第一章软件工程概述 PPT

一、软件过程模型（软件开发模型）：>第一章 P35

1. 瀑布模型（线性顺序模型）
2. 原型模型

   有效的使用原型模式是：建造原型仅是为了定义需求，之后就被抛弃（或被部分抛弃），实际的软件在充分考虑了质量和可维护性之后才被开发。
3. 增量模型

   是一种渐进地开发逐步完善的软件版本的模型。

   特点：

   （1）反复的应用瀑布模型的基本成分和原型模型的迭代特征，每一个线型过程产生一个“增量”的发布或提交，该增量均是一个可运行的产品。

   （2）第一个增量往往是核心产品，即满足用户的基本需求，提供给用户评估的平台，以便制定下一个增量计划。
4. 螺旋模型

   对于复杂的大型软件，开发一个原型往往达不到要求。螺旋模型将瀑布模型和增量模型结合起来，加入了风险分析。
5. 形式化方法模型（变换模型）

是基于形式化语言和程序变换的模型。强调分析和设计上的严格性，以及使用基于数学的正确性证明来对设计模型的每个元素进行形式化验证。

6. 构件组装模型

该模型以构件(组件)的 方式支持软件重用，对缩短软件开发周期、降低项目成本有重要的现实意义。

二、统一过程（RUP）：>第一章 P55

是一个二维的迭代结构

RUP 的主要特点：P60

RUP 主要强调的是过程，认为具有好的过程才能生产出稳定质量的软件。

1. 用例驱动

用例作为系统分析、设计、实现和测试的基本输入。即用例不只是一种确定系统需求的工具，它还能驱动系统的设计、实现和测试的进行。

2. 以构架为中心

软件系统的构架从不同角度描述了即将构造的系统，它刻画了系统的整体设计，去掉了细节部分，突出了系统的重要特征，包含了系统中最重要的静态结构和动态行为。

3. 迭代与增量的过程

迭代指工作流中的步骤，是反复求精的概念；

增量指产品中增加的部分,是逐块建造的概念。

构架提供了一种结构来指导迭代过程中的工作，**用例**则确定了目标并驱动每次迭代的工作。

4. 基于构件

   统一过程所构造的软件系统，是由软件构件通过明确定义的接口相互连接所建造起来的。

5. 使用 UML(统一建模语言)

6. 过程可剪裁

   RUP 是一个过程模式，或过程定义（PD）而非过程（P）。可以根据系统的规模、项目的性质、开发组织的经验对过程加以裁剪。

问题：

【例】假设你被任命为一家软件公司的项目负责人，你的工作是管理该公司已被广泛应用的字处理软件的新版本开发。由于市场竞争激烈，公司规定了严格的完成期限并且已对外公布。你打算采用哪种软件生命周期模型？为什么？

解：

对这个项目的一个重要要求是，严格按照已对外公布了的日期完成产品开发工作，因此，选择生命周期模型时、应该着重考虑哪种模型有助于加快产品开发的进度。使用增量模型开发软件时可以并行完成开发工作，因此能够加快开发进度。

这个项目是开发该公司已被广泛应用的字处理软件的新版本，从上述事实至少可以得出 3 点结论：第一，旧版本相当于一个原型，通过收集用户对旧版本的反映，较容易确定对新版本的需求，没必要再专门建立一个原型系统来分析用户的需求；第二，该公司的软件工程师对字处理软件很熟悉，有开发字处理软件的丰富经验，具有采用增量模型开发新版字处理软件所需要的技术水平；第三，该软件受到广大用户的喜爱，今后很可能还要开发更新的版本，因此，应该把该软件的体系结构设计成开放式的，以利于今后的改进和扩充。

第二章需求工程 PPT

一、需求获取：P10

1. 需求的概念

（1）需求的定义

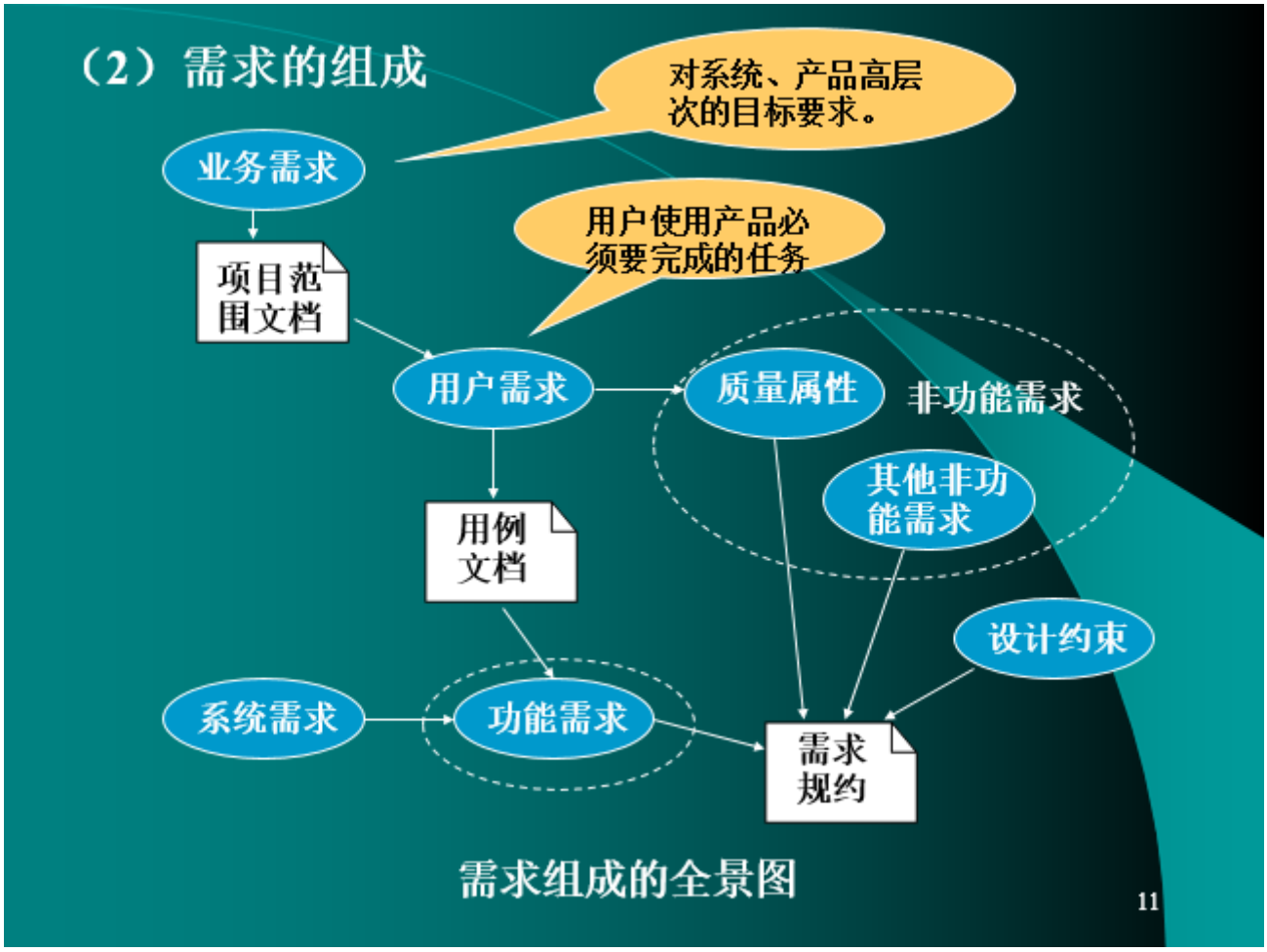
需求是一段…描述：意思是每个需求是相对短小简明的一段信息，表现为一个事实。它可以是一段话或用各种图表示。一组需求的集合成为需求文档。

…关于系统将要完成什么工作…：需求描述了系统应当完成的任务，不描述系统将如何实现。

…必须经过所有相关人员的认可…：意指需求必须经过评审，才能成为正式的需求。

…其目的是彻底地解决用户的问题。有助于解决用户的问题，该需求才有存在的价值。

（2）需求的组成



• 业务需求：反映组织机构和客户对系统、产品高层次的目标要求。

• 用户需求：从用户使用的角度给出需求的描述。

• 系统需求：从系统或技术的角度描述为实现业务需求和用户需求而提供的服务以及所受到的约束。

• 功能性需求：描述系统应该做什么，即为用户和其它系统完成的功能。

• 非功能性需求：产品必须具备的属性或品质。

• 设计约束：设计与实现必须遵循的标准、约束条件。如运行平台、协议、选择的技术、编程语言和工具等。

（3）需求的描述

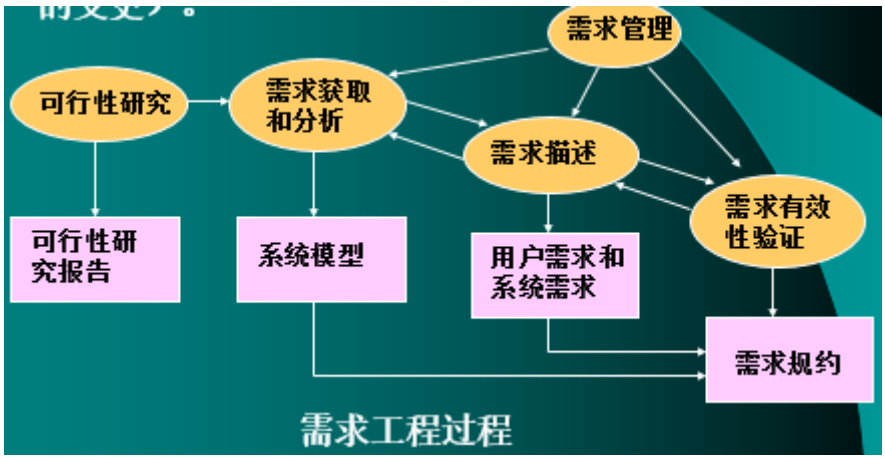
• 自然语言、结构化语言、PDL

• 图形化表示

• 数学描述（形式化语言描述）

2. 需求工程过程 P14

需求工程是一个包括创建和维持系统需求文档所必需的一切活动的过程。它包含了如下活动：系统可行性研究、需求获取和分析、需求描述和文档编写、需求有效性验证、需求管理（管理需求工程的变更）。



可行性研究：系统是否值得开发？给出具体的意见和建议

需求获取与需求分析的区别：

需求获取是调查研究，收集系统需求的过程；

需求分析是将获取到的需求准确的理解、求精，转化为完整的需求定义。

3. 需求分析与建模 P24

需求分析活动具有以下任务：

（1）分析需求的可行性；

（2）对于渐增式开发要确定需求的优先级别，以便确立产品版本；

（3）建模；

（4）生成需求规格说明。

两种主要分析建模：结构化分析、面向对象分析

结构化分析：把软件看作信息转换器，辅助开发人员识别数据、数据之间的关联以及这些数据流经软件处理功能时转换的方式。

面向对象分析：检查定义为一组用例的问题域，提取该域的类，每个类都有自己的属性和操作，类和类之间有多种关联方式，并使用 UML 建模。

具体建模方法：

1.上下文模型（ERD、包图）

2.面向流的建模：数据流图（DFD/CFD）

3.数据建模：实体关系图（ERD）

4.基于场景的建模：用例图、顺序图、活动图

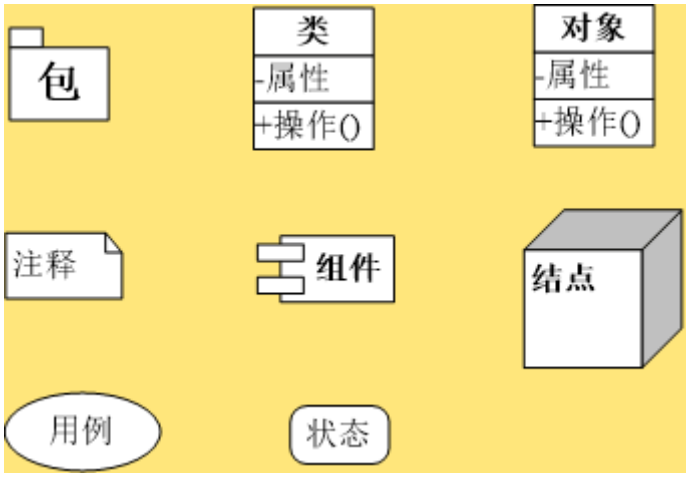
5.基于类的建模：类图、包

6.基于行为的建模： Petri 网、状态图、顺序图、协作图、活动图

第三章构建分析模型 PPT

状态图和顺序图是用于行为建模的 UML 表达方式。

一、UML 中基本模型元素：P8



二、UML 中的基本模型分类：P9

- 用户模型视图

P10 **用例图：** 从用户的角度描述系统能提供哪些功能以及功能的使用者。

用例之间可以相互关联：

包含（inclusion）：一个用例复用另一个用例中的步骤。

扩展（extension）：通过对已有用例增加步骤创建另一个新用例。

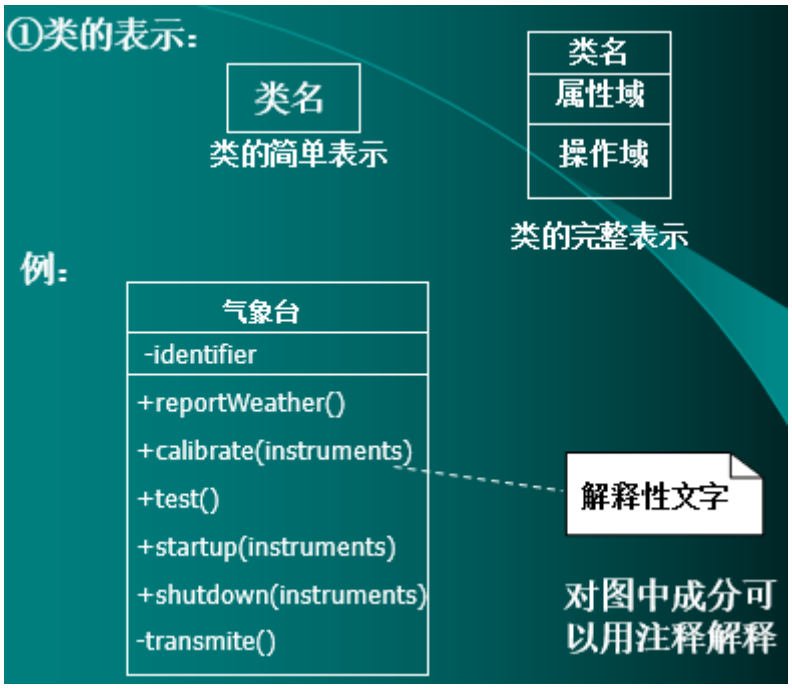
泛化（generalization)：一个用例可以继承另一个用例的行为和含义。

分组（grouping）：一组用例的简单组织方式（如相关的用例放在一个包中）。

- 结构模型视图

P15 **类图**：用来描述系统中类和类之间的关系，是系统的静态结构；

+、#、－分别表示 public、protected、private



②类的关联(association)：

关联表示类之间的语义关系（是运 行时实例之间的联系），关联可以有方向。

角色表示该类在这个关联中的作用。

关联中可以有重数，重数指一个角色可以有多少个对象来扮演

③类的聚合(aggregation)：

是表示“整体-部分”的特殊关联。

类的聚合　　类的强聚合（组合）

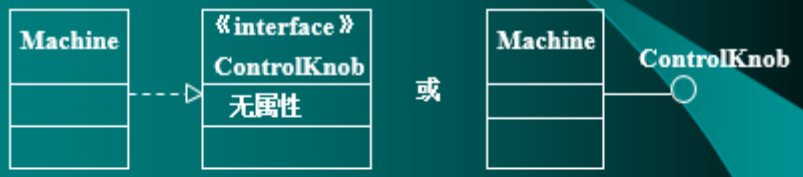
组合(composition)是一种强类型的聚合，整体类和部分类共存亡，如果整体类被撤销，部分类也不存在。部分类的存在只是为整体类服务。

④类的泛化(generalization):

由一个超类和几个直接子类构成的结构通常称为泛化 。类的这种关系也称为一般-特殊关系或继承关系，将现实世界实体的共同特性抽象为一般类，通过增加独有的特性而成为各种特殊类。如图所示：

⑤接口与实现（interface and realization)

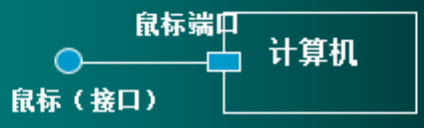
接口是描述类的部分行为的一组描述，是该类提供给别的类的一组操作。下图中右图为左图的省略表示。



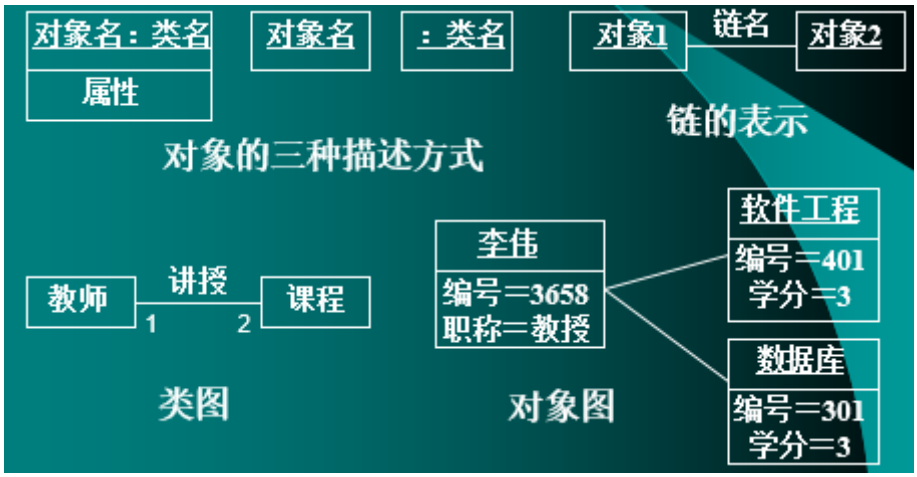
一个类和它接口之间的关系叫做实现。一个类可以实现多个接口，一个接口也可以被多个类实现。

⑥接口与端口（port)

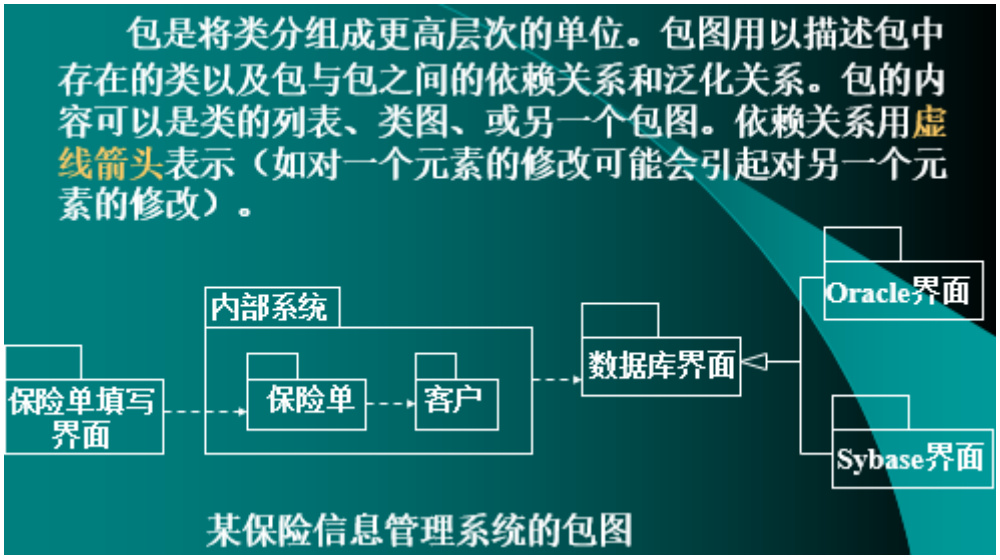
UML2.0增加了一个符号来表示端口。类通过端口与它的环境交互，即端口展示了类与环境交互的点。见下图：



P25 **对象图**：是类的实例图，描述系统在某个时刻的静态结构；



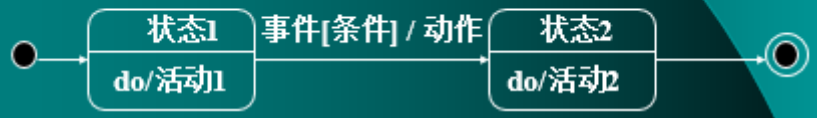
P27 **包图**：将类分组成更高层次的静态结构。



- 行为模型视图

P28 **状态图**：描述系统元素对事件的响应引起的状态转换；

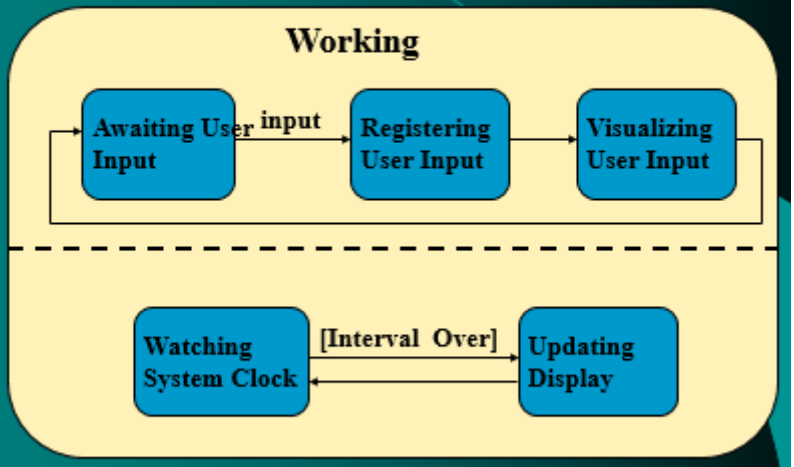
② 转移



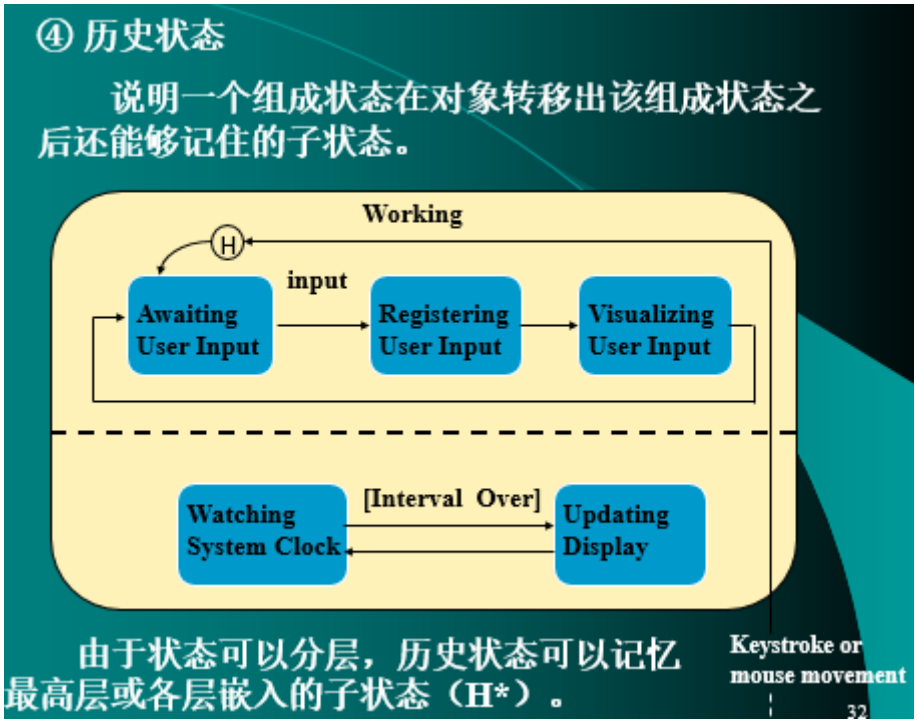
“动作”是为响应事件而执行的行为。do活动指在某个状态内发生的持续一段时间的活动，执行中可被外部的某事件中断。

③ 子状态

状态可以包含子状态：顺序子状态、并发子状态。该状态称为组成状态。

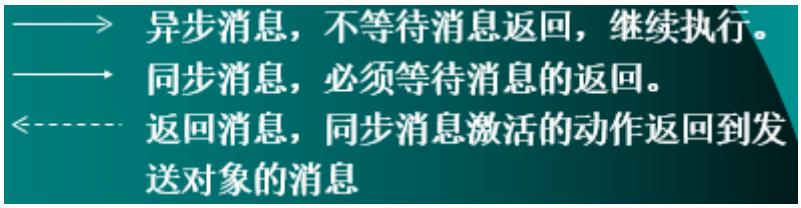


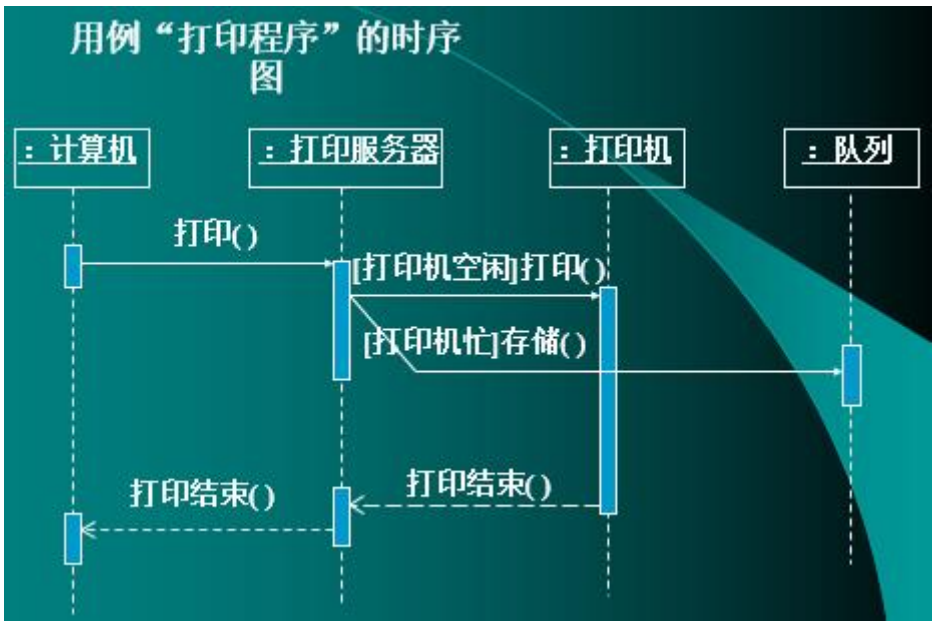
两个状态序列（顺序子状态）之间是并发关系，用虚线隔开。



P34 **顺序图：**按时间顺序描述系统元素之间的交互；

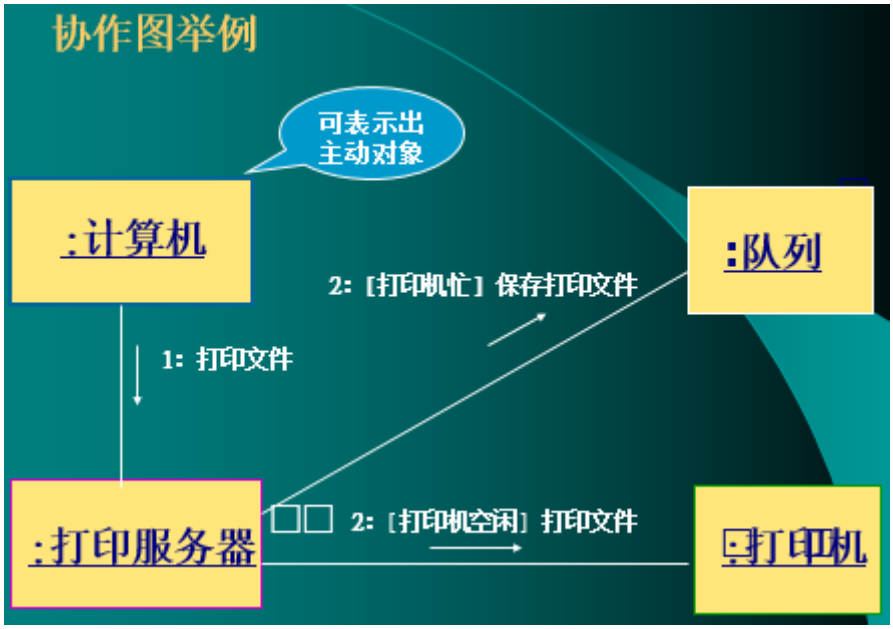
每个对象图符下面的垂直虚线表示对象的生命线，生命线上的细长矩形称为激活（activation)，表示该对象正在执行某个操作，矩形的长度表示执行操作的持续时间。





P40 **协作图：**着重描述系统元素之间的协作；

协作图中可用箭头表示消息，并注明消息名称和可选参数。箭头附在有协作关系的每对对象的通信链上。在每个消息可加消息的序号，代表该消息在消息队列中的序号。



P42 **活动图：**描述系统元素的活动。

其中：

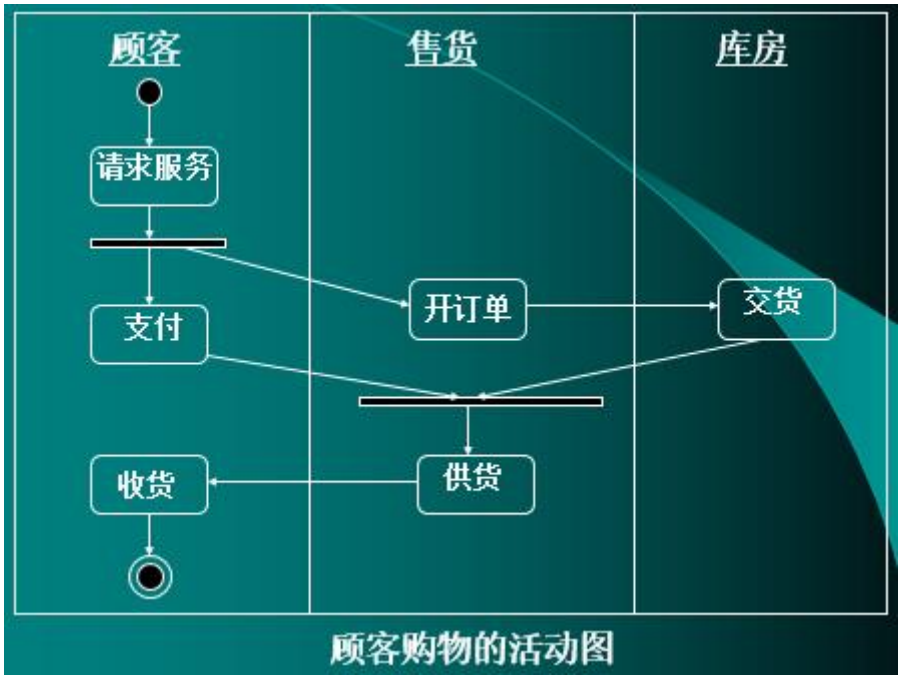
泳道，完成任务的纵向区域；

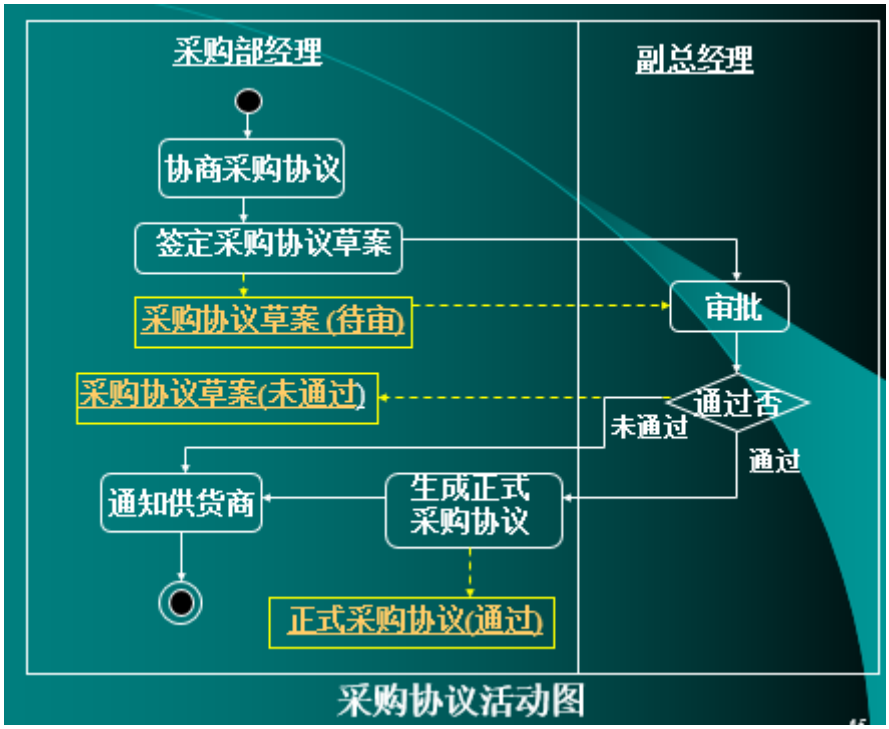
活动，用圆角矩形表示；

决策特殊活动用于判断，用菱形表示；

同步，用粗线表示；

业务对象，用虚线箭头表示数据传送的方向；

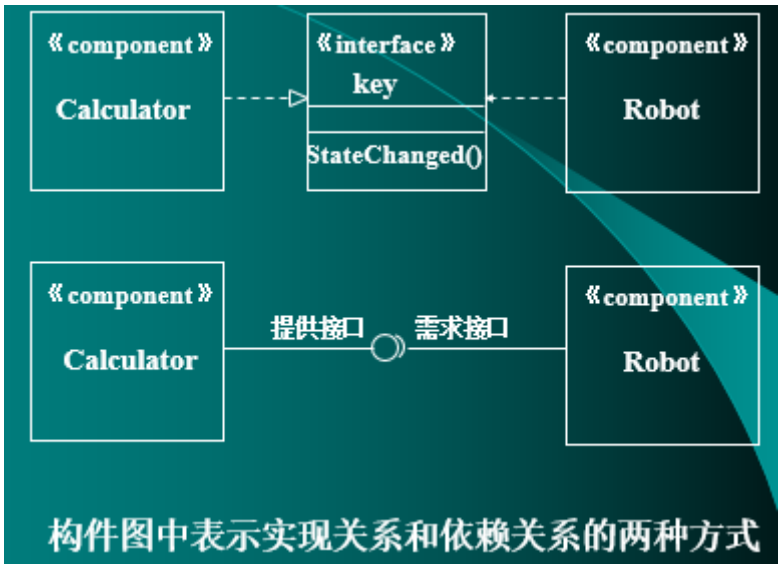


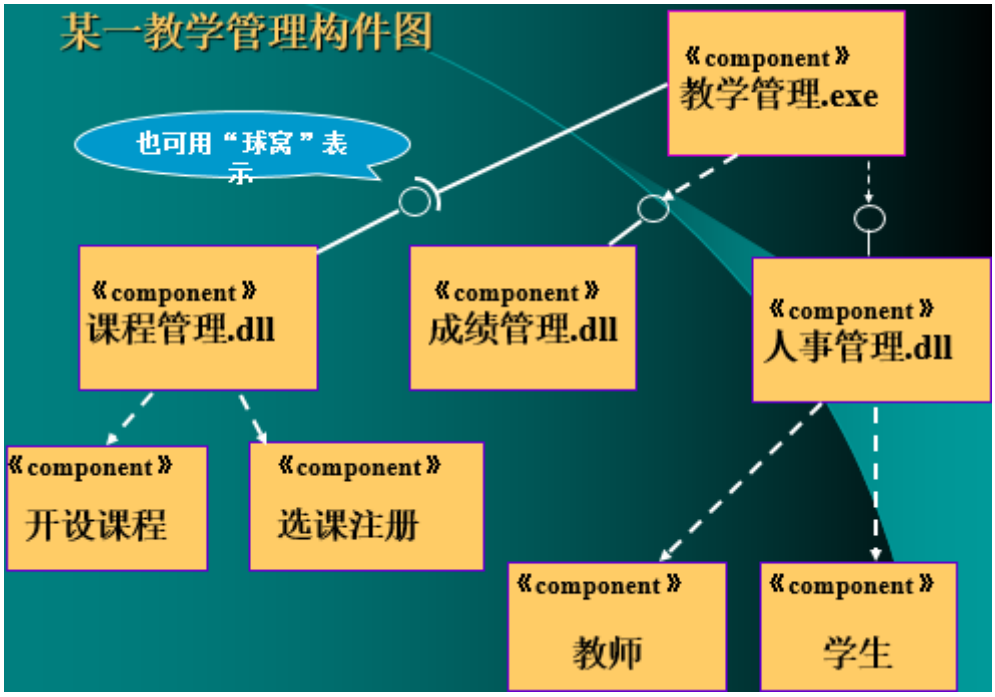


- 实施模型视图

P49 **构件图：**描述实现系统的元素（构件）的组织结构；

构件是一个实际文件。



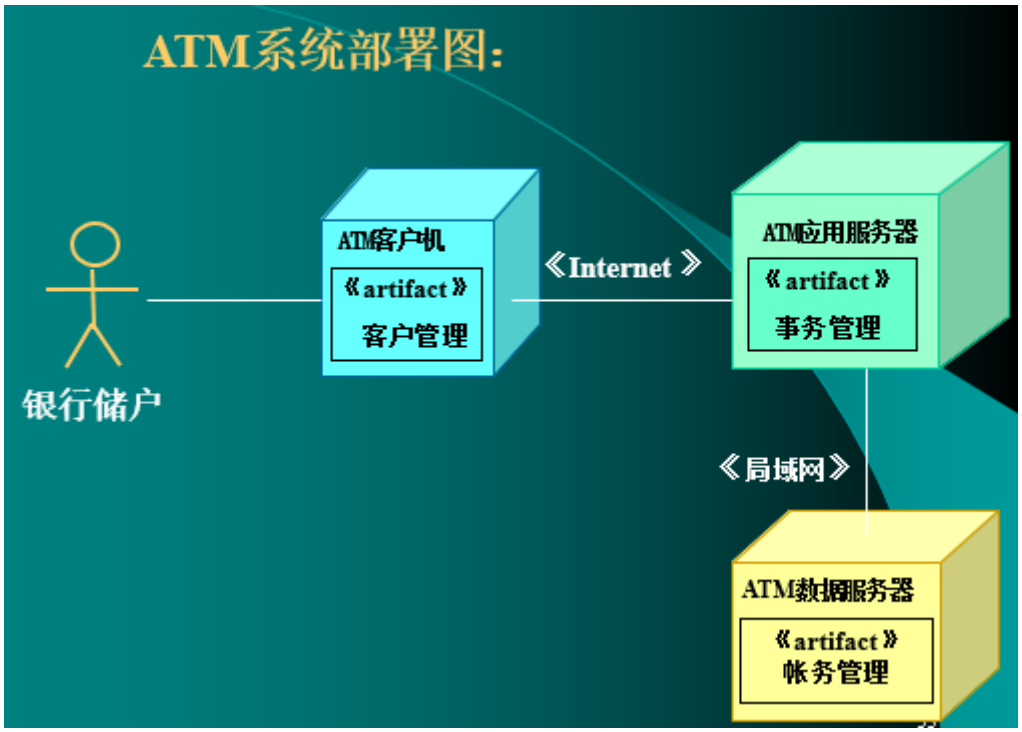


P52 **配置图：**描述环境元素的配置，并把实现系统的元素映射到配置上。

能清楚地展示分布式系统运行时的体系结构。

双尖括号括起来的元素为构造型（stereotype)元素。

构造型使设计者能够在现有的 UML 元素的基础上创建新的元素，被括起来的名称称为关键字。

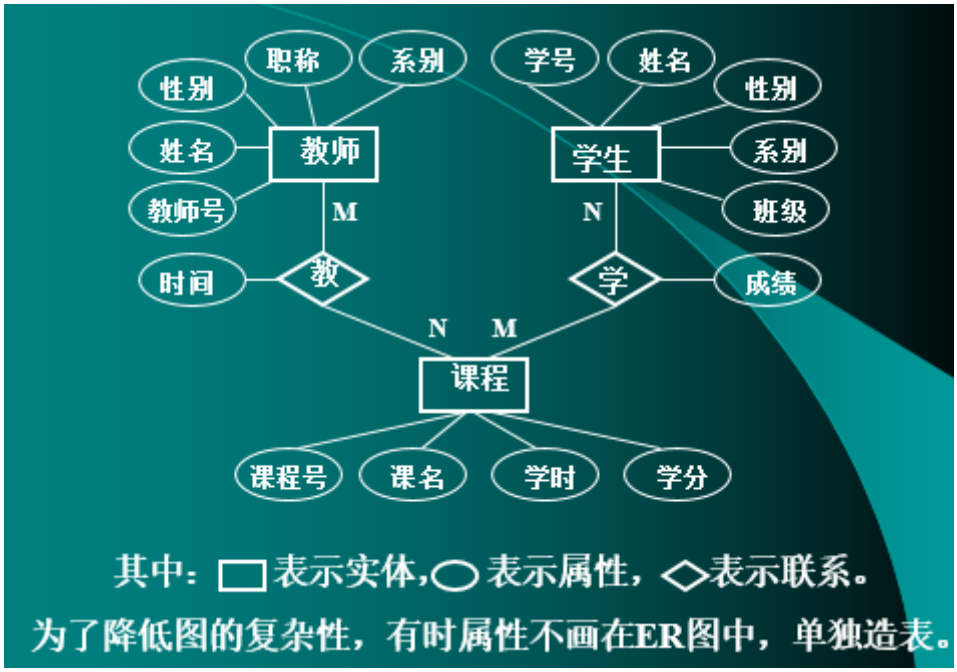


三、数据建模：

P60 **ER 图：**实体-联系图

包括三种相互关联的信息：

数据对象、数据对象的属性、数据对象相互连接的关系(1:1，1:N，M:N)



四、面向流的建模(DFD)：

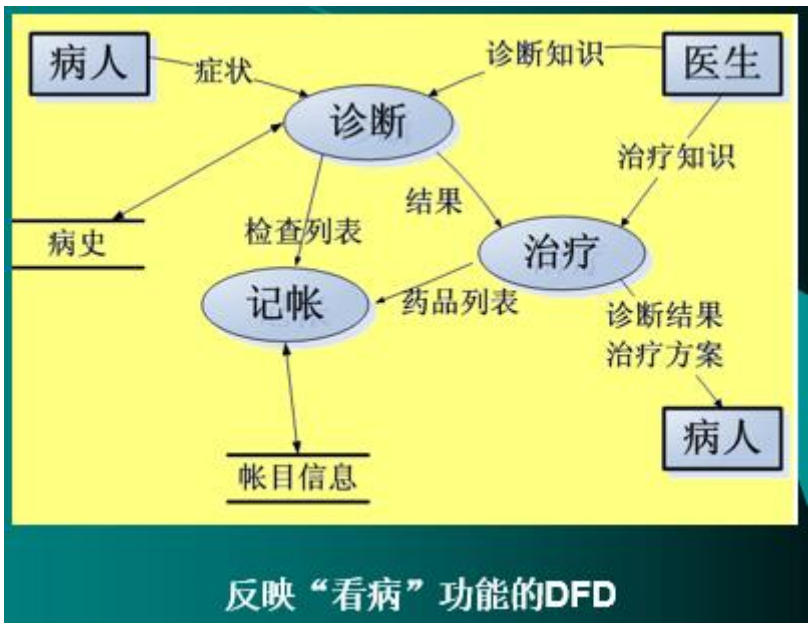
数据流模型用来描述数据流经软件处理时转换的方式，是从业务功能角度来看待系统。

箭头：表示流动的数据；

泡泡：表示对数据的变换(加工或处理)；

平行线：表示数据存储(文件或数据库)；

矩形：表示外部实体(数据源)



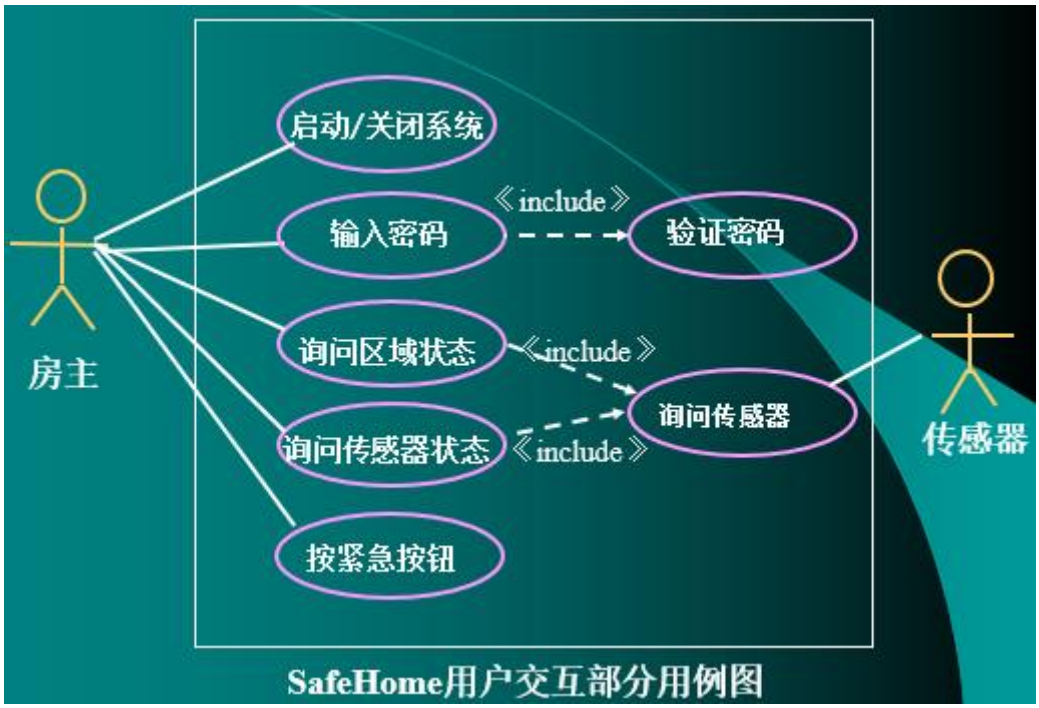
五、基于场景的建模：

1.确定参与者与用例

与用户一起确定与系统有交互活动的所有角色，并为每个角色设计用例。

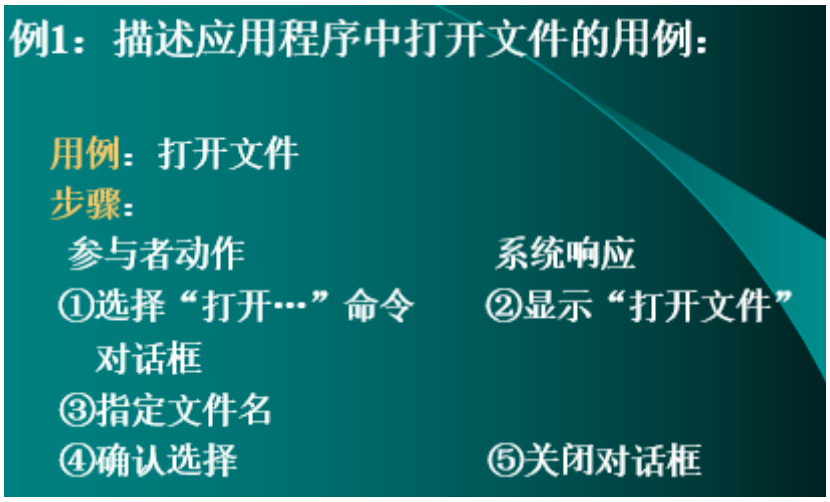
2.建立用例图

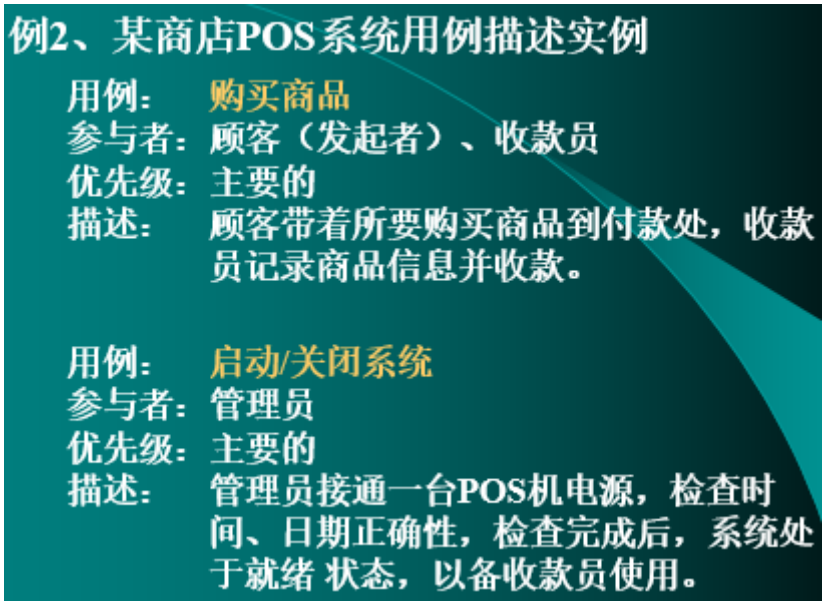
用例图用来显示一系列用例和参与者之间的关系，有助于图形化的表达系统功能的使用。

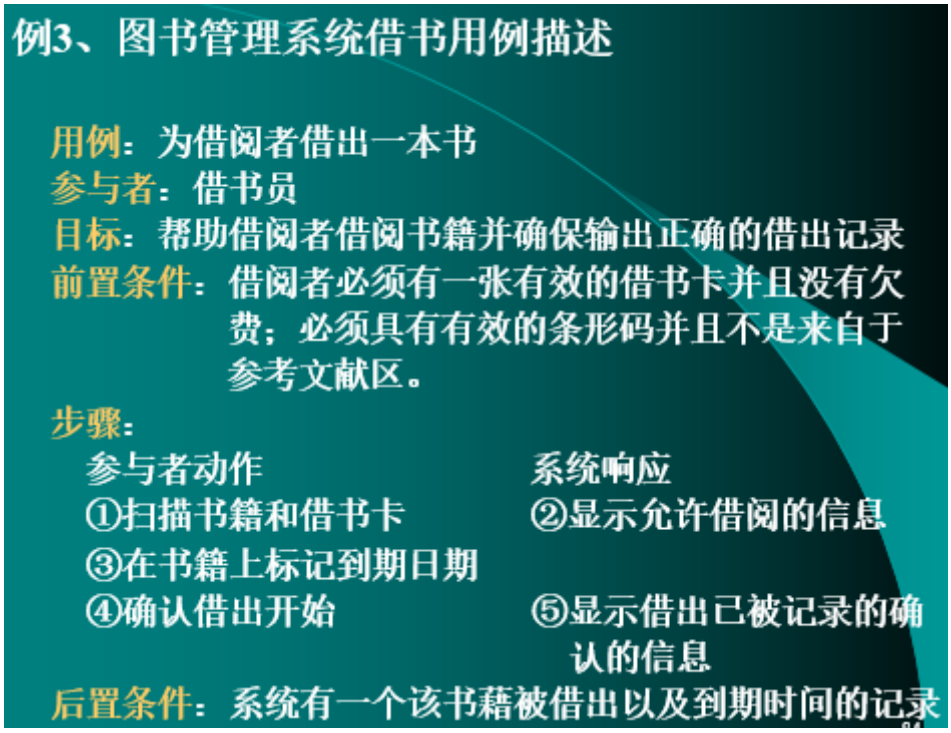


3. 描述每个用例

其中只有名称和步骤是必须的。







六、基于类的建模：

基于类的建模使用从基于用例和面向流的建模元素中提取的信息确定分析类。

建立类图是面向对象分析与设计的核心技术。