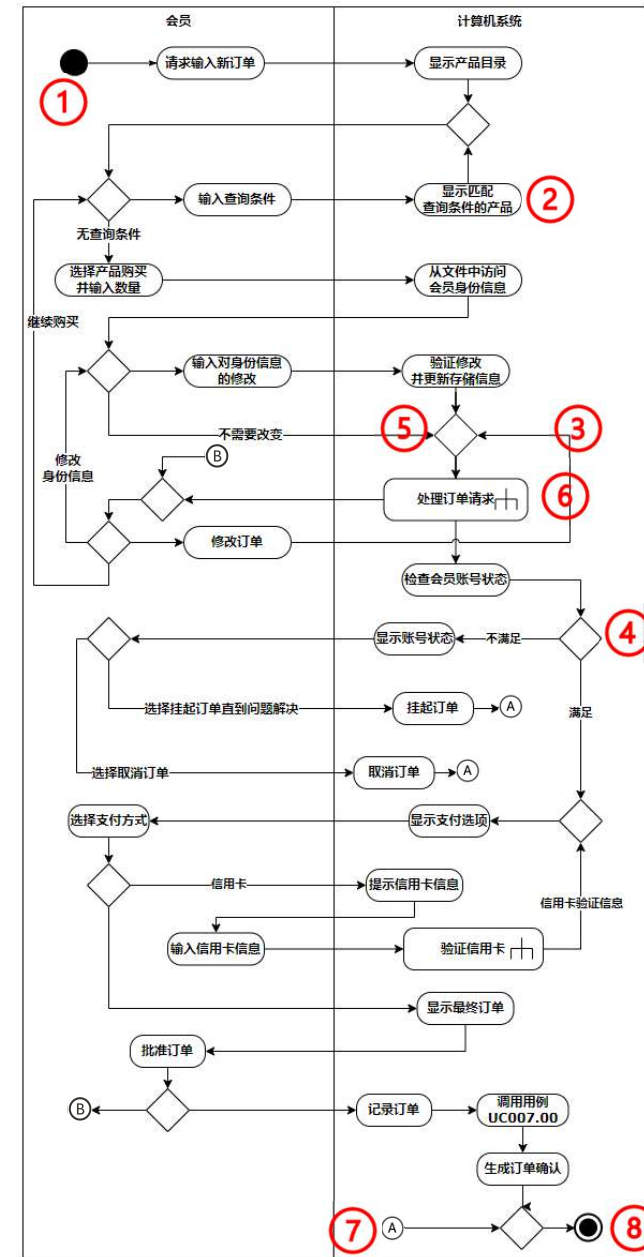
活动图 Activity Diagram

泳道(Swimlane) — 按照特定类或角色执行的工作分割活动图



活动图 Activity Diagram

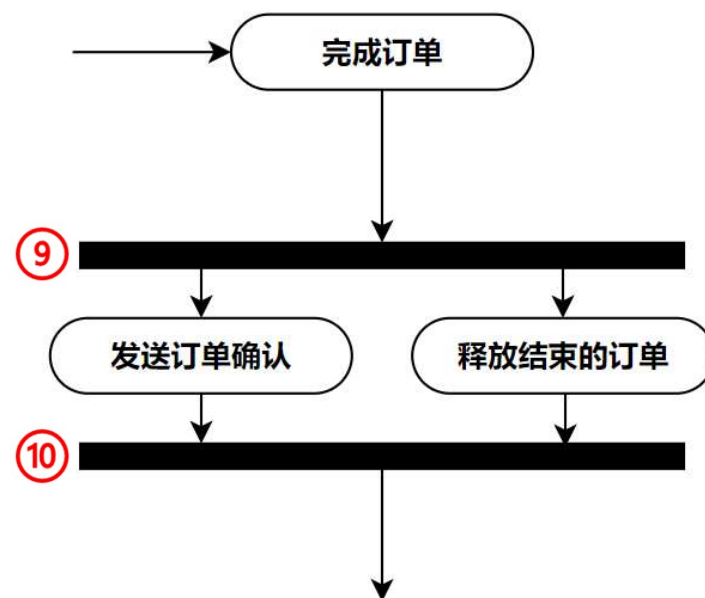
- ① 初始节点(Initial Node)—实心圆表示过程的开始
- ② 动作(Action)—圆角矩形表示单个步骤。动作的序列构成了图形描述的活动
- ③ 流(Flow)—图上的箭头指示通过动作的进展
- ④ 决策(Decision)—具有一个进入流和两个或多个输出流的菱形。输出流被标记以指示条件
- ⑤ 合并(Merge)—具有两个或多个进入流和一个输出流的菱形
- ⑥ 分支(Fork)—具有一个进入流和两个或多个输出流的黑条
- ⑦ 联合(Join)—具有两个或多个进入流和一个输出流的黑条
- ⑧ 活动终止(Activity Final)—空心圆内的实心圆表示过程的结束



活动图



- ⑨ 分支(Fork)—具有一个进入流和两个或多个输出流的黑条
- ⑩ 联合(Join)—具有两个或多个进入流和一个输出流的黑条

Activity Diagram



活动图 VS 用例描述

Activity Diagram VS Use Case Narratives

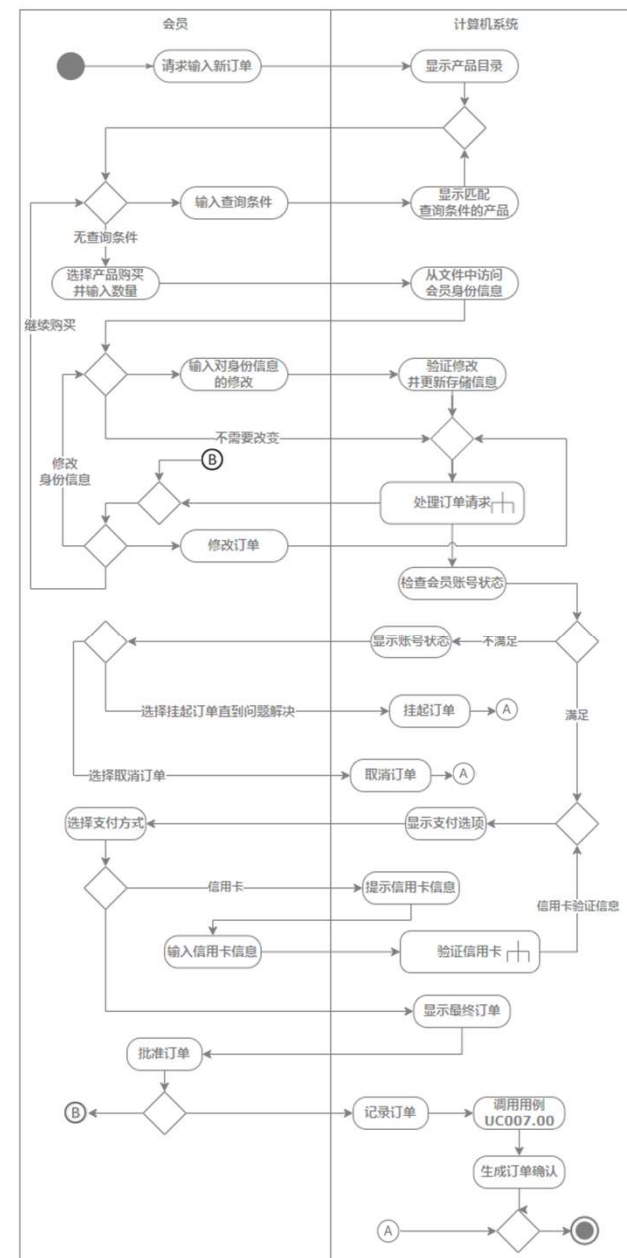
优/缺点	活动图	用例描述
	<ul style="list-style-type: none">▸ 可用于系统分析和设计▸ 可以保证复杂过程的完整性和有效性	<ul style="list-style-type: none">▸ 利益相关者易于理解▸ 映射到每个用例并且易于管理
	<ul style="list-style-type: none">▸ 对利益相关者不友好	<ul style="list-style-type: none">▸ 在冗长的文本叙述中难以确保过程逻辑的完整性和有效性

♥ 用活动图补充用例叙述

构造活动图指南

Guidelines for Constructing Activity Diagrams

- ① 从一个作为起点的**初始节点**开始
- ② 如果它们与你的分析有关则增加**分割(泳道)**
- ③ 为用例的每个主要步骤添加一个**动作**
- ④ 从一个活动到另一个活动、决策点或终点添加一条**流**。
每个动作应该只有一个输入流和一个输出流
- ⑤ 在流分解成不同路线的地方添加**决策**。确保用一个**合并**将各个流重新合并
- ⑥ 在并行执行活动的地方添加**分支**和**联合**
- ⑦ 用一个**单一的**活动终止符号结束





案例分析: 学生信息管理系统

Case Study: Student Information System



用例描述

Use Case Narratives

SAD-001: 申请课程

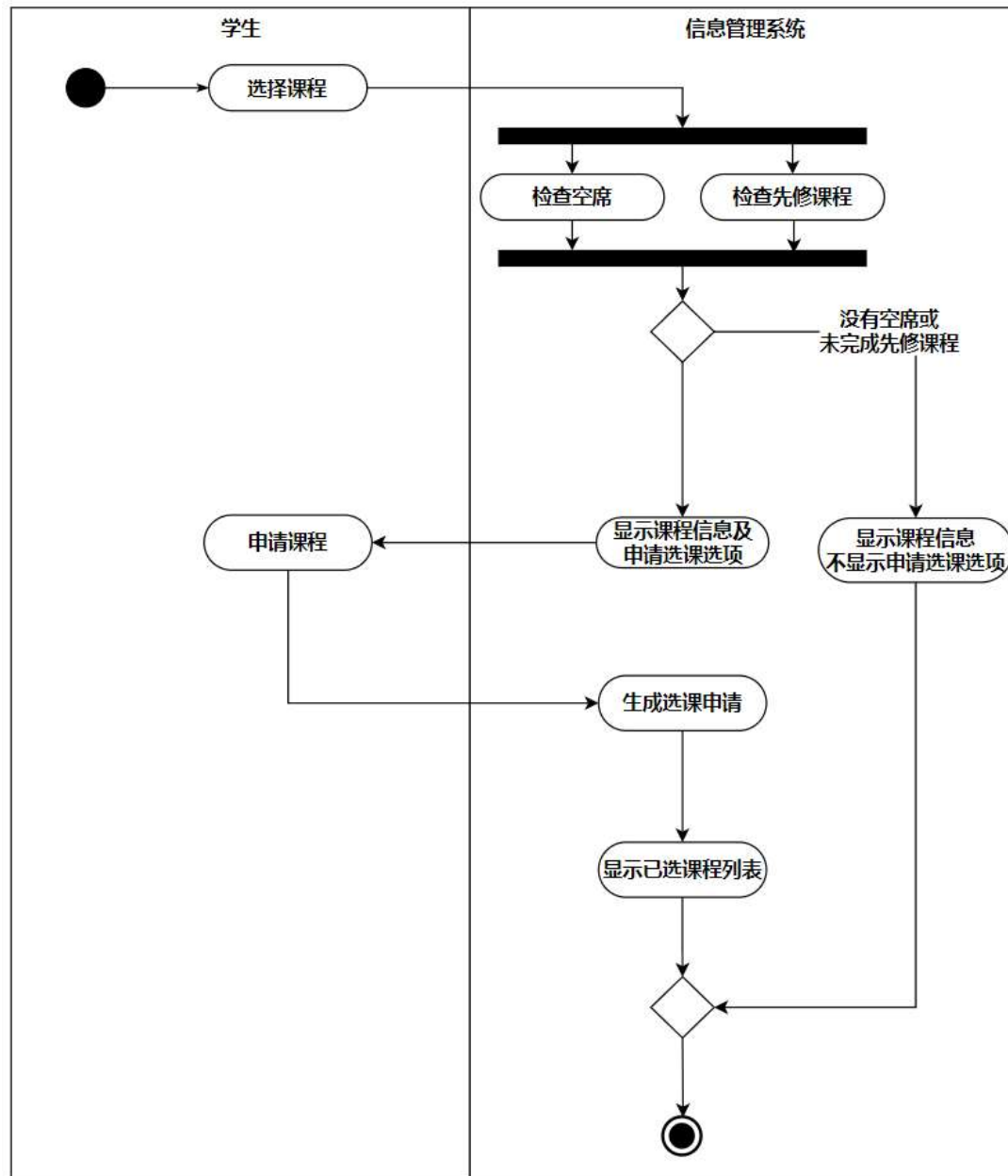
Use Case ID	SAD-001	
用例名	申请课程	
描述	<p>这个用例描述了学生通过系统申请课程的事件。学生从开放课程列表中选择一门课程。学生可以获得课程信息、剩余可选数以及是否完成了预先要求的课程。一旦学生提交申请，创建一个申请并等待批准。</p>	
优先级	高	
主要参与者	学生	
前置条件	学生必须登录系统进行选课	
触发器	当学生选择申请课程时启动用例	
典型事件过程	参与者动作	系统响应
	<p>第1步： 学生从打开的课程列表中选择一门课程</p> <p>第4步： 学生浏览课程信息并选择申请</p>	<p>第2步： 系统响应检查课程是否有空席，并检查学生是否完成所选课程的所有先修课程</p> <p>第3步： 系统向学生显示课程信息供学生申请</p> <p>第5步： 系统生成新课程申请</p> <p>第6步： 系统在新页面显示学生课程申请列表</p>
替代事件过程	<p>替代第3步：课程没有空席或没有完成所有预修课程，系统显示课程信息，但学生无法申请课程。</p> <p>替代第4步：学生浏览课程信息。终止用例。</p>	
结论	当学生收到生成应用程序的确认信息时，此用例就结束了。	
后置条件	申请已生成并等待批准。	
业务规则	学生不是休学状态。	
实现约束和说明	<ul style="list-style-type: none">- 用例必须24 * 7小时在课程申请期间提供给学生。- 频率：预计此用例每天将执行 10,000 次。它应该支持多达 50 名学生的并发量。	
假设	<ul style="list-style-type: none">- 开放课程列表中的课程可供申请- 学生可以在获得批准之前取消申请	
开放问题	N/A	



活动图 Activity Diagram

SAD-001: 申请课程

典型事件过程	参与者动作	系统响应
	第1步： 学生从打开的课程列表中选择一门课程 第4步： 学生浏览课程信息并选择申请	第2步： 系统响应检查课程是否有空席，并检查学生是否完成所选课程的所有先修课程 第3步： 系统向学生显示课程信息供学生申请 第5步： 系统生成新课程申请 第6步： 系统在新页面显示学生课程申请列表
替代事件过程	替代第3步：课程没有空席或没有完成所有预修课程，系统显示课程信息，但学生无法申请课程。 替代第4步：学生浏览课程信息。终止用例。	



作业其二

Homework-2

Q1. 完成“报销系统”的用例图

Q2. 完成用例“编辑费用报销报告”的用例描述

Q3. 使用活动图描述“费用报销”的全过程

想象你是一名项目经理，正在负责霍格沃茨学校“报销系统”的业务分析。以下是一个关于“费用报销”的用户故事：

作为霍格沃茨学校的员工，我可以为工作中所产生的办公、差旅等费用申请报销，以便我不必个人为此类活动付费。

这个“费用报销”的用户故事包括以下步骤：

- 员工提交费用报销报告
- 员工的经理审核并批准/拒绝报告
- 财务主管审核并批准/拒绝报告
- 银行向员工的银行账户付款
- 每次报告被拒绝时，员工可以编辑更新并重新提交报告。审查过程从他/她的经理重新开始。



练习

Q3. 使用活动图描述“费用报销”的全过程

作为霍格沃茨学校的员工，我可以为工作中所产生的办公、差旅等费用申请报销，以便我不必个人为此类活动付费。

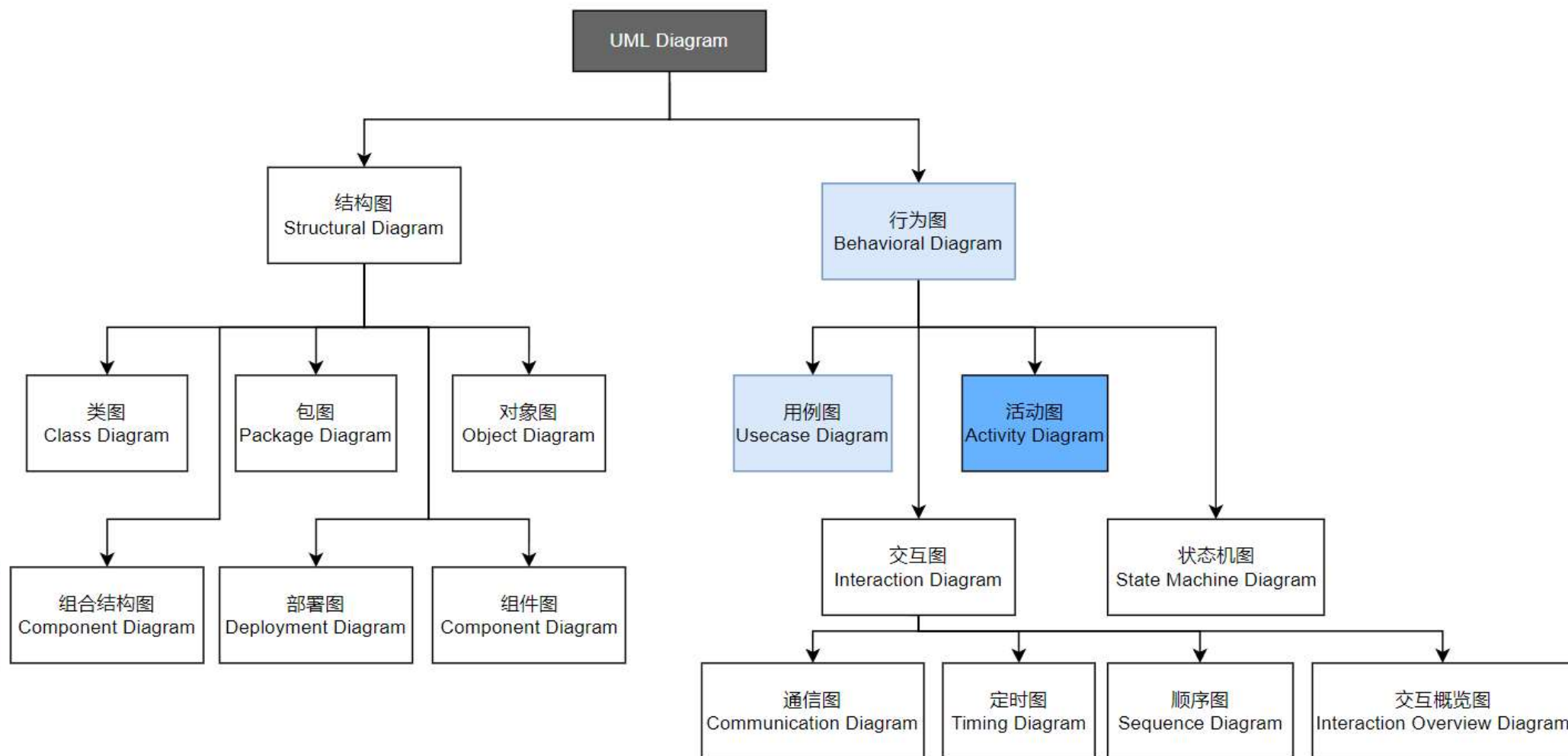
这个“费用报销”的用户故事包括以下步骤：

- ▶ 员工提交费用报销报告
- ▶ 员工的经理审核并批准/拒绝报告
- ▶ 财务主管审核并批准/拒绝报告
- ▶ 银行向员工的银行账户付款
- ▶ 每次报告被拒绝时，员工可以编辑更新并重新提交报告。审查过程从他/她的经理重新开始。

员工	经理	财务主管	银行
			

UML图

UML Diagrams



建模系统功能

Modeling the Functions of the System

✓ 建模系统的**功能性描述**

– 基于需求使用**用例建模**所有业务事件

✓ 构造**分析**用例模型

– 识别、定义并记录参与者和用例

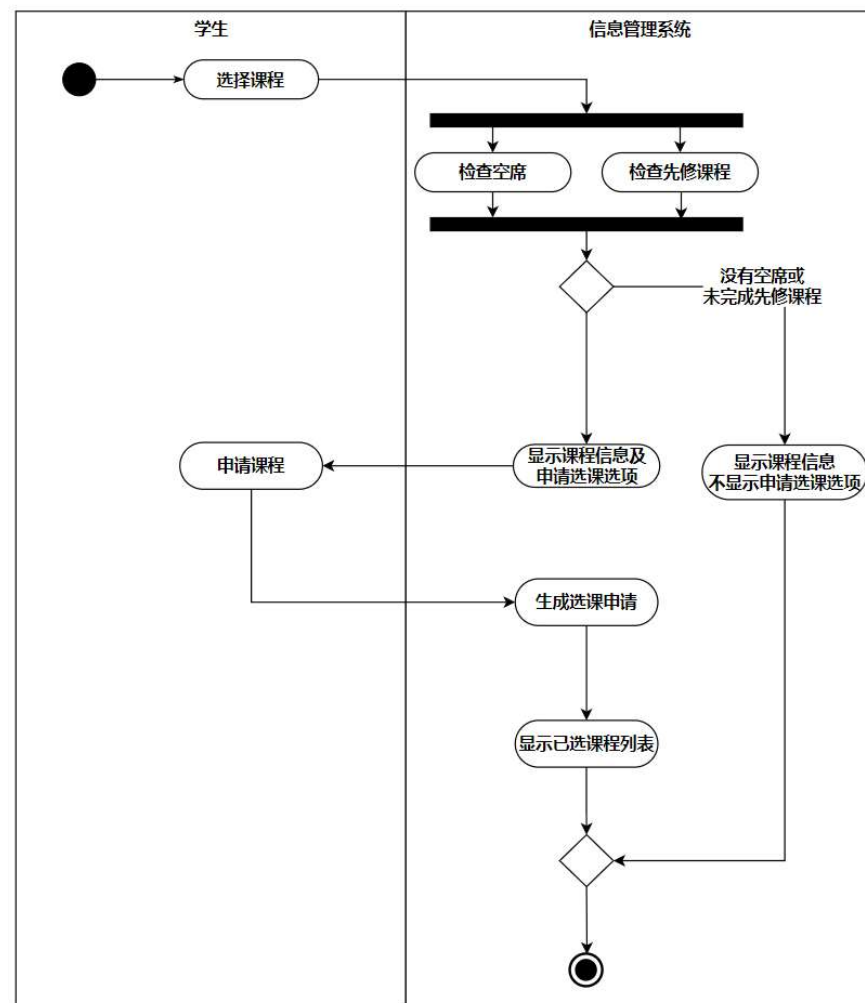
– 确认**任何复用的可能**

– **细化用例模型**（如有需要）

– 记录系统分析**用例描述**

✓ 建模用例**活动**

▸ 绘制系统**顺序图**

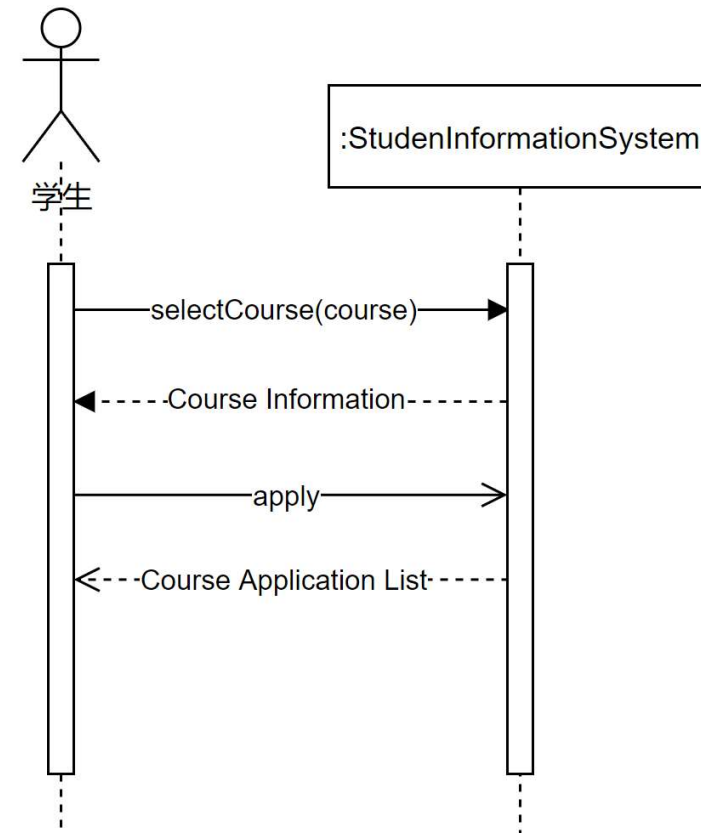


绘制系统顺序图 Sequence Diagram

●顺序图(Sequence Diagram):

又称时序图。描述了用例执行或操作过程中对象如何通过消息相互交互。

- 确定了进入和退出系统的高层消息
- 顺序图只对用例的单一——条路径描述单个场景
- 一个用例可能包括几幅顺序图
- 更好地理解用例步骤的流程和顺序





用例描述

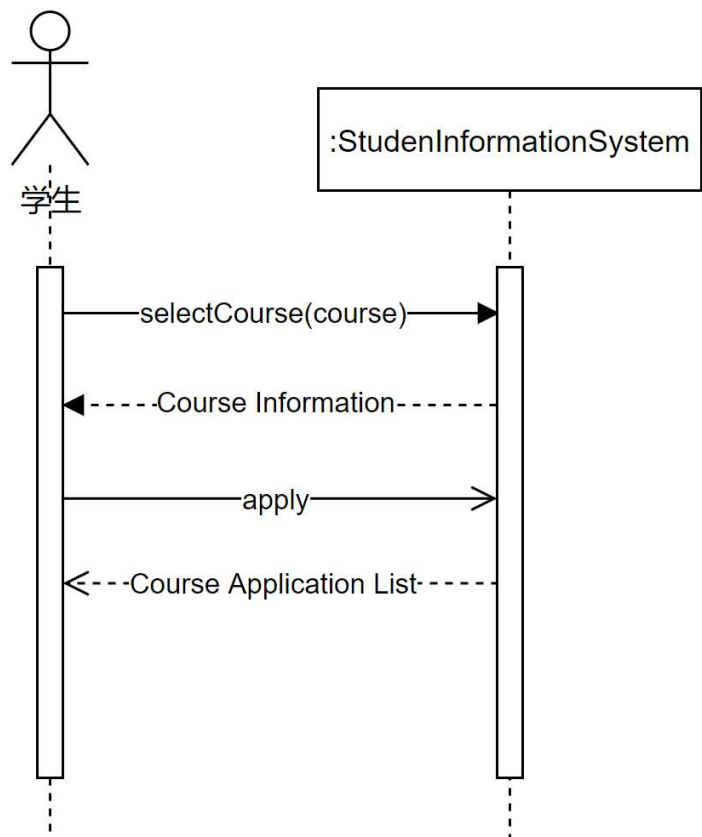
Use Case Narratives

SAD-001: 申请课程

Use Case ID	SAD-001	
用例名	申请课程	
描述	这个用例描述了学生通过系统申请课程的事件。学生从开放课程列表中选择一门课程。学生可以获得课程信息、剩余可选数以及是否完成了预先要求的课程。一旦学生提交申请，创建一个申请并等待批准。	
优先级	高	
主要参与者	学生	
前置条件	学生必须登录系统进行选课	
触发器	当学生选择申请课程时启动用例	
典型事件过程	参与者动作	系统响应
	第1步：学生从打开的课程列表中选择一门课程 第4步：学生浏览课程信息并选择申请	第2步：系统响应检查课程是否有空席，并检查学生是否完成所选课程的所有先修课程 第3步：系统向学生显示课程信息供学生申请 第5步：系统生成新课程申请 第6步：系统在新页面显示学生课程申请列表
替代事件过程	替代第3步：课程没有空席或没有完成所有预修课程，系统显示课程信息，但学生无法申请课程。 替代第4步：学生浏览课程信息。终止用例。	
结论	当学生收到生成应用程序的确认信息时，此用例就结束了。	
后置条件	申请已生成并等待批准。	
业务规则	学生不是休学状态。	
实现约束和说明	- 用例必须24 * 7小时在课程申请期间提供给学生。 - 频率：预计此用例每天将执行 10,000 次。它应该支持多达 50 名学生的并发量。	
假设	- 开放课程列表中的课程可供申请 - 学生可以在获得批准之前取消申请	
开放问题	N/A	

顺序图 Sequence Diagram

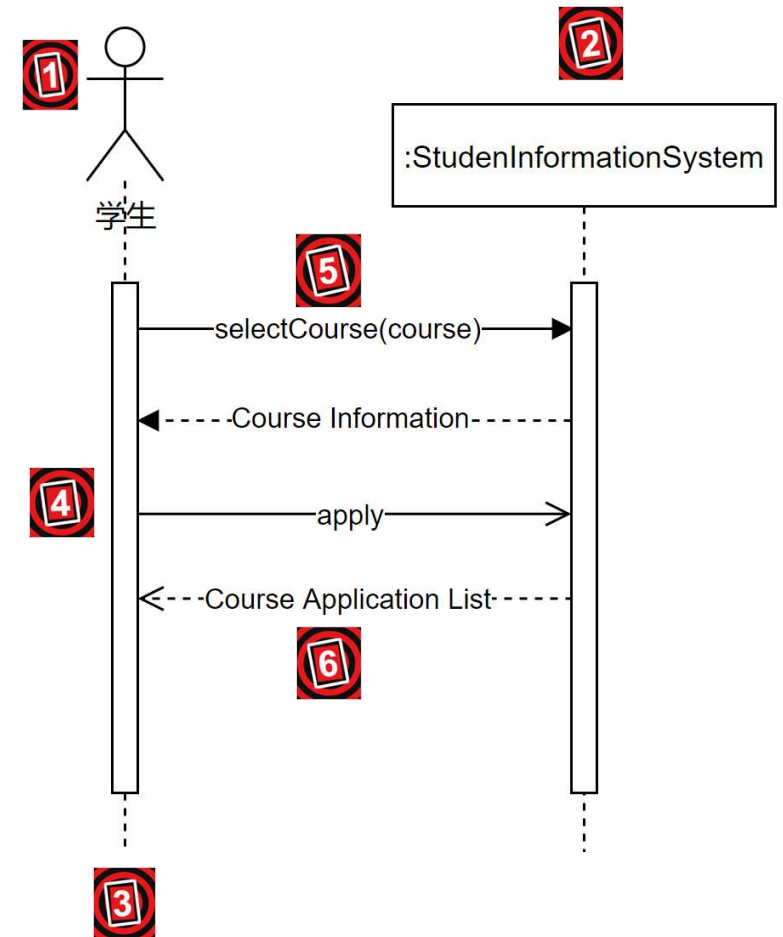
描述了用例执行或操作过程中对象如何通过消息相互交互。



Use Case ID	SAD-001	
用例名	申请课程	
描述	这个用例描述了学生通过系统申请课程的事件。学生从开放课程列表中选择一门课程。学生可以获得课程信息、剩余可选数以及是否完成了预先要求的课程。一旦学生提交申请，创建一个申请并等待批准。	
优先级	高	
主要参与者	学生	
前置条件	学生必须登录系统进行选课	
触发器	当学生选择申请课程时启动用例	
典型事件过程	参与者动作	系统响应
	第1步：学生从打开的课程列表中选择一门课程	第2步：系统响应检查课程是否有空席，并检查学生是否完成所选课程的所有先修课程
	第4步：学生浏览课程信息并选择申请	第3步：系统向学生显示课程信息供学生申请
		第5步：系统生成新课程申请
		第6步：系统在新页面显示学生课程申请列表
替代事件过程	替代第3步：课程没有空席或没有完成所有预修课程，系统显示课程信息，但学生无法申请课程。 替代第4步：学生浏览课程信息。终止用例。	
结论	当学生收到生成应用程序的确认信息时，此用例就结束了。	
后置条件	申请已生成并等待批准。	
业务规则	学生不是休学状态。	
实现约束和说明	- 用例必须24 * 7小时在课程申请期间提供给学生。 - 频率：预计此用例每天将执行 10,000 次。它应该支持多达 50 名学生的并发量。	
假设	- 开放课程列表中的课程可供申请 - 学生可以在获得批准之前取消申请	
开放问题	N/A	

顺序图 Sequence Diagram

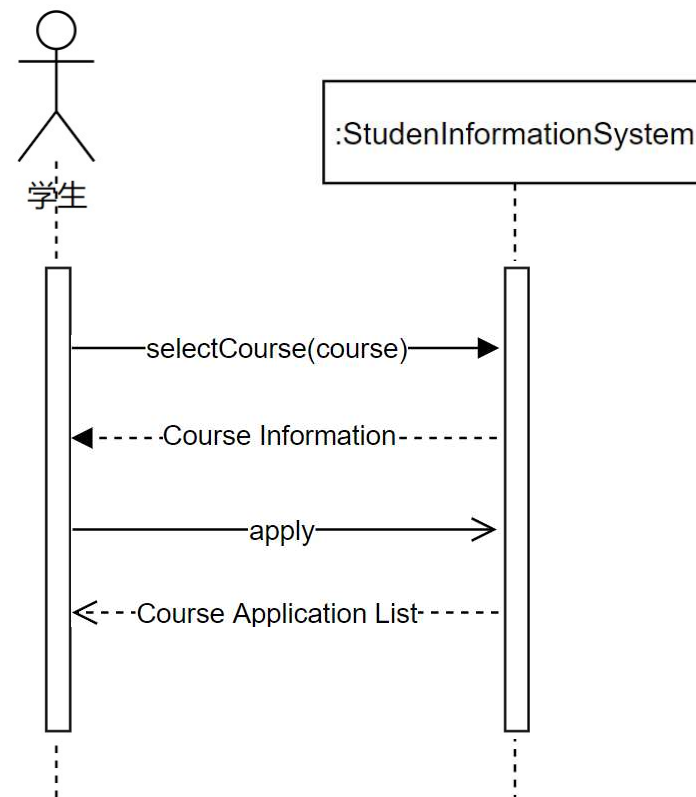
- ① 角色—用例的发起角色使用用例**参与者**符号表示
- ② 系统—盒子表示系统作为一个“黑盒子”或一个整体。**冒号(:)**用来表示系统的一个运行“实例”
- ③ 生命线—从角色和系统符号**向下延伸的垂直虚线**，表示生命顺序
- ④ 活动线—放置在**生命线上的条形**表示参与者进行交互活动的一段时间
- ⑤ 输入消息—从角色到系统的**水平箭头**表示消息输入
第一个单词字母小写，后续单词首字母大写，单词间无空格。
括号内包含了要传递的参数，逗号分隔每个参数。
- ⑥ 输出消息—从系统到角色**虚线的水平箭头**。
不需要使用标准的命名规范。（但是想用也可以用）



构造顺序图指南

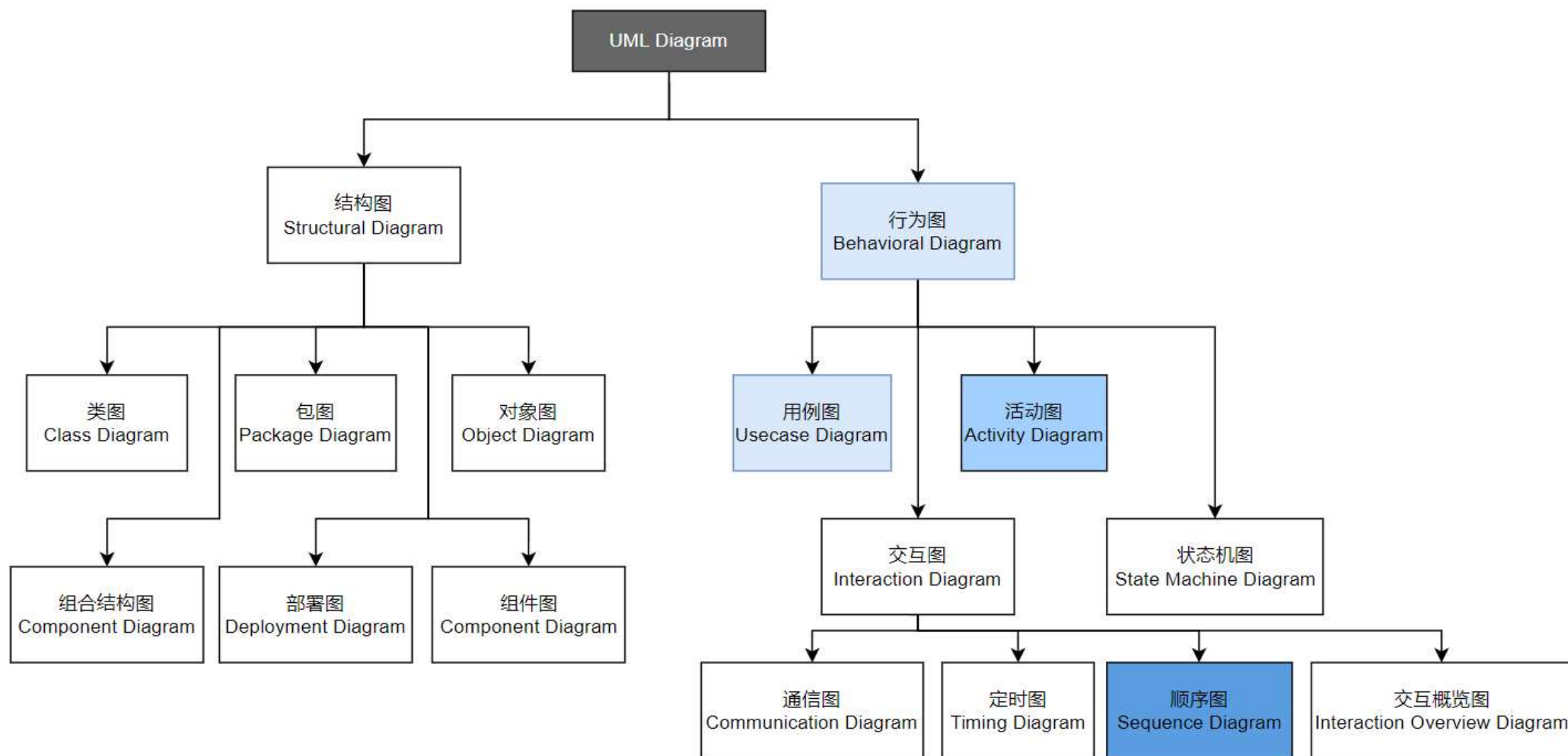
Guidelines for Constructing Sequence Diagrams

- ① 确定描述用例的哪个场景。顺序图的目的是发现消息，不是建模逻辑
- ② 绘制一个矩形表示一个系统，并在其下面延伸生命线
- ③ 确定每个直接给系统提供一条输入的角色，或者直接从系统接收输出的角色。在角色下延伸生命线
- ④ 检查用例描述来确定系统的输入和输出。忽略系统内部消息。
- ⑤ 添加框体以表示带条件的可选消息。框体也可以表示循环和替代片段。
- ⑥ 自顶而下验证消息按照正确的顺序显示出来



UML图

UML Diagrams



本节内容

Readings

《系统分析与设计方法》

- 第10章 使用UML进行面向对象分析和建模
- 第18章 使用UML进行面向对象设计和建模

《UML系统分析与设计教程》

- 第2章 面向对象分析与设计方法

- 关键词：面向对象方法；UML；OOA；OOD；活动图；顺序图