Southeast University

软件工程导论

廖 力

lliao@seu.edu.cn

课程结构

Unit1. 软件工程概述

Unit2. 软件工程技术

一、系统工程

二、需求工程

（一）需求工程概要

（二）面向过程的需求工程方法

（三）面向对象的需求工程方法

（四）相关工具

三、设计工程

四、软件构建与测试

Unit3. 软件项目管理

（一）、需求工程概要

1. 什么是需求？

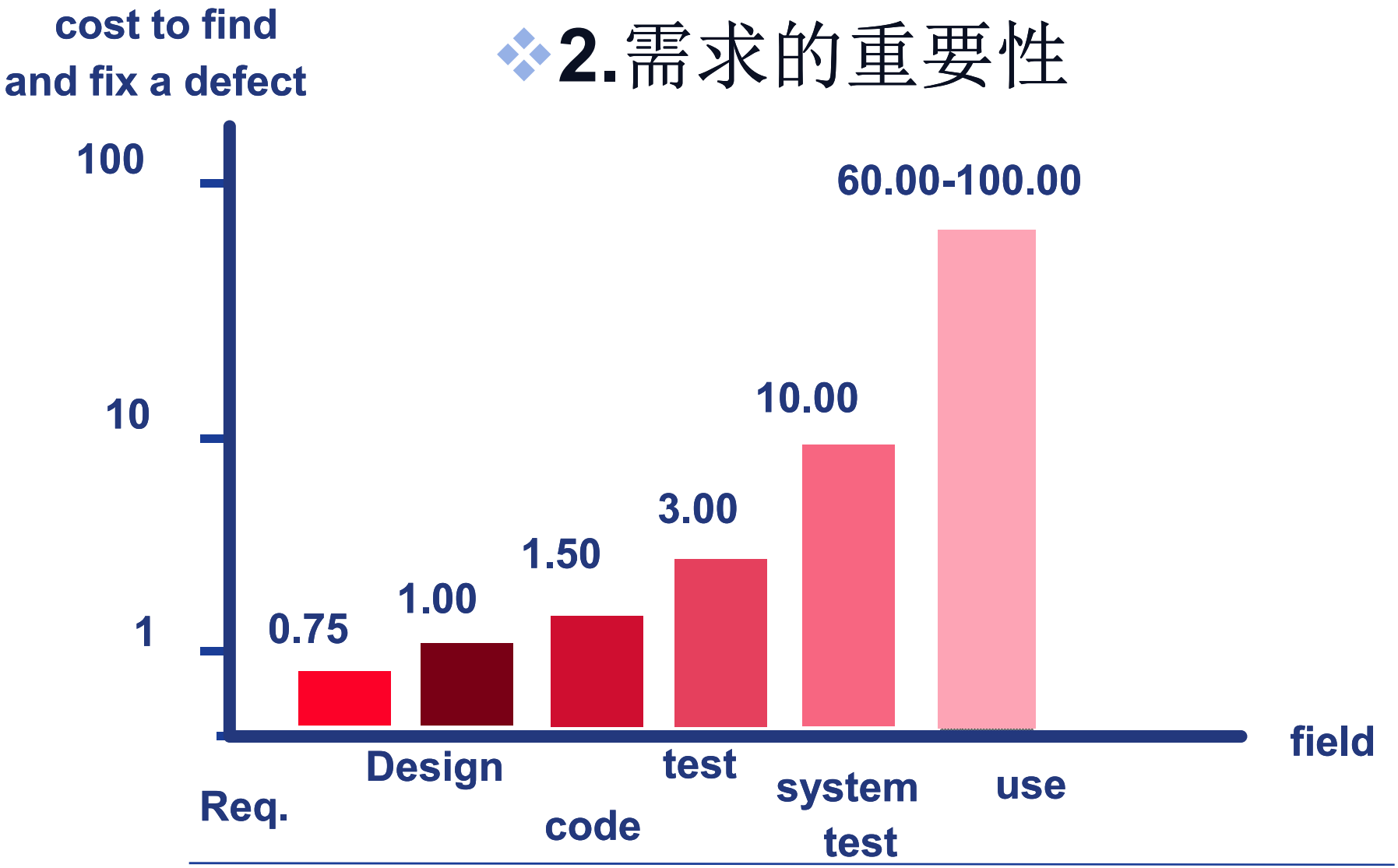
- 软件开发的需求：需求来源于用户的一些“需要”，这些“需要”被分析、确认后形成完整的文档，该文档详细地说明了产品“必须或应当”做什么。

2.需求的重要性

- 需求是产品的根源，需求工作的优劣对产品影响最大。就像一条河流，如果源头被污染了，那么整条河流也就被污染了。

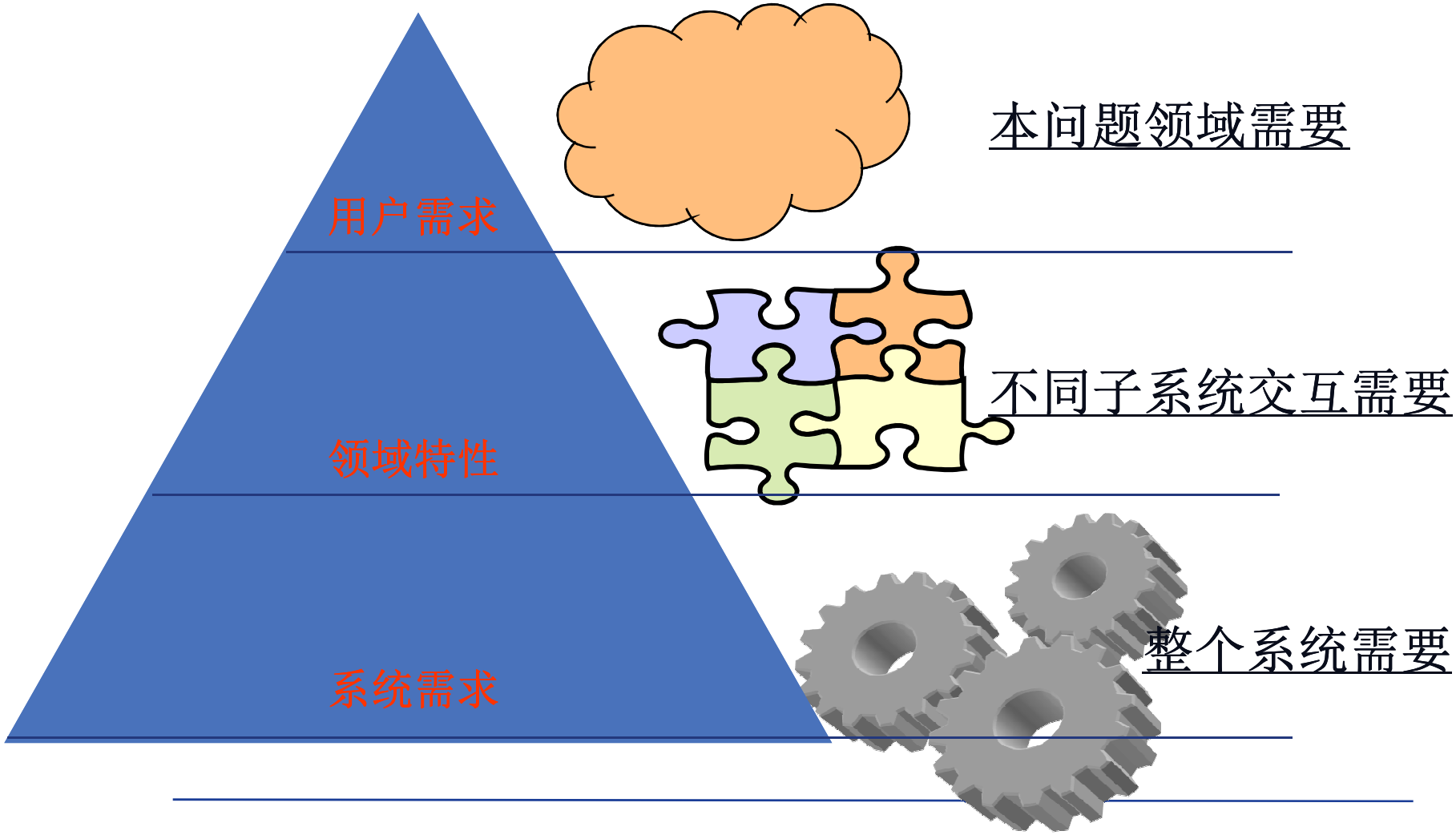
（一）、需求工程概要

❖2.需求的重要性



（一）、需求工程概要

3.需求的分解



（一）、需求工程概要

3.需求的分解

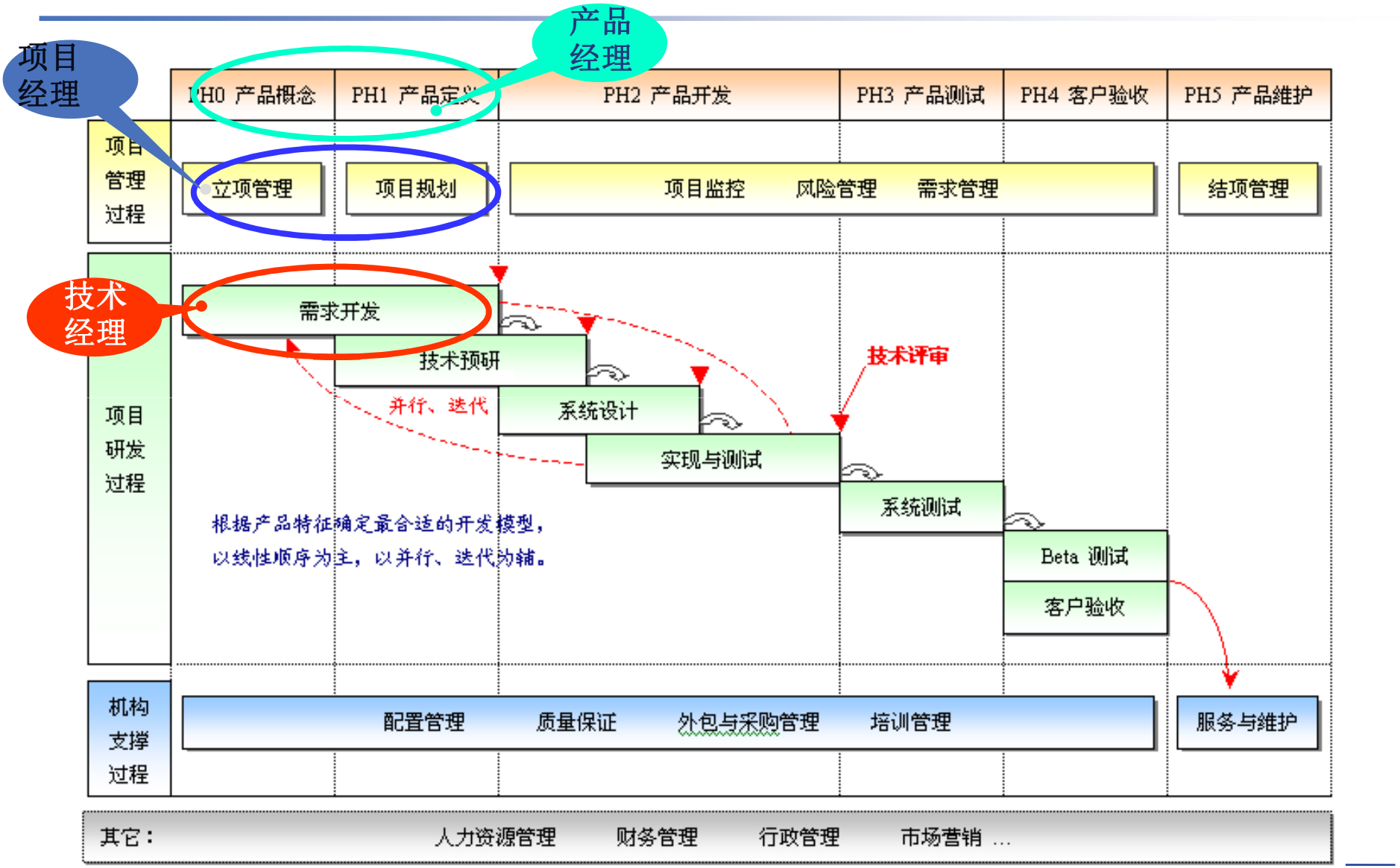
- 功能性需求
  - 系统需要提供的服务或功能：
    - 如图书检索
  - 系统对特定输入的处理方式：
    - 如对非法输入的提示
  - 系统在特定环境下的行为：
    - 如长时间无操作时自动退出

（一）、需求工程概要

3.需求的分解

- 非功能性需求
  - 对系统功能或服务附加的质量约束，例如响应时间、容错性、安全性等——客户所关心的(外部质量)
  - 从系统开发和维护角度出发的质量属性，例如可理解性、可扩展性、可配置性等——软件开发或维护者所关心的(内部质量、软件所特有)

4. 需求工程所处的位置?


项目经理
产品经理
技术经理
PH0 产品概念
PH1 产品定义
PH2 产品开发
PH3 产品测试
PH4 客户验收
PH5 产品维护
项目管理过程
立项管理
项目规划
项目监控
风险管理
需求管理
结项管理
项目研发过程
需求开发
技术预研
系统设计
实现与测试
系统测试
Beta 测试
客户验收
技术评审
并行、迭代
根据产品特征确定最合适的开发模型，
以线性顺序为主，以并行、迭代为辅。
机构支撑过程
配置管理
质量保证
外包与采购管理
培训管理
服务与维护
其它：
人力资源管理
财务管理
行政管理
市场营销 …

（一）、需求工程概要

5.需求工程的发展

- 传统的软件工程中需求分析是通过问题识别、分析与综合、制订规格说明和评审等阶段，达到为系统设计提供依据的目标。因此，需求分析过程包括：
  - 确定对系统的综合要求
  - 分析系统的数据要求
  - 抽象出并确立目标系统的逻辑模型
  - 编写需求规格说明书

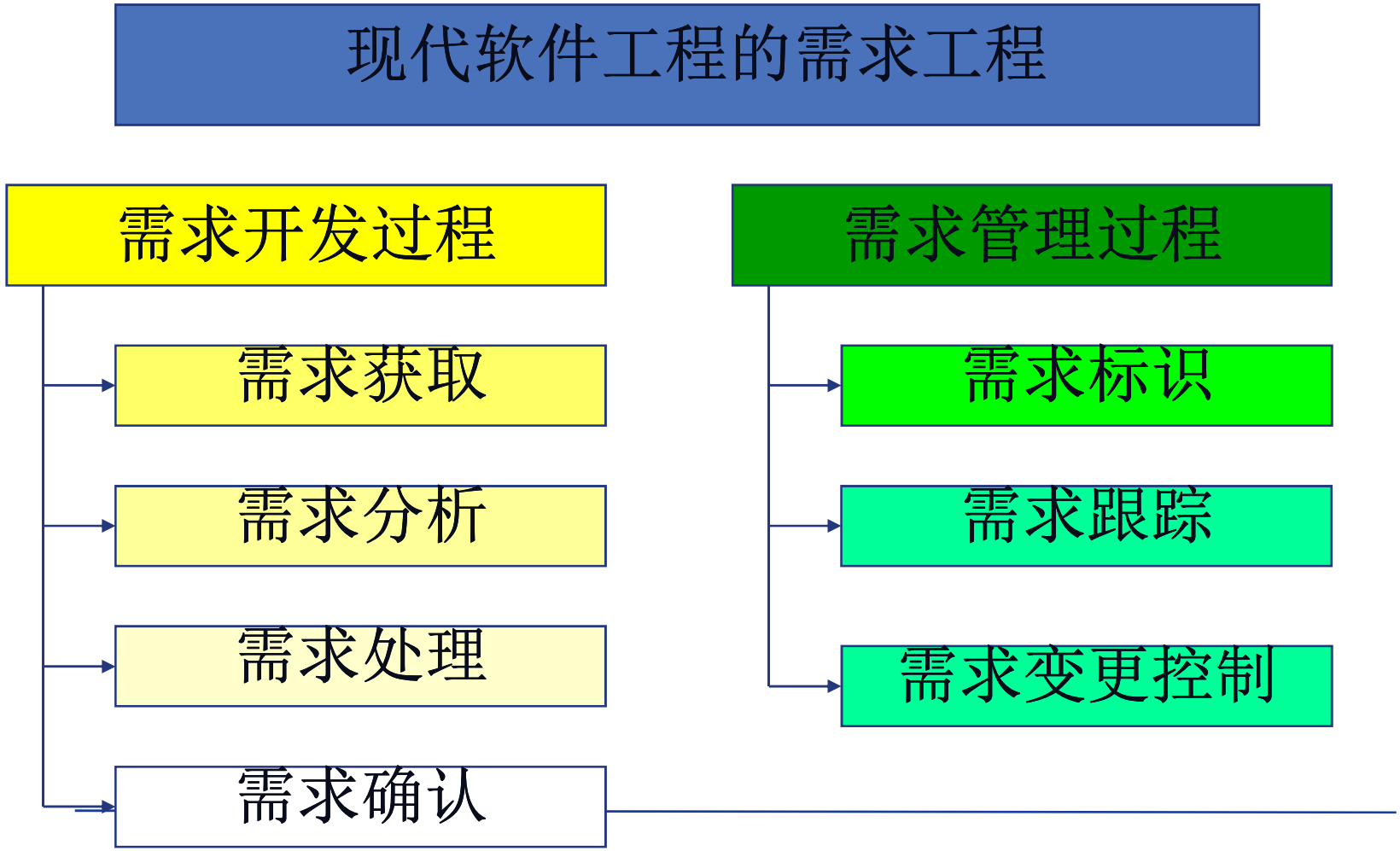
（一）、需求工程概要

5.需求工程的发展

- 仅仅靠需求分析无法解决软件项目中普遍存在的需求不确定性问题，为此发展出“需求工程”的概念。
- 需求工程提供一种适当的机制，以了解用户想要什么、分析需求、评估可行性、协商合理的解决方案、无歧义地规约解决方案、确认规约以及在开发过程中管理这些被确认的需求规约的过程。
- 因此，需求工程的活动也可分为两大过程领域，需求开发和需求管理。

（一）、需求工程概要

5.需求工程的发展



（一）、需求工程概要

- **6.需求开发过程**
- **1）需求获取：**
  - 起始过程（**Inception**）：与客户建立初步交流
  - 导出过程（**Elicitation**）：通过各种途径获取用户的需求信息（原始材料）
- **2）需求分析：**
  - 精化过程（**Elaboration**）：分析建模，建立精确的技术模型，用以说明软件的功能、特征和约束。

（一）、需求工程概要

- 6.需求开发过程
- 3）需求处理
  - 协商过程（Negotiation）：开发者与客户之间就所要交付的系统的功能、性能、交付时间等问题进行估算和协商。
  - 形成规格说明（Specification）：根据需求调查和需求分析的结果，进一步定义准确无误的产品需求，产生《产品需求规格说明书》。这是后续所有工作的基础。
    - 采用原型法的话，specification可能是一系列的原型系统。

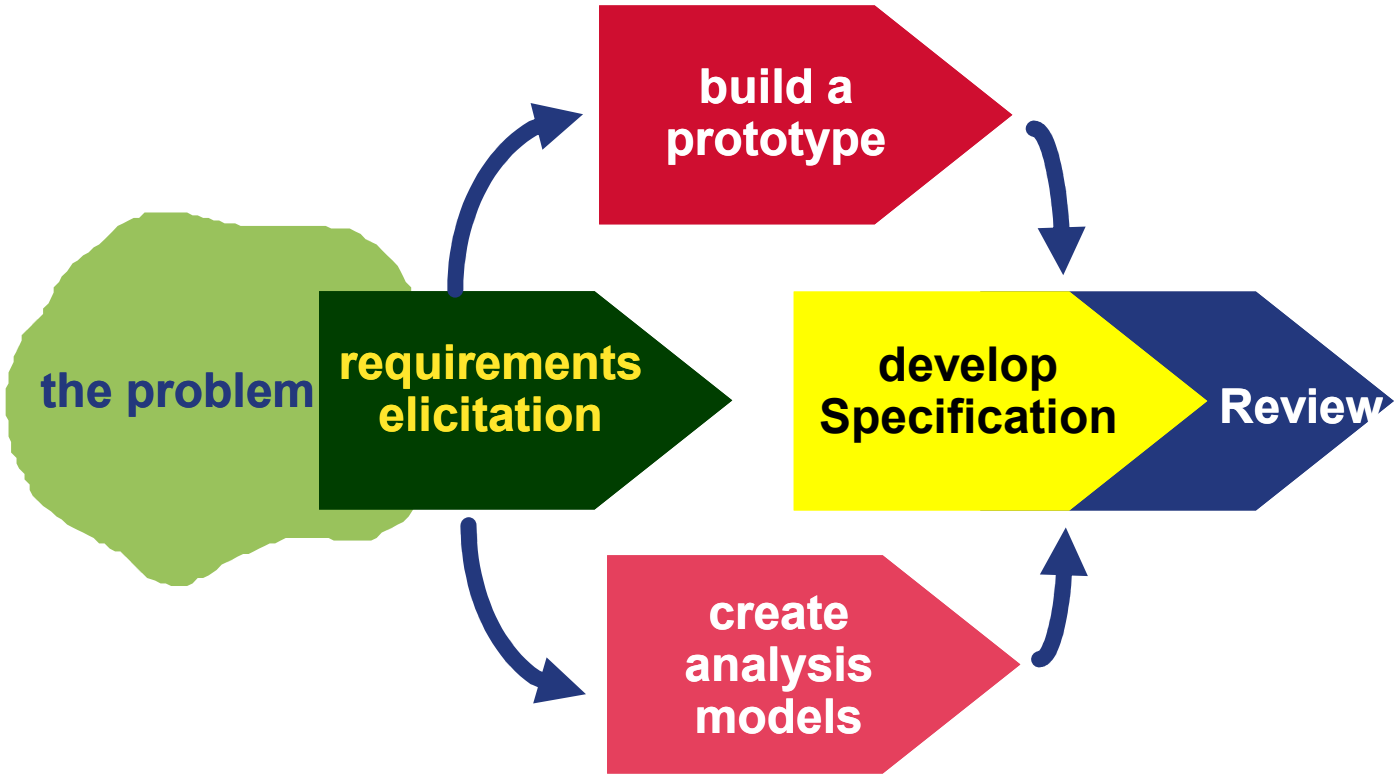
（一）、需求工程概要

- **6.**需求开发过程
- **4）**需求确认
  - **Validation：**开发方和客户共同对需求文档进行评审，双方对需求达成共识后作出书面承诺，使需求文档具有商业合同效果。
  - 评审（**review**）：发现**specification**中的描述错误，描述不清之处，信息缺失，不一致之处等。

（一）、需求工程概要

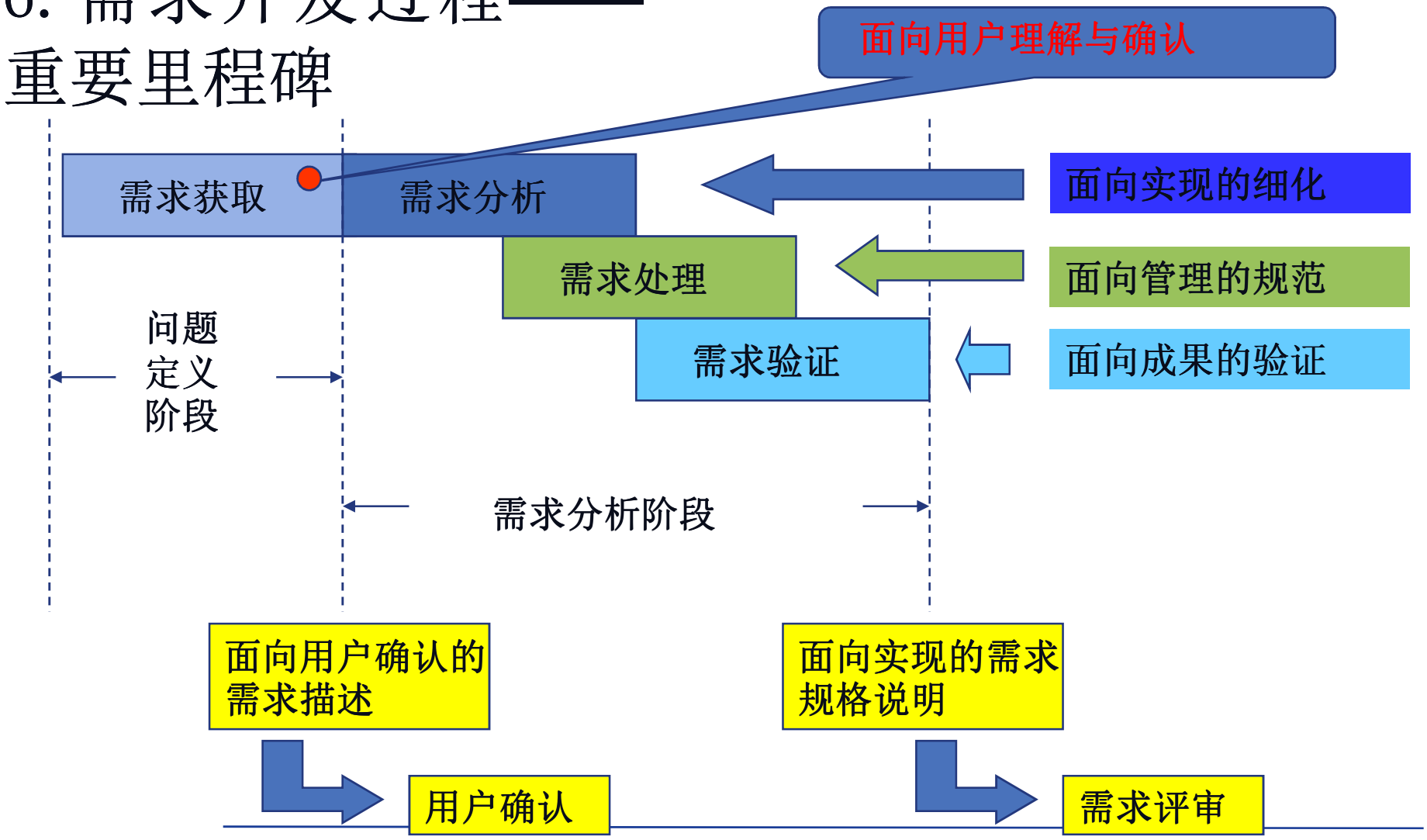
6.需求开发过程

- 需求开发过程流程



（一）、需求工程概要

6. 需求开发过程——重要里程碑



（一）、需求工程概要

7.需求管理过程

- 管理过程的必要性
  - 基于计算机的系统需求会经常变更，而且变更的要求贯穿于系统的整个生存期。
- 需求管理的作用
  - 帮助项目组在项目进展中标识、控制和跟踪需求，并对需求变更进行跟踪和控制。
  - 主要内容将在“软件配置管理”部分学习。

（二）面向过程的需求工程方法

- 1.需求获取方法
  - 访谈与调查（会议，问卷，原型）
- 2.需求分析方法
  - 数据流图，实体关系图，数据字典，控制流图
- 3.需求处理方法
  - 需求规格说明书模板
- 4.需求确认方法
  - 评审

（二）面向过程的需求工程方法

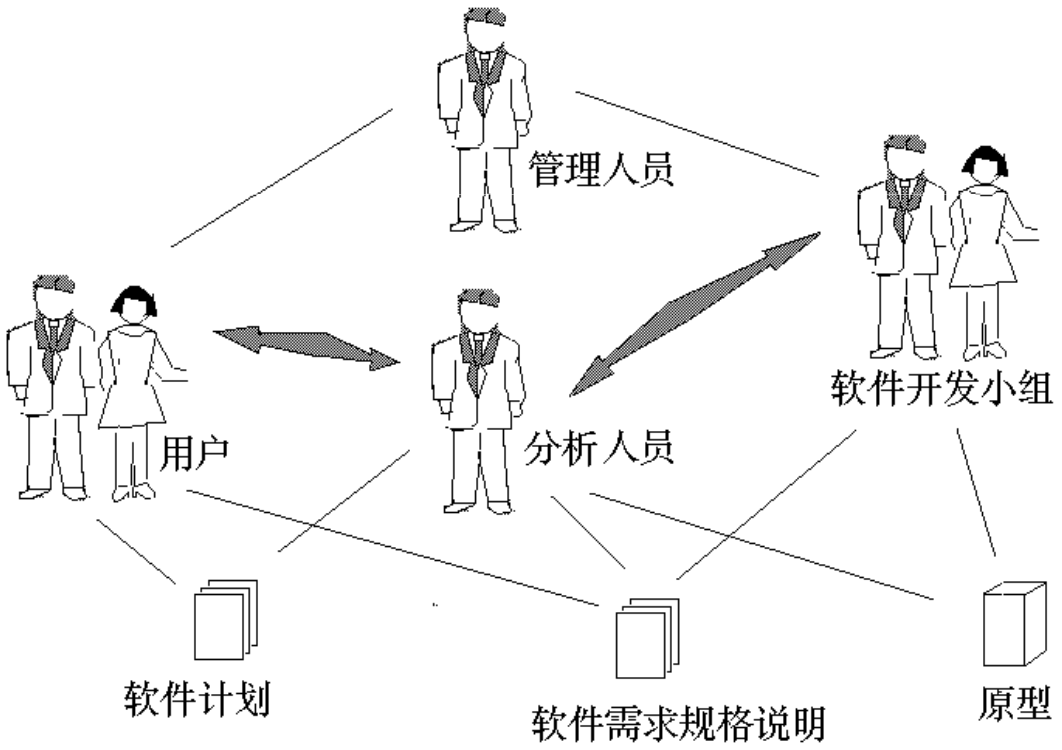
1.需求获取方法

- 需求获取步骤
  - 1）确定stakeholders
  - 2）从不同视角，不同角度收集信息
  - 3）stakeholders协同合作收集需求
    - 会议，问卷，原型
  - 4）创建用户场景，构建用况

（二）面向过程的需求工程方法

1.需求获取方法

- 1）确定stakeholders



（二）面向过程的需求工程方法

1.需求获取方法

- 2）从不同视角，不同角度收集信息
  - 访问三个负责不同方面的**stakeholders**，你会得到四个或更多的观点……
    - 市场营销部门关注能激发市场的功能和特点
    - 业务经理关注预算
    - 最终用户关注界面友好
    - 软件工程师关注软件基础设施
    - 支持工程师关注可维护性

（二）面向过程的需求工程方法

- 1.需求获取方法
  - 2）从不同视角，不同角度收集信息
    - 注意：
    - 多角度收集，需求很可能不一致甚至矛盾。
    - 需求工程师的工作就是将这些信息分类，以便于决策者从中选取一致的需求集合。

（二）面向过程的需求工程方法

- **1.需求获取方法**
  - **3）stakeholders协同合作收集需求**
  - 方法：访谈和调查
  - **访谈与调查的原则**
    - 所提问的问题应该循序渐进，从整体的方面开始提问，接下来的问题应有助于对前面的问题更好的理解和细化
    - 不要限制用户对问题的回答，这有可能会引出原先没有注意的问题
    - 提问和回答在汇总后应能够反映用户需求的全貌——不断汇总-反馈-汇总

（二）面向过程的需求工程方法

1.需求获取方法

- **3）stakeholders协同合作收集需求**
- 方法：会议讨论法
  - 适用于需求调研早期
  - 特点：需求获取的信息量大，但有时全面性和深入性不足
  - 做好调研计划，同时掌握好计划与灵活性的平衡
    - 一般由客户方的基本情况介绍开始
    - 随着情况了解的深入，分析人员逐渐成为主导：要求分析人员有较强的理解和交流能力、思维敏锐，能够有效地引导并把握讨论的议题和方向

（二）面向过程的需求工程方法

1.需求获取方法

- 3）stakeholders协同合作收集需求
- 方法：调查问卷
  - 由分析人员拟定问卷(选择、判断居多)请客户方代表回答
  - 适用于需求调研的中后期，往往用于对前期发现的一些不明确或不一致的地方进行确认
  - 特点：信息量较小，但能够引导客户对某些关键问题进行思考，给出明确、严谨的回答和判断

（二）面向过程的需求工程方法

1.需求获取方法

- **3）stakeholders协同合作收集需求**
- 方法：原型系统
  - 由开发小组快速开发一个近似的功能原型(往往以操作界面为主)，分析人员与客户围绕着原型系统的演示进行需求讨论
  - 适用于客户仅有一些宏观的设想而自身需求还不明确的情况
  - 特点：能够帮助客户将自己的设想落实为具体需求，能够有效激发客户的思维，但需要额外的原型开发开销
  - 要求分析人员自身对需要有较强的认识，基本能够把握客户的潜在想法或者开发方有类似的成品软件

（二）面向过程的需求工程方法

1.需求获取方法

4）创建用户场景，构建用况

- 收集需求时，系统功能和特点开始具体化。此时，容易弄不清不同类型的最终用户如何使用这些系统功能和特点。
- 场景描述了系统将如何被使用。
- 场景描述被称为用况（或用例，**use case**）

详细用况说明的结构

用况的组成部分	注　释
用况名称	以动词开始
范围	要设计的系统
级别	“用户目标”(基本流程)或“子功能”
主要参与者	调用系统以提供服务的参与者
涉众及其关注点	关注该用例的人，以及他们各自的需要
前置条件	用例启动前必须成立的条件
后置条件	用例结束后必须成立的条件
主成功场景	典型的、理想的成功场景
替换场景	其它可能的场景（成功或失败）
特殊需求	相关的非功能需求
技术和数据变元表	不同的I/O方法及数据格式
发生频率	当前用例的发生频率，可能影响调查、测试和实现的时间安排
其它问题	仍待解决或确认的问题

示例：POS系统收银用况详细说明-1

范围：下一代POS系统

级别：用户目标

主要参与者：收银员

涉众及其关注点：

收银员：准确、快速地完成收银操作...

顾客：快速完成付款并获得购物凭证以方便退货...

.....

前置条件：收银员经过认证、顾客是超市会员

后置条件：正确更新库存、正确计算税金.....

示例：POS系统收银用况详细说明-2

主成功场景：

1. 顾客携带商品到收银台付款
2. 收银员启动一次销售过程……
n. 顾客付款系统打印票据完成整个销售过程

替换场景：

a. ……顾客信用卡额度不足要求退货……

b. ……条码无法刷出使用键盘输入唯一码……

……

特殊需求：

1. 90%情况下信用卡刷卡响应时间小于30秒
2. 顾客能在1米范围内清楚看到单价和累计金额显示

……

示例：POS系统收银用况详细说明-3

技术与数据变元：

1. 商品ID获取可以通过扫描和键盘输入两种方式

2. 商品ID支持中国、欧洲及日本三种编码标准

……

发生频率：可能在16小时内不间断发生

其它问题：

1. 收银员下班后是否需要清理现金并进行结帐处理

2. 该超市是否可能在未来实行24小时营业

……

（二）面向过程的需求工程方法

- 1.需求获取方法
- 示例：考务处理系统
  - 1）确定stakeholders
    - 如：程序员等级考试
    - 客户可能是考试中心，但用户包括学生、在职考生、考试中心职员、管理人员等
    - 其他stakeholders还包括项目组成员及其上级领导等

（二）面向过程的需求工程方法

- 1.需求获取方法
- 示例：考务处理系统
  - 2）从不同视角，不同角度收集信息
    - 请考试中心老师介绍背景，如往年考生总数、考生分布、考务相关的基本制度等
    - 考生报名规则(先期要求、人数限制等)、退考规则
    - 从报考到发证的操作流程和规则，
    - 管理员的管理流程和规则
    - 了解客户对系统的期望：准确、访问速度快...
    - ……

（二）面向过程的需求工程方法

- 1.需求获取方法
- 示例：考务处理系统
  - 3）stakeholders协同合作收集需求
    - 会议：经过上阶段的分析整理，请客户和用户代表讨论不清晰不一致之处。
      - 如：进一步了解通常情况下的报名持续时间、是否按地区逐步开放报名和查分，高校人数的一般分布等—与性能设计密切相关
    - 问卷：可以对学生进行问卷调查，以明确需求。
    - 原型：是否有类似成熟系统，可结合系统演示进行需求调研

（二）面向过程的需求工程方法

- 1.需求获取方法
- 示例：考务处理系统
  - 4）创建用户场景，构建用况
    - 确定谁会直接使用该系统，即参与者（Actor）,选取其中一个参与者
    - 定义该参与者希望系统做什么，参与者希望系统做的每件事将成为一个用况
    - 对每件事来说，何时参与者会使用系统，通常会发生什么，这就是用况的基本过程
    - 描述该用况的基本过程

示例：考生报名用况详细说明-1

范围：**考务处理系统

级别：用户目标

主要参与者：已注册的考生

涉众及其关注点：

考务中心管理员：审核考生信息

考生：快速完成考试申请并获得准考证...

前置条件：考生已注册并登陆

后置条件：正确更新考生信息库、正确编排准考证号

示例：考生报名用况详细说明-2

主成功场景：

1. 考生注册登陆后准备申请考试
2. 考生在线提交考试申请
3. 考生在线支付考试报名费
4. 管理员收到考生申请信息及付费信息
5. 经管理员审核，两个信息均正确，自动生成准考证号，打印准考证寄给考生.
6. 报名成功的考生信息记入考生名册

替换场景：

a. 支付失败，告知考生支付失败，转入支付窗口；

b. 考生信息错误，告知考生信息错误，转入信息录入窗口；

C. 考生不符合考试要求，告知考生，并退费。

示例：考生报名用况详细说明-3

技术与数据变元：

1. 支持身份证确认
2. 准考证号支持二维码认证

发生频率：

其它问题：

- 软工作业
-
- 教务处需要开发一款手机应用程序“教室在哪儿？”给全校师生使用。请根据本校具体情况进行需求分析。
-

（二）面向过程的需求工程方法

2.需求分析方法

- 需求分析的步骤
  - 1）进行系统分析，确定对系统的综合要求
  - 2）分析系统的功能要求
  - 3）分析系统的数据要求
  - 4）抽象出并确立目标系统的逻辑模型

（二）面向过程的需求工程方法

2.需求分析方法

1）进行系统分析，确定对系统的综合要求

- 功能要求：从整个系统的角度提出系统整体功能要求；
- 性能要求：包括系统的相应时间、资源限制、数据精确性、系统适应性等；
- 运行要求：包括系统硬件环境、网络环境、系统软件、接口等的具体要求；
- 其他要求：安全保密、可靠性、可维护性、可移植性、可扩展性等等。

（二）面向过程的需求工程方法

2.需求分析方法

2）分析系统的功能要求

- 主要思想：抽象与自顶向下的逐层分解(控制复杂性的两个基本手段)
- 抽象：在每个抽象层次上忽略问题的内部复杂性，只关注整个问题与外界的联系
- 分解：将问题不断分解为较小的问题，直到每个最底层的问题都足够简单为止

（二）面向过程的需求工程方法

2.需求分析方法

3）分析系统的数据要求

- 主要指系统分析师根据用户的信息流抽象、归纳出系统所要求的数据定义、数据逻辑关系、输入/出数据定义、数据采集方式等

（二）面向过程的需求工程方法 Next

2.需求分析方法

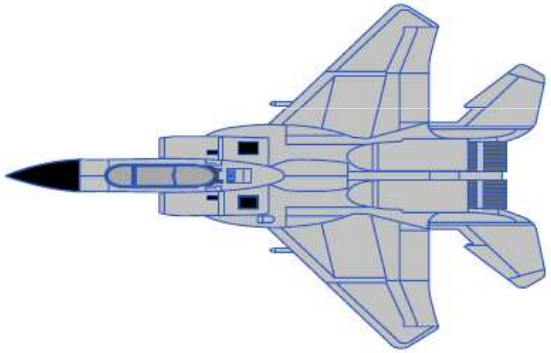
- 4）抽象出并确立目标系统的逻辑模型 Go
  - 模型的作用
    - 整个系统太复杂，难以一下子抓住，通过模型简洁地描述系统某个方面
    - 交流。（项目组成员之间，与客户）
    - 将系统体系结构归档
  - 模型建立的思路
    - 自顶向下、逐步求精
    - 自底向上、综合集成

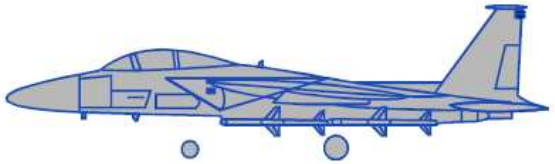
Model

- A model is a pattern, plan, representation (especially in miniature), or description designed to show the main object or workings of an object, system, or concept.

Modeling Back

- A model is a simplified representation of the actual system intended to promote understanding.





（二）面向过程的需求工程方法

2.需求分析方法

- **4）抽象出并确立目标系统的逻辑模型**
  - 模型的表述：模型语言
  - 模型语言=模型Model+表示法Notation
    - **Model:** 表示系统的结构
      - » 设计系统时可以在高层进行讨论，
      - » 而不用太早进入代码的细节
    - **Notation:** 以图、表、文字等将模型文档化

（二）面向过程的需求工程方法

2.需求分析方法

4）抽象出并确立目标系统的逻辑模型

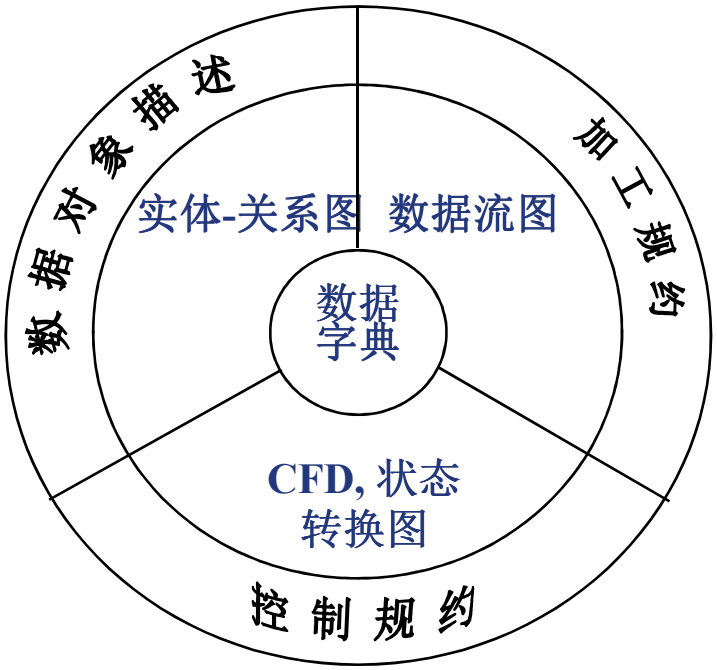
- 描述模型的方式：形式化描述和图示化描述。
  - 形式化描述方法非常精确、严谨，易于系统以后的实现，但难以掌握和理解，模型可读性差，往往只有专业人员才会使用，因而难于推广。
  - 图示化方法直观、自然，易于描述系统的层次结构、功能组成，且简单易学，通常还有工具软件支持，因而成为信息系统的主要描述工具，但这种方法的精确性和严谨性不够。

（二）面向过程的需求工程方法 Next

2.需求分析方法

- 4）抽象出并确立目标系统的逻辑模型
  - 数据流图（Data flow diagram ,DFD）
  - E-R图 (Entity Relationship diagram, ERD)
  - 控制流程图（Control Flow diagram ,CFD）
  - 状态转换图（Statement Transition diagram, STD)
  - 数据字典（Data Dictionary ,DD)

结构化分析模型的描述 Back



- 数据字典是模型的核心，它包含了软件使用和产生所有数据的描述
- 数据流图：用于功能建模，描述系统的输入数据流如何经过一系列的加工变换逐步变换成系统的输出数据流
- 实体—关系图（E-R图）：用于数据建模，描述数据字典中数据之间的关系
- CFD和状态转换图：用于行为建模，描述系统接收哪些外部事件，以及在外部事件的作用下的状态转换情况

数据流图——(1)基本元素 Next

- **Data Flow Diagram(简称DFD)**：描述输入数据流到输出数据流的变换(即加工)过程，用于对系统的功能建模，基本元素包括：

数据流(data flow)：由一组固定成分的数据组成，代表数据的流动方向 Go

加工(process)：描述了输入数据流到输出数据流的变换，即将输入数据流加工成输出数据流

文件(file)：使用文件、数据库等保存某些数据结果供以后使用。Go

数据输入的源点或数据输出的汇点(source or sink)：由一组固定成分的数据组成，代表数据的流动方向

源点或汇点 Back

- 存在于软件系统之外的人员或组织，表示软件系统输入数据的来源和输出数据的去向。
  - 例如，对一个考务处理系统而言
    - 考生向系统提供报名单(输入数据流)，所以考生是考试系统(软件)的一个源点
    - 考务处理系统要将考试成绩的统计分析表(输出数据流)传递给考试中心，所以考试中心是该系统的一个汇点
- 源或汇点用相同的图形符号表示
  - 当数据流从该符号流出时表示是源点
  - 当数据流流向该符号时表示是汇点
  - 当两者皆有时表示既是源点又是汇点

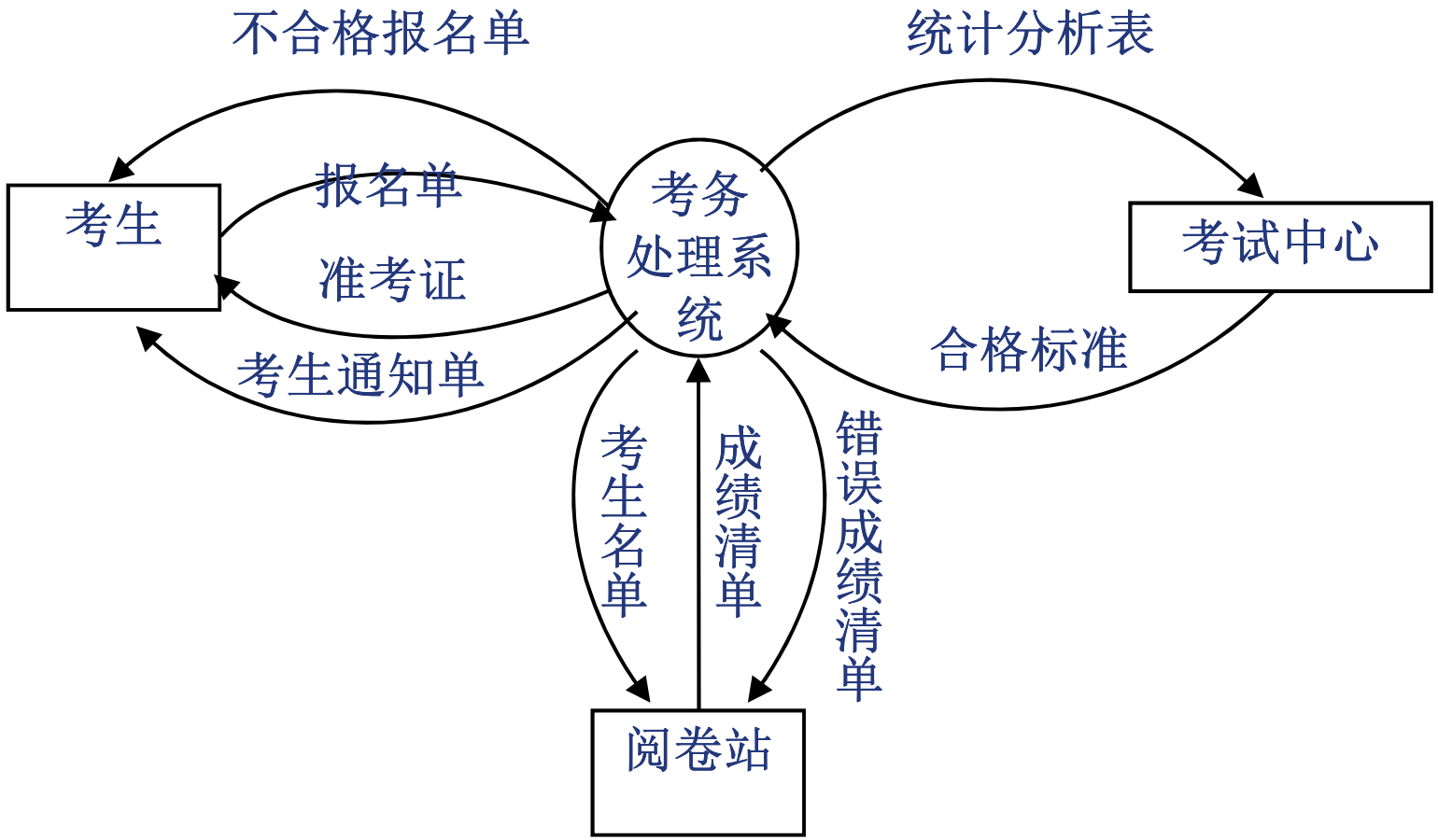
加工和文件 Back

- 加工○：描述输入数据流到输出数据流的变换
  - 每个加工用一个定义明确的名字标识
  - 至少有一个输入数据流和一个输出流
  - 可以有多个输入数据流和多个输出数据流
- 文件═：保存数据信息的外部单元
  - 每个文件用一个定义明确的名字标识
  - 由加工进行读写
  - DFD中称为文件，但在具体实现时可以用文件系统实现也可以用数据库系统等实现

数据流 Back

- 每个数据流用由一组固定成分的数据组成并拥有一个定义明确的名字标识
  - 如：运动会管理系统中，报名单(数据流)由队名、姓名、性别、参赛项目等数据组成
- 数据流的流向
  - 从一个加工流向另一个加工
  - 从加工流向文件(写文件)
  - 从文件流向加工(读文件)
  - 从源流向加工
  - 从加工流向汇点

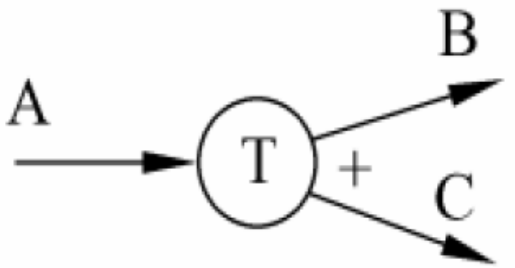
示例：考务管理系统DFD



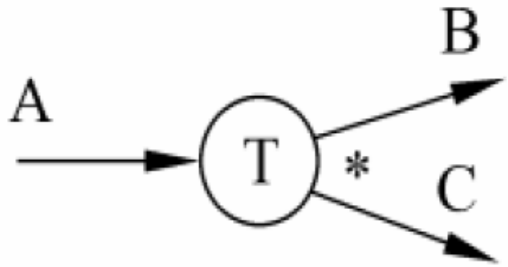
数据流图——（2）扩充符号 Next

- 描述一个加工的多个数据流之间的关系
  - 星号(*)：表示数据流之间存在“与”关系
    - 所有输入数据流同时存在时，才能进行加工处理
    - 或加工处理的结果是同时产生所有输出数据流
  - 加号(+)：表示数据流之间存在“或”关系
    - 至少存在一个输入数据流时才能进行加工处理
    - 或加工处理的结果是至少产生一个输出数据流
  - 异或(⊕)：表示数据流之间存在“异或”(互斥)关系
    - 必须存在且仅存在一个输入数据流时，才能进行加工处理
    - 或加工处理的结果是产生且仅产生一个输出数据流

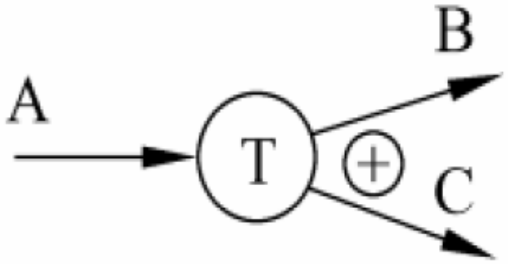
数据流图——（2）扩充符号



数据 A 变换成 B 或 C，或 B 和 C

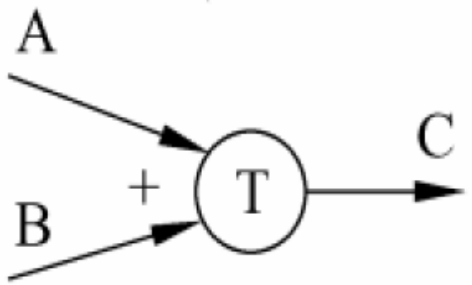


数据 A 变换成 B 和 C

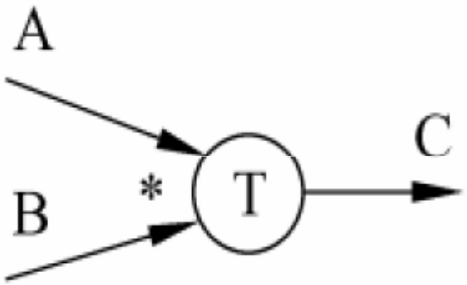


数据 A 变换成 B 或 C，但不能变换成 B 和 C

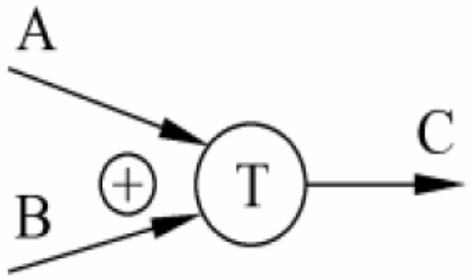
数据流图——（2）扩充符号



数据 A 或 B，或 A 和 B 同时输入变换成 C



数据 A 和 B 同时输入才能变换成数据 C



只有数据 A 或只有数据 B(但不能 A、B 同时) 输入时变换成 C

数据流图——（3）分层

- 分层数据流图：
  - 根据自顶向下逐层分解的思想将数据流图画成层次结构
  - **George Miller**在著名的论文“神奇的数字7加减2：我们处理信息的能力的某种限制”中指出：人们在一段时间内的短期记忆似乎限制在**5～9**件事情之内
  - 每个层次画在独立的数据流图中，加工个数可大致控制在“**7**加减**2**”的范围中

数据流图——（3）分层

- 顶层图：只有代表整个软件系统的1个加工，描述了软件系统与外界(源或汇点)之间的数据流
- 1层图：顶层图中的加工经分解后的图称为1层图(只有1张)
- 中间层：中间层图中至少有一个加工(也可以有多个)在下层图中分解成一张子图
- 底层图：处于最底层的图称为底层图，其中所有的加工不再分解成新的子图

数据流图——（3）分层

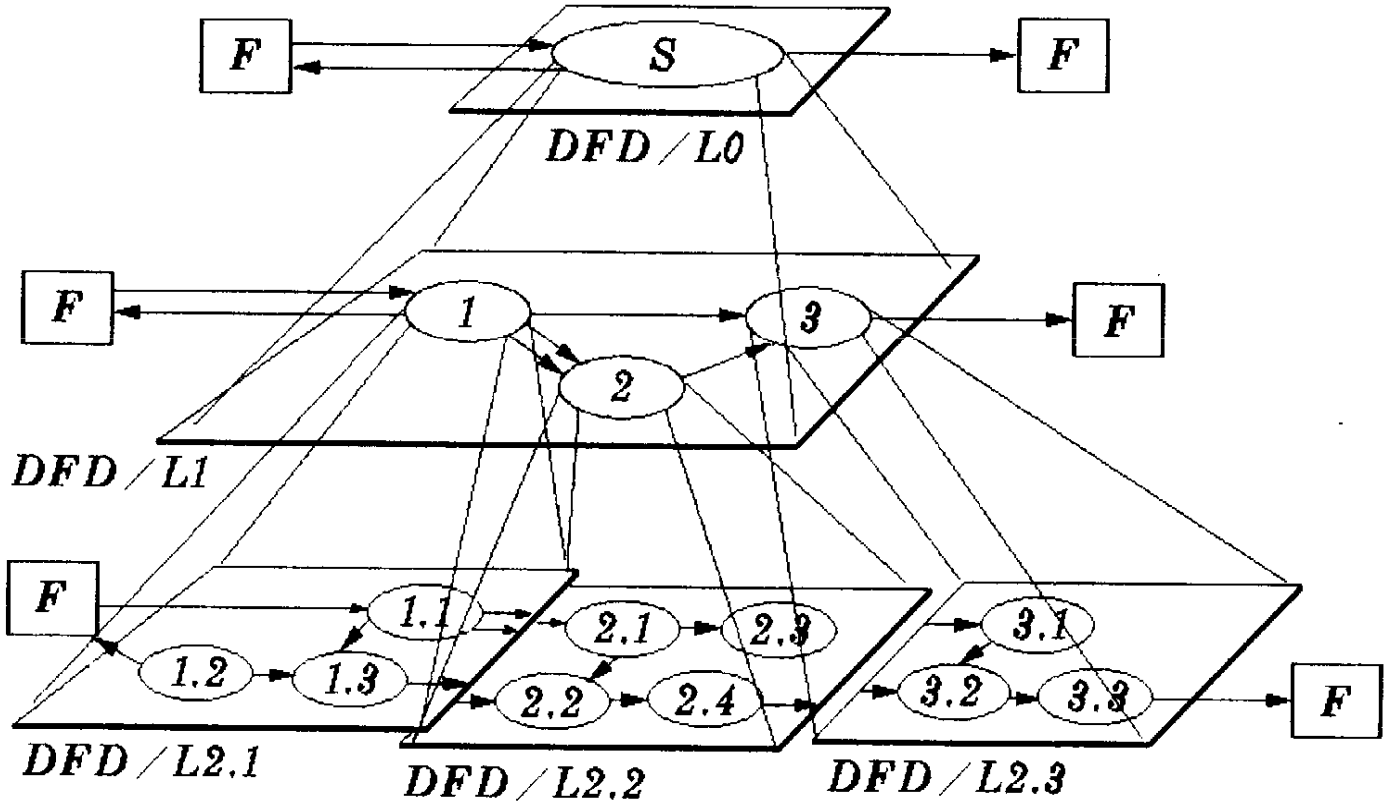


图 2.9　分层数据流图

数据流图——（4）示例

- 一个简化的水平考试的考务处理系统
  - 分成多个级别，如初级程序员、程序员、高级程序员、系统分析员等，凡满足一定条件的考生都可参加某一级别的考试
  - 考试的合格标准将根据每年的考试成绩由考试中心确定
  - 考试的阅卷由阅卷站进行，因此，阅卷工作不包含在软件系统中

数据流图——（4）示例

1.需求获取：考务系统的流程

- 1．对考生送来的报名单进行检查
- 2．对合格的报名单编好准考证号后将准考证送给考生，并将汇总后的考生名单送给阅卷站
- 3．对阅卷站送来的成绩清单进行检查，并根据考试中心制订的合格标准审定合格者
- 4．制作考生通知单送给考生
- 5．进行成绩分类统计(按地区、年龄、文化程度、职业、考试级别等分类)和试题难度分析，产生统计分析表

数据流图——（4）示例

- 2.需求获取和分析：考务系统的相关数据构成
  - 报名单＝地区＋序号＋姓名＋文化程度＋职业＋考试级别＋通信地址
  - 正式报名单＝准考证号＋报名单
  - 准考证＝地区＋序号＋姓名＋准考证号＋考试级别＋考场
  - 考生名单＝｛准考证号＋考试级别｝
    其中｛｝表示其内容重复多次
  - 考生名册＝正式报名单
  - 统计分析表＝分类统计表＋难度分析表
  - 考生通知单＝准考证号＋姓名＋通信地址＋考试级别＋考试成绩＋合格标志

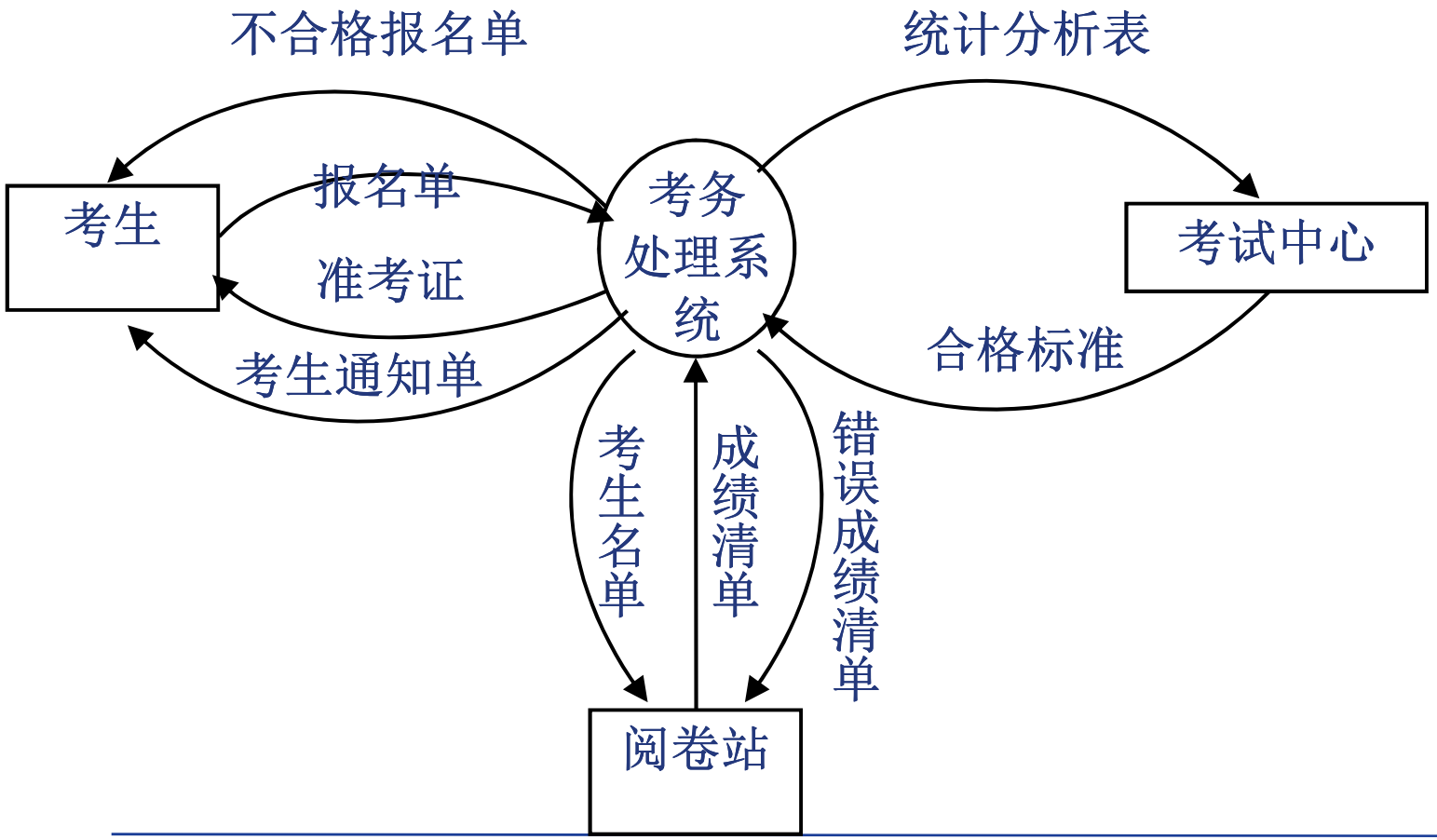
数据流图——（4）示例

3.需求分析：画出系统的顶层数据流图

- 确定源点汇点：考生、阅卷站和考试中心
  - 它们都既是源点又是汇点
- 顶层图唯一的加工：软件系统(考务处理系统)
- 确定数据流：系统的输入/输出信息
  - 输入数据流：报名单(来自考生)、成绩清单(来自阅卷站)、合格标准(来自考试中心)
  - 输出数据流：准考证(送往考生)、考生名单(送往阅卷站)、考生通知书(送往考生)、统计分析表(送往考试中心)
  - 额外的输出流(考虑系统的健壮性)：不合格报名单(返回给考生)，错误成绩清单(返回给阅卷站)
- 顶层图通常没有文件

数据流图——（4）示例

❖3.需求分析：考务处理系统顶层图

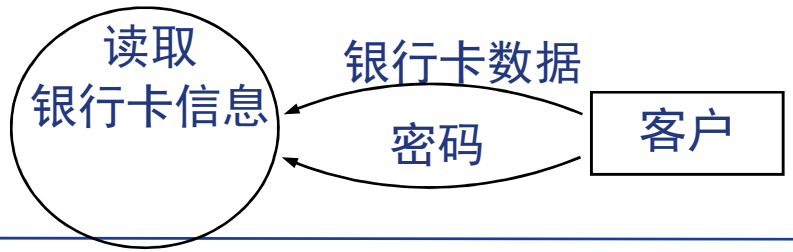


数据流图——（4）示例

- 4.需求分析：考务处理系统顶层图分解
  - 1）确定加工：将父图中某加工分解而成的子加工
    - 根据功能分解来确定加工：将一个复杂的功能分解成若干个较小的功能，较多应用于高层DFD中的分解
    - 根据业务处理流程确定加工：分析父图中待分解加工的业务处理流程，业务流程中的每一步都可能是一个子加工
    - 特别要注意在业务流程中数据流发生变化或数据流的值发生变化的地方，应该存在一个加工，例如：

数据流图——（4）示例

- 4.需求分析：考务处理系统顶层图分解
  - 2）确定数据流
    - 在父图中某加工分解而成的子图中，父图中相应加工的输入/输出数据流都是且仅是子图边界上的输入/输出数据流
    - 分解后的子加工之间应增添相应的新数据流表示加工过程中的中间数据
    - 如果某些中间数据需要保存以备后用，那么可以成为流向文件的数据流
    - 同一个源或加工可以有多个数据流流向一个加工，如果它们不是一起到达和一起加工的，那么可以将它们分成若干个数据流，例如：



数据流图——（4）示例

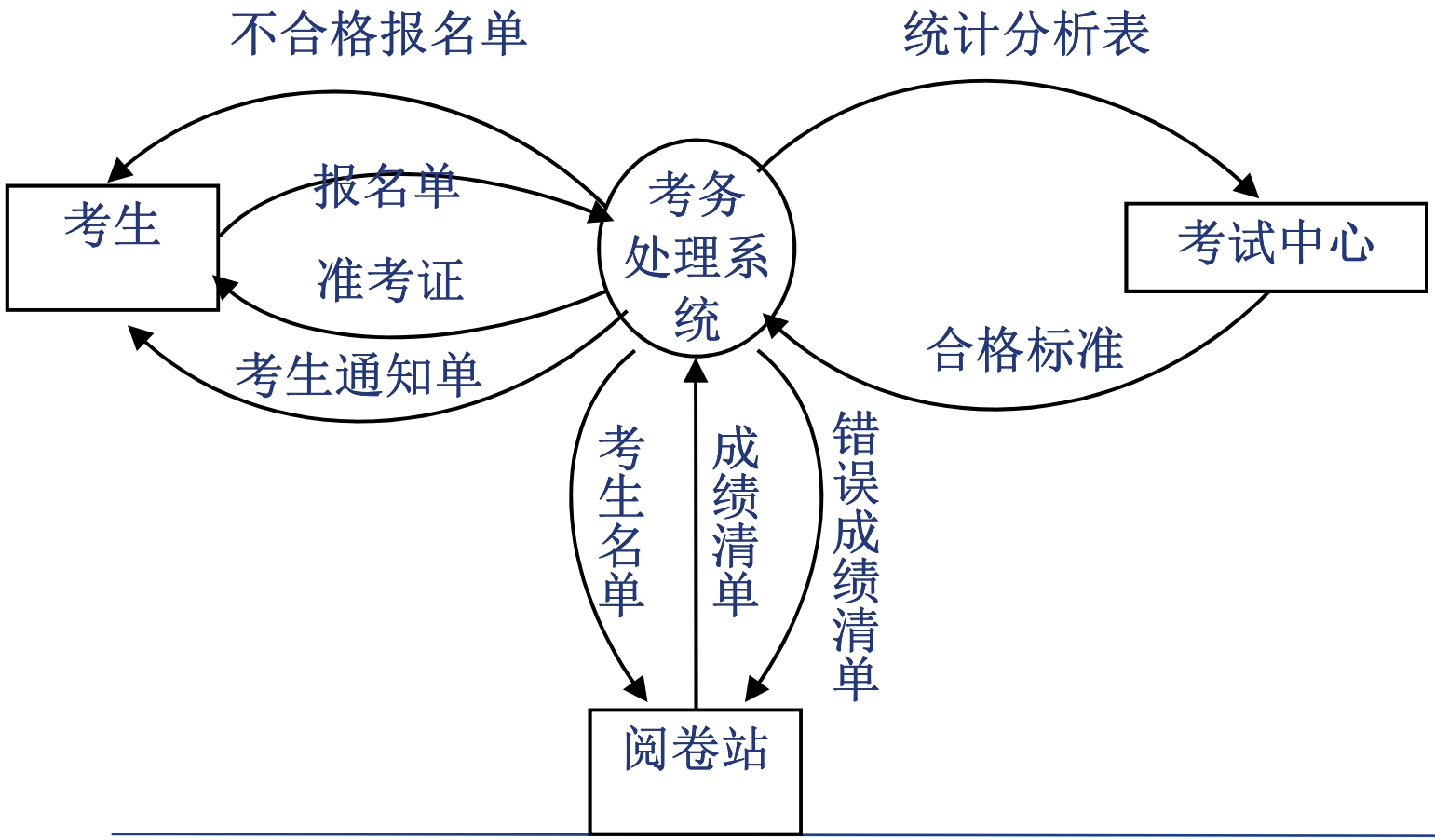
- 4.需求分析：考务处理系统顶层图分解
  - 3）确定文件
    - 如果父图中该加工存在读写文件的数据流，则相应的文件和数据流都应画在子图中
    - 在分解子图中，如果需要保存某些中间数据以备后用，则可以将这些数据组成一个新的文件
    - 新文件(首次出现的文件)至少应有一个加工为其写入记录，同时至少存在另一个加工来读该文件的记录
    - 注意：从父图中继承下来的文件在子图中可能只对其进行读，或只进行写

数据流图——（4）示例

- 4.需求分析：考务处理系统顶层图分解
  - 4）确定源点和汇点
    - 1层图和其它子图中通常不必画出源点和汇点
    - 有时为了提高可读性，可以将顶层图中的源和汇点画在1层图中

数据流图——（4）示例

- 考务处理系统顶层图



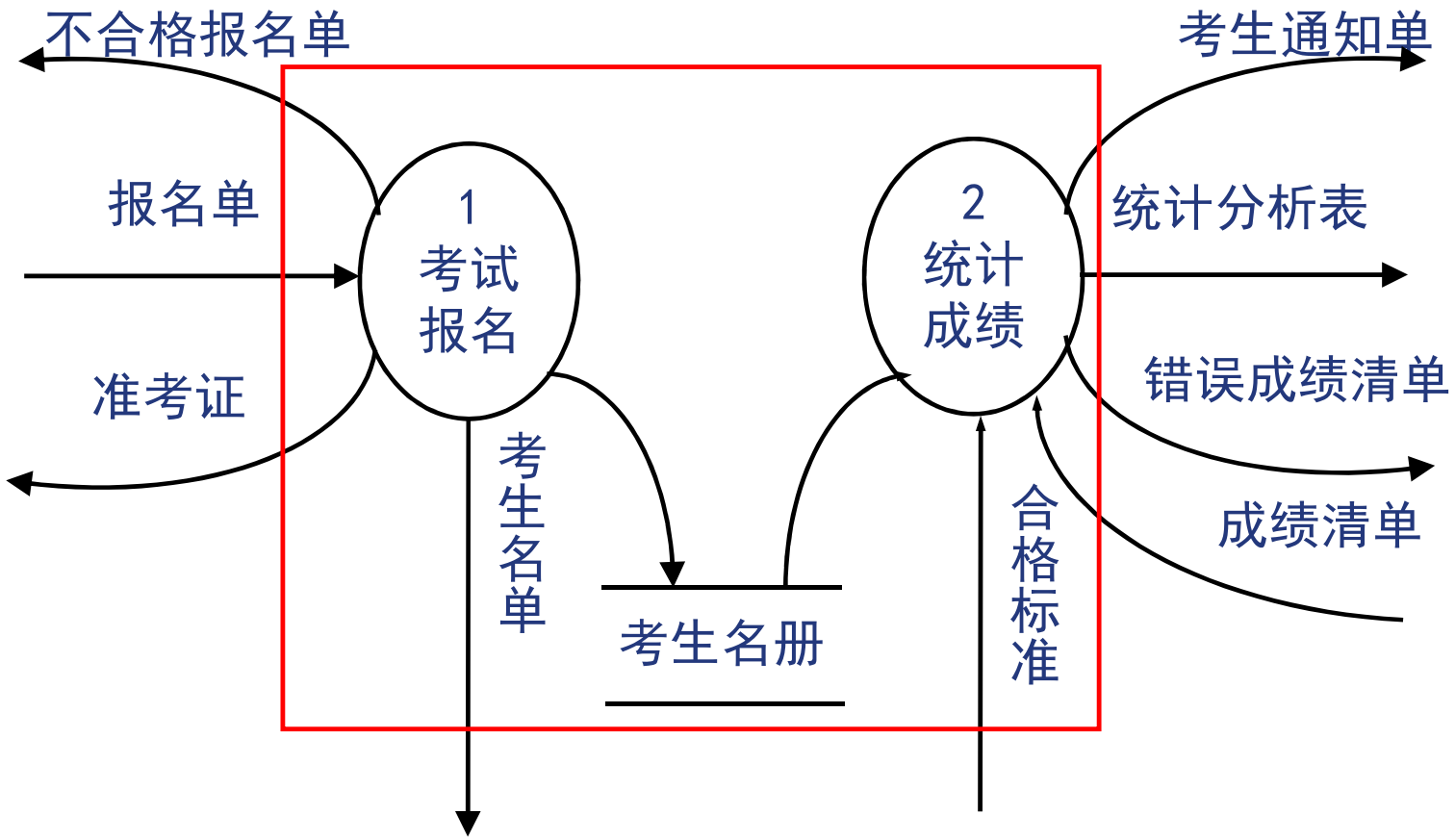
数据流图——（4）示例

考务处理系统1层图

- 根据功能分解方法识别出两个加工：考试报名、统计成绩
- 数据流
  - 继承顶层图中的输入数据流和输出数据流
  - 定义二个加工之间的数据流：由于这二个加工分别在考试前后进行，因此登记报名单所产生的结果“考生名册”应作为文件保存以便考试后由统计成绩加工引用

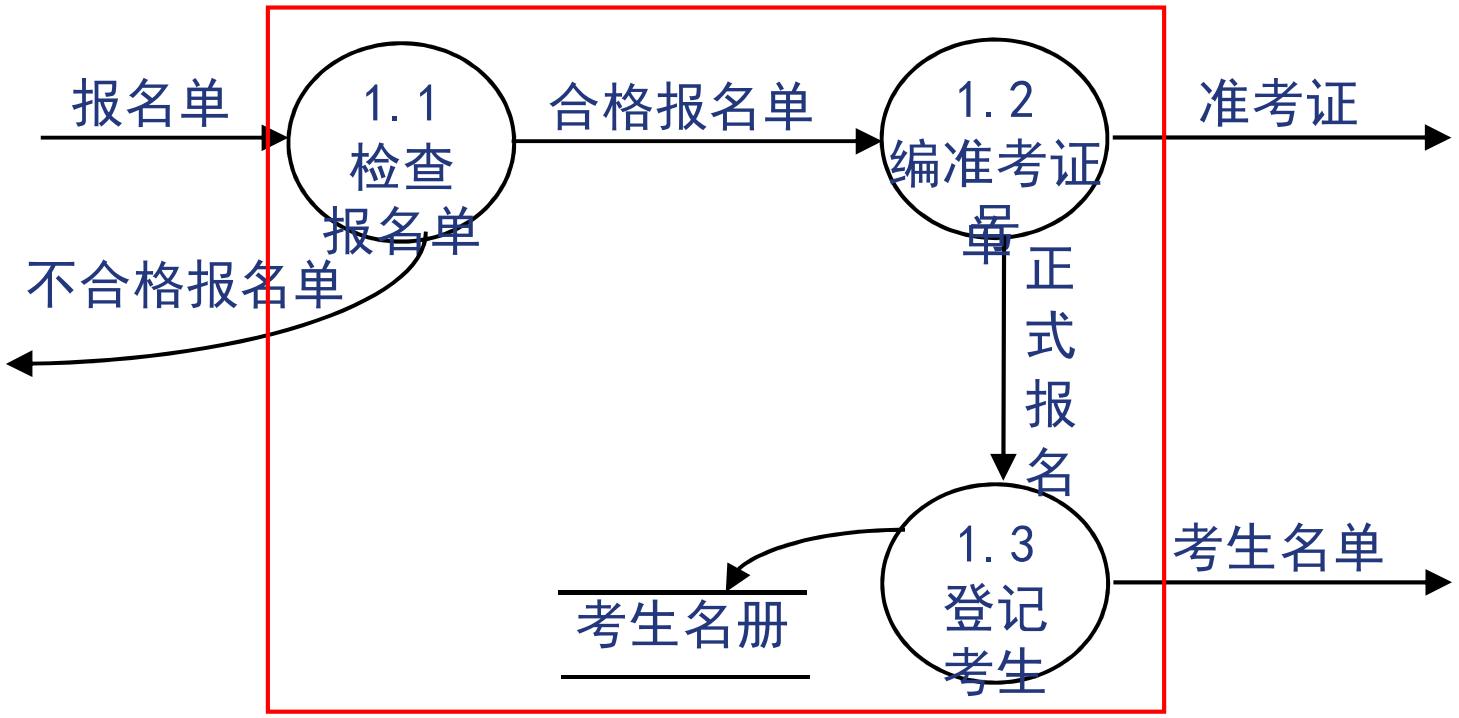
数据流图——（4）示例

考务处理系统1层图



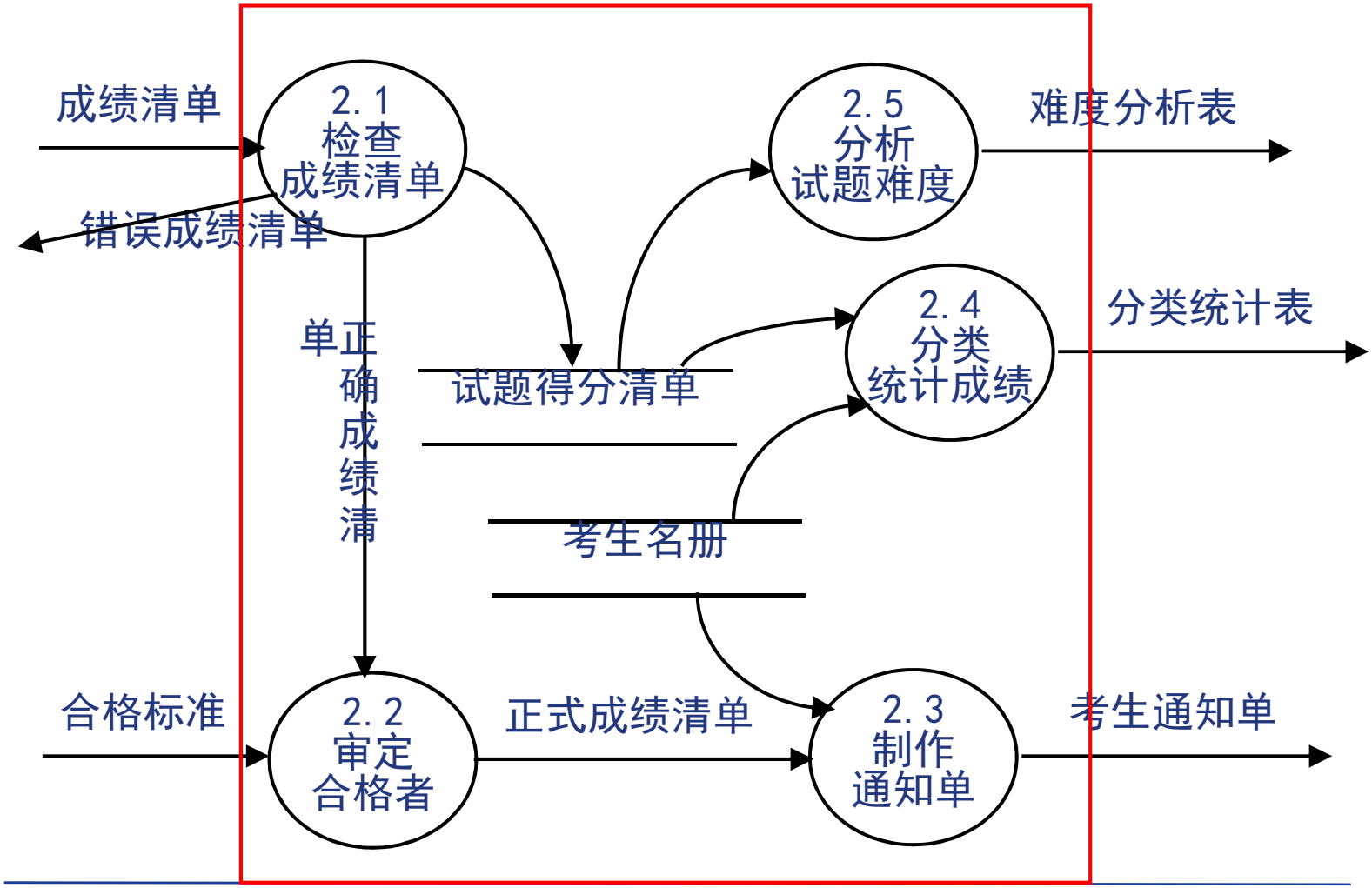
数据流图——（4）示例

考务处理系统2层图：考试报名



数据流图——（4）示例

考务处理系统2层图：成绩统计



数据流图——（5）总结 Back

- 总结：画分层数据流图的步骤
  - 1. 画系统的输入和输出
  - 2. 画系统内部
  - 3. 画加工内部
  - 4. 重复第3步，直至每个尚未分解的加工都足够简单(即不必再分解)
  - 5. 数据流图完成后，需要检查图中是否存在错误或不合理(不理想)的部分
    - 一致性：分层DFD中不存在矛盾和冲突
    - 完整性：分层DFD本身的完整性，即是否有遗漏的数据流、加工等元素

（二）面向过程的需求工程方法 Next

2.需求分析方法

- 4）抽象出并确立目标系统的逻辑模型
  - 数据流图（Data flow diagram ,DFD）
  - E-R图 (Entity Relationship diagram, ERD)
  - 控制流程图（Control Flow diagram ,CFD）
  - 状态转换图（Statement Transition diagram, STD)
  - 数据字典（Data Dictionary ,DD)

数据流图——（3）分层

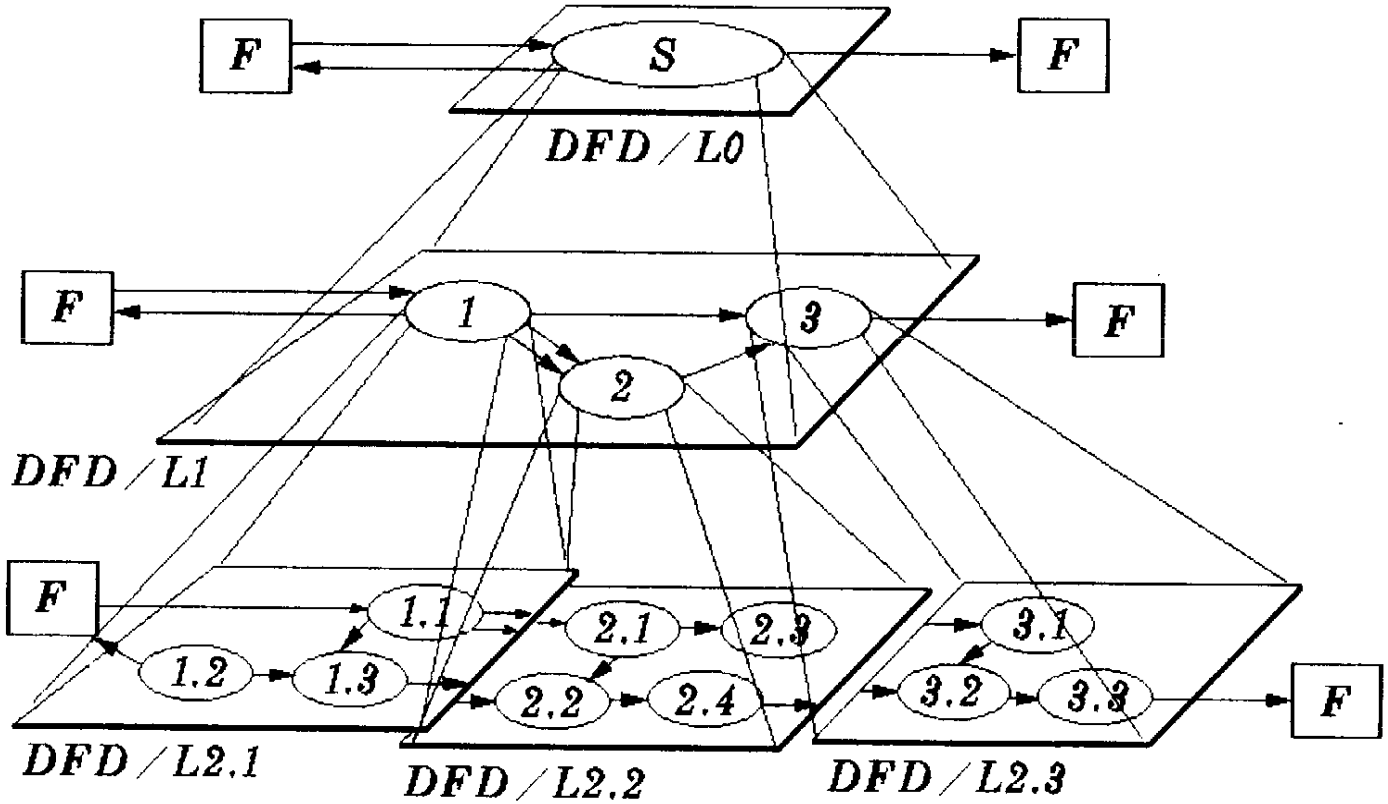


图 2.9　分层数据流图

实体关系图——（1）概念

- 实体关系图（**ER**图）用来建立数据模型 （ 1976年，P. P. S. Chen提出）
- ER建模主要关注的对象是：
  - 要处理的数据对象是什么
  - 数据对象的组成形式
  - 数据对象的属性
  - 数据对象的位置和相互关联关系
  - 对数据对象的加工和处理
- 因此，数据模型当中包含三种相互关联的信息
  - 实体 （数据对象）
  - 属性
  - 关系

实体关系图——（1）概念

（1）实体：

- 是客观世界中存在的、且可相互区分的事物。它可以是人或物，也可以是具体事物或抽象事物。
  - 例如：教师、学生、课程是实体。
- 实体用矩形框表示，如：

实体关系图——（1）概念

（2）联系：客观世界中的事物彼此之间有联系，描述实体与实体之间的关系。联系有三种：

- 1：1（一对一联系）

  例如：实体“校长”与“大学”之间的联系为“1：1”

- 1：N（一对多联系）

  例如：实体“学校”与“院系”之间的联系为“1：N”

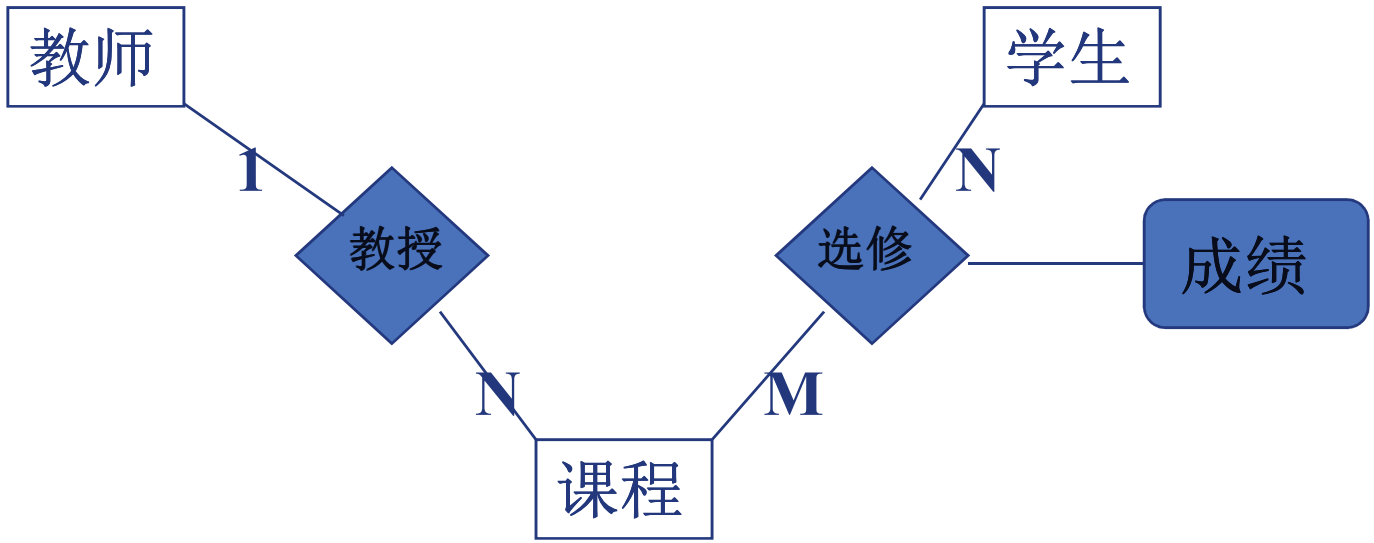
- M：N（多对多联系）

  例如：实体“学生”与“课程”之间的联系为“M：N”

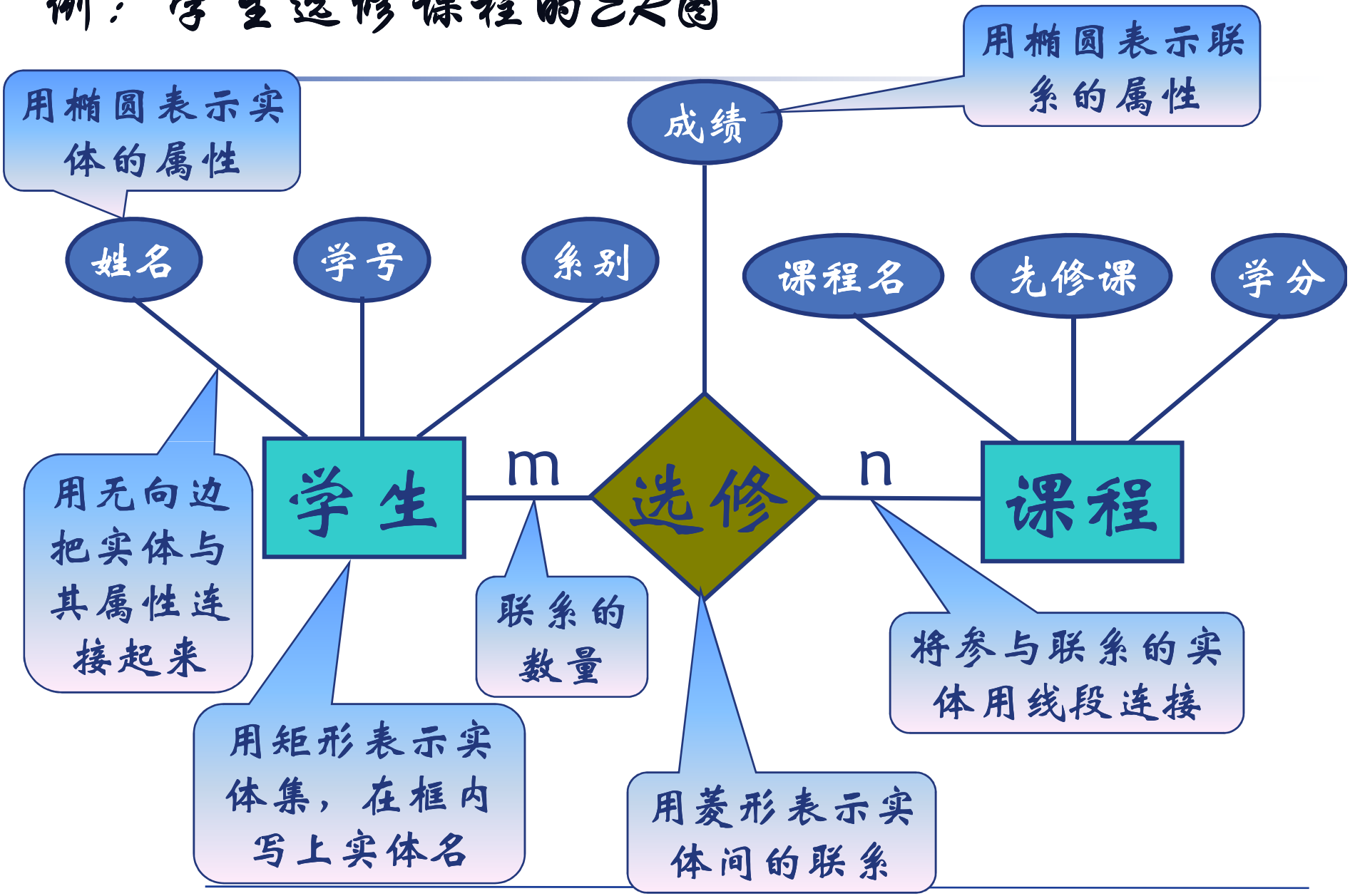
- 联系用菱形框表示，如：

实体关系图——（1）概念

（3）属性：属性是实体或联系所具有的性质。
通常一个实体或联系由若干属性来刻画。



例：学生选修课程的ER图



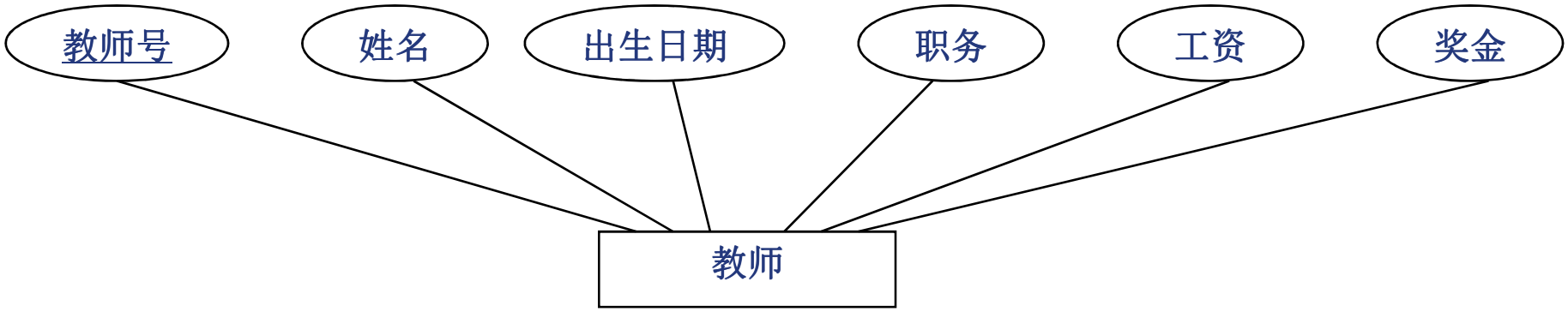
实体关系图——（2）应用

- 应用于以数据库应用为中心的分析过程
- 实现数据库应用的主要过程
  - 用传统软件工程方法，分析数据流，得到数据对象，确定数据字典
  - 分析数据间联系，构建实体关系图
  - 定义数据库表结构
  - 编制与调试应用程序
  - 数据库试运行
    - 功能测试
    - 性能测试（时空代价）

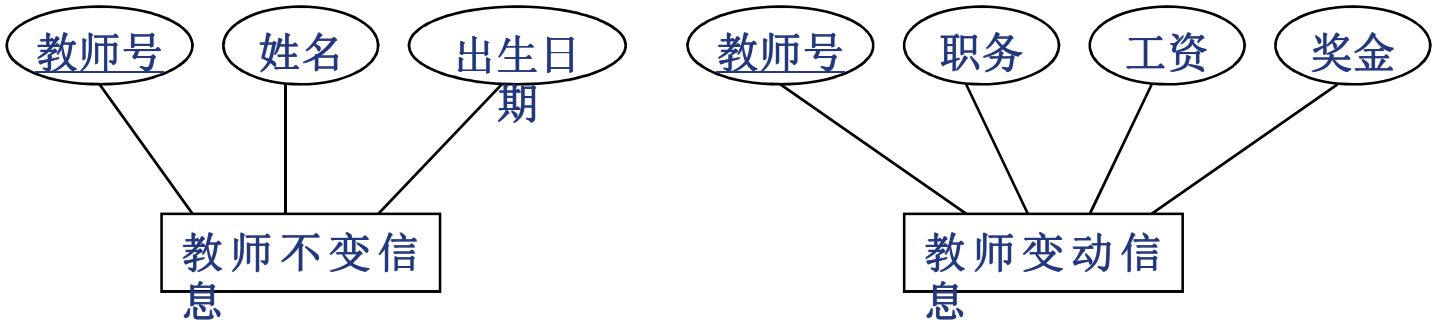
实体关系图——（3）构建

- 1.根据需求获取的资料，得到数据对象（实体）
- 2.确定实体间联系
- 3.构建局部实体关系图
- 4.对实体关系图进行精化，包括实体类型、联系类型和属性的分裂、合并、增删等等

实体关系图——（3）构建



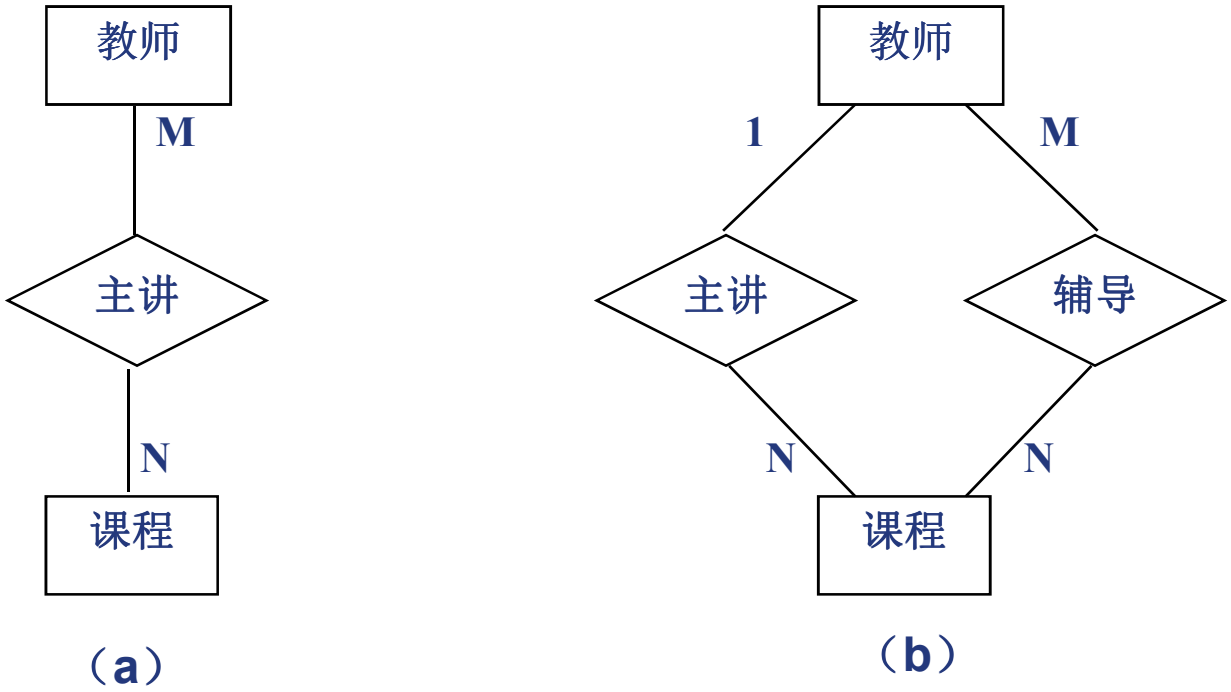
（a）



（b）

实体类型的垂直分裂

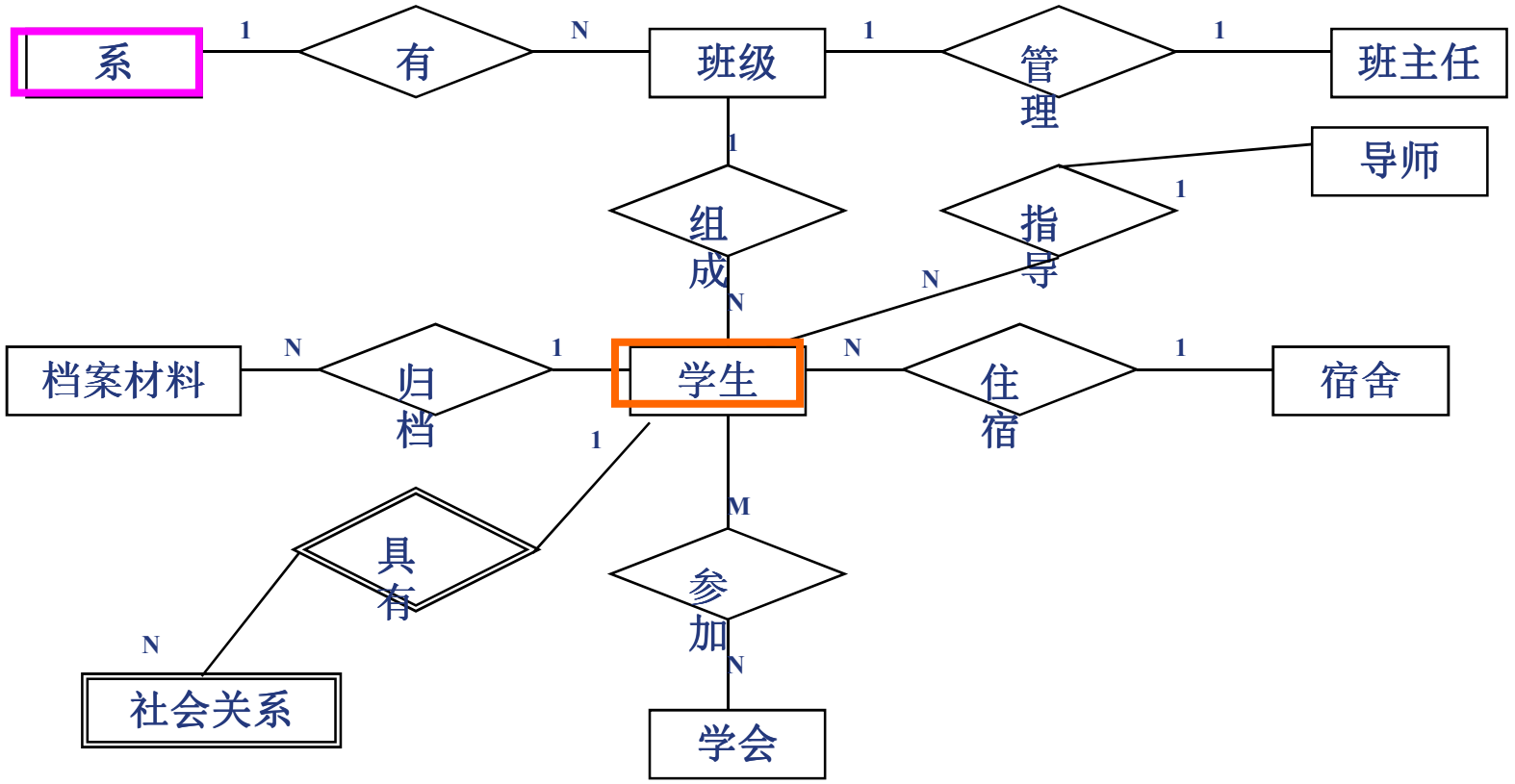
实体关系图——（3）构建



联系类型的分裂

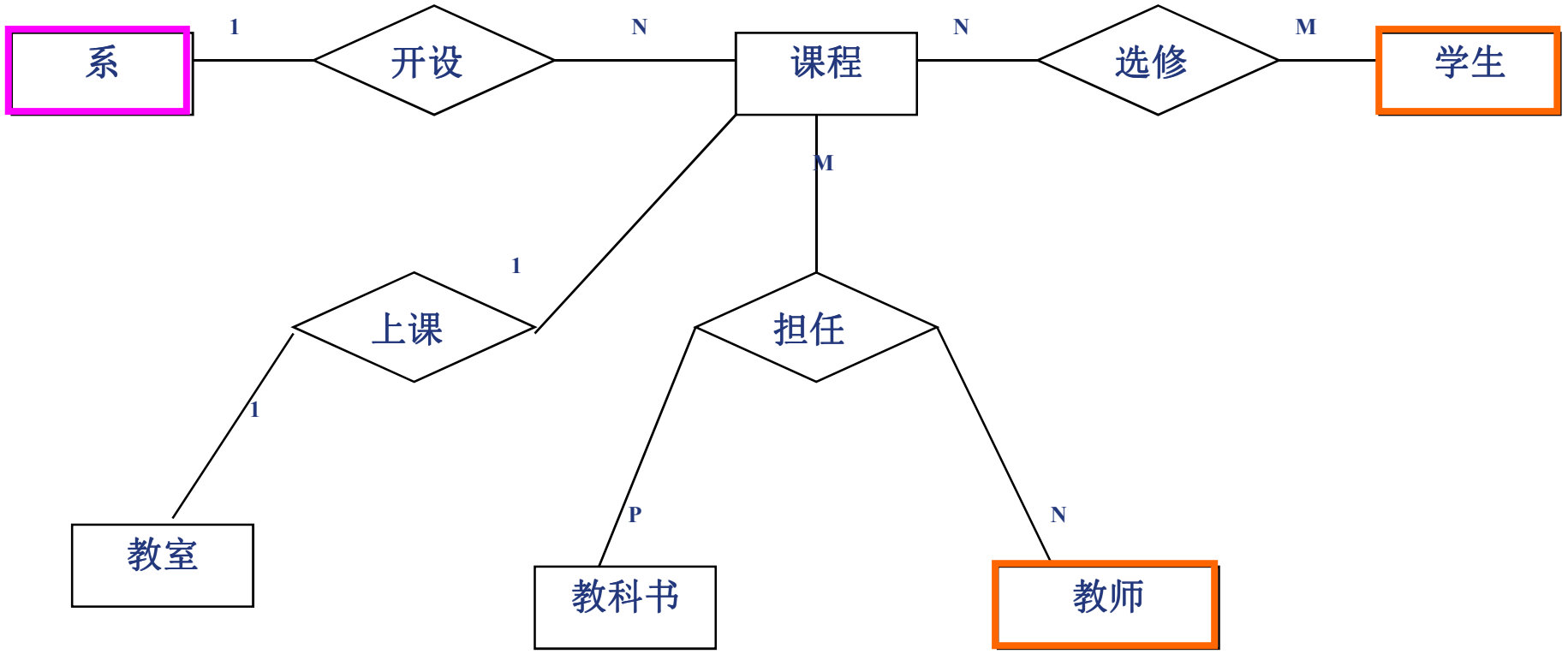
实体关系图——（3）构建

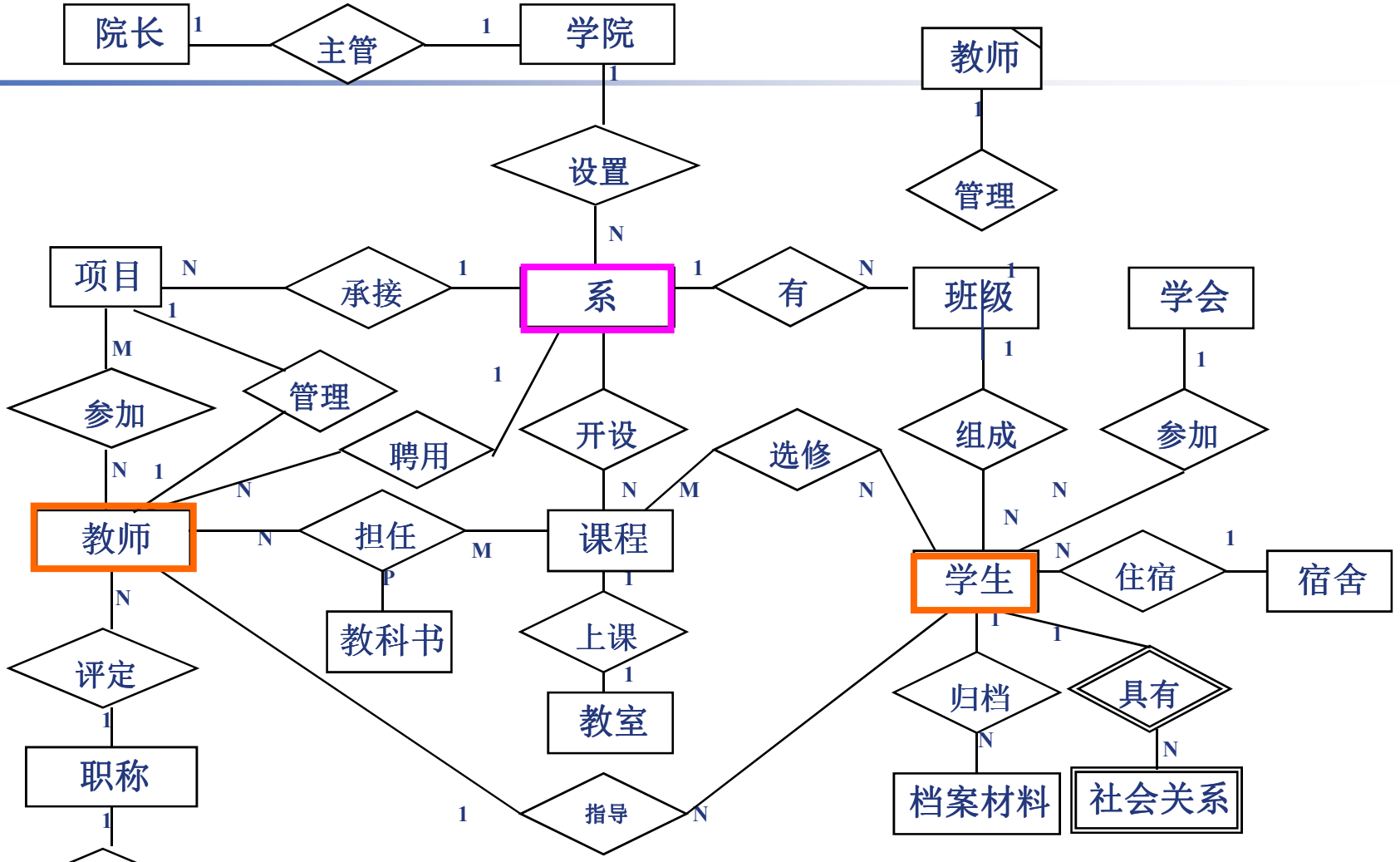
局部ER图：学籍管理部分



实体关系图——（3）构建

局部ER图：课程管理部分





局部ER图合并成为教学管理E-R图

实体关系图——（4）转换到关系模型

仓库（仓库号，仓库名，地址）

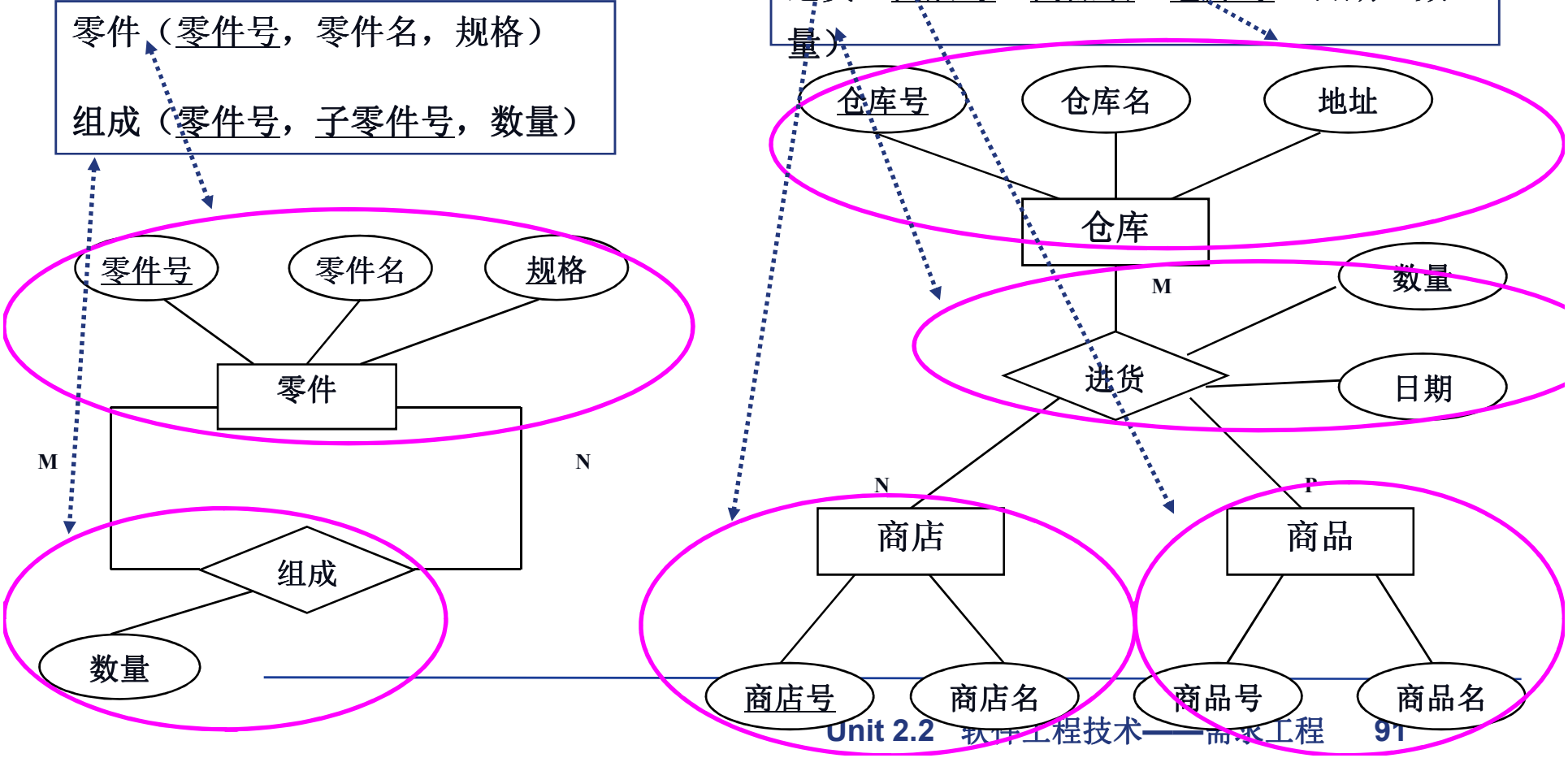
商店（商店号，商店名）

商品（商品号，商品名）

进货（商店号，商品名，仓库号，日期，数量）

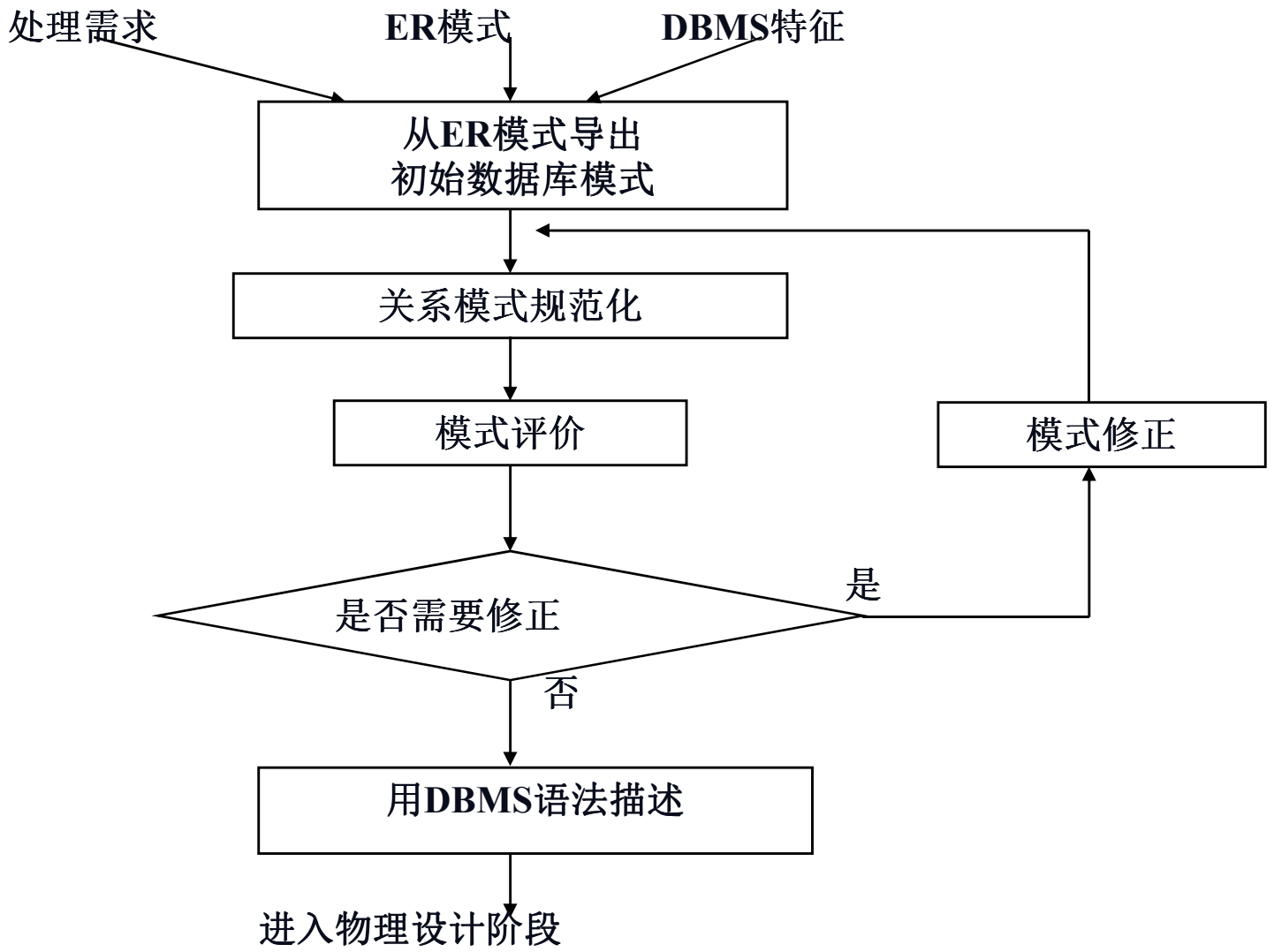
零件（零件号，零件名，规格）

组成（零件号，子零件号，数量）



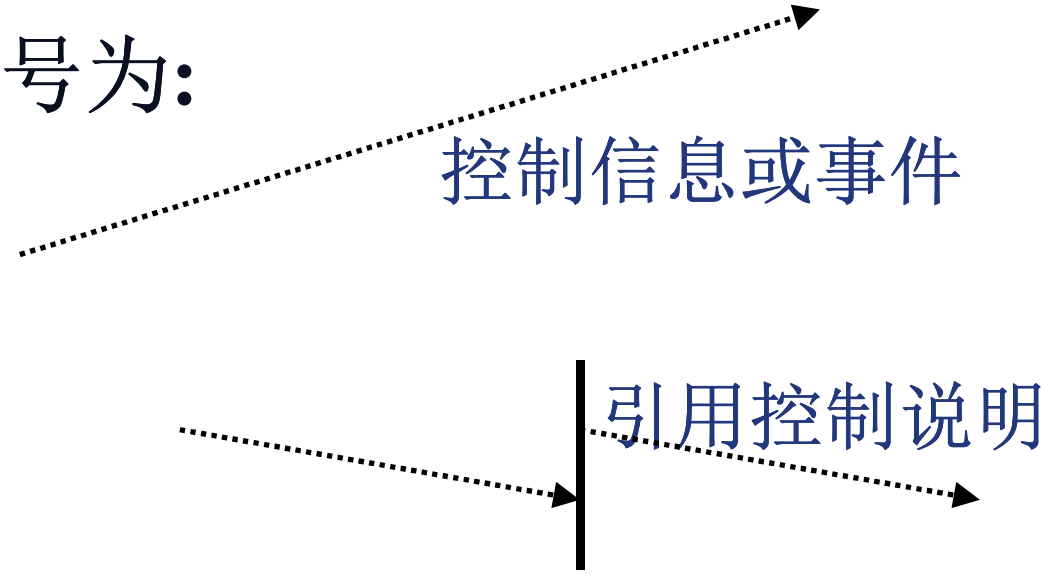
采用ER方法的逻辑设计步骤

Back



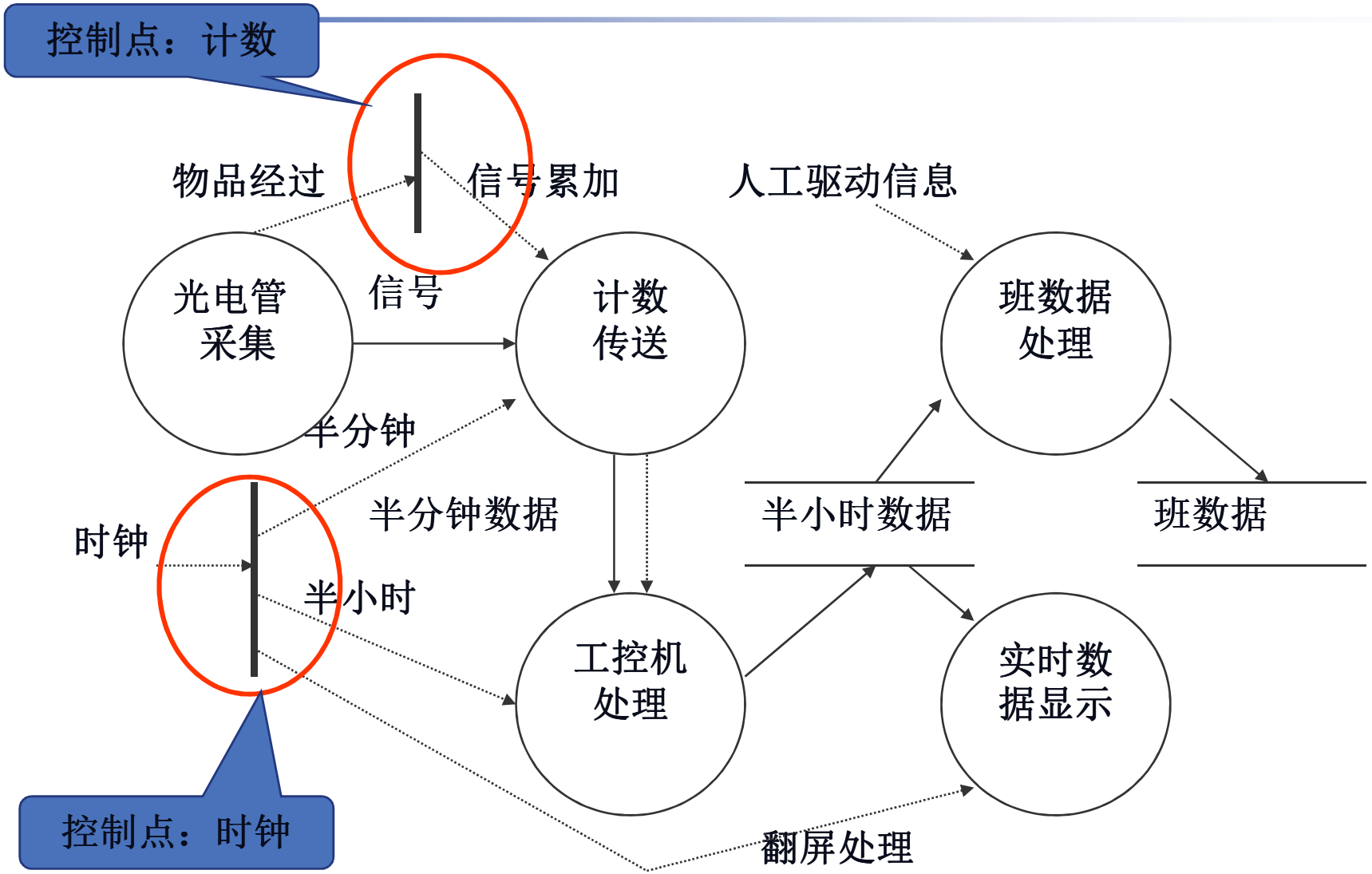
控制流图 CFD

- 控制流程图（**Control Flow diagram ,CFD**）与**DFD**类似
- 适合实时系统的分析
- 表示控制流（相对**DFD**是数据流）
- 其符号为:



DFD和CFD的结合应用

Back



状态转换图——（1）概念

- 状态转换图是描述系统的状态如何，根据外部的信号进行推移的一种图形表示。
- 与CFD的不同，更多的是反映控制点的状态，CFD反映的控制流和控制点。
- 在面向对象的需求分析中也使用状态转换图。
- 状态：
  - 状态代表系统的一种行为模式，规定了系统对事件的响应方式。
  - 状态分为：初态（初始状态），终态（最终状态）和中间状态。
  - 一张状态图中只有一个初态，但可以有0到多个终态。

状态转换图——（1）概念

- 事件：
  - 是在某个特定时刻发生的外部事情，它会引起系统做某动作或（和）从一个状态转换到另一个状态。
- 符号：

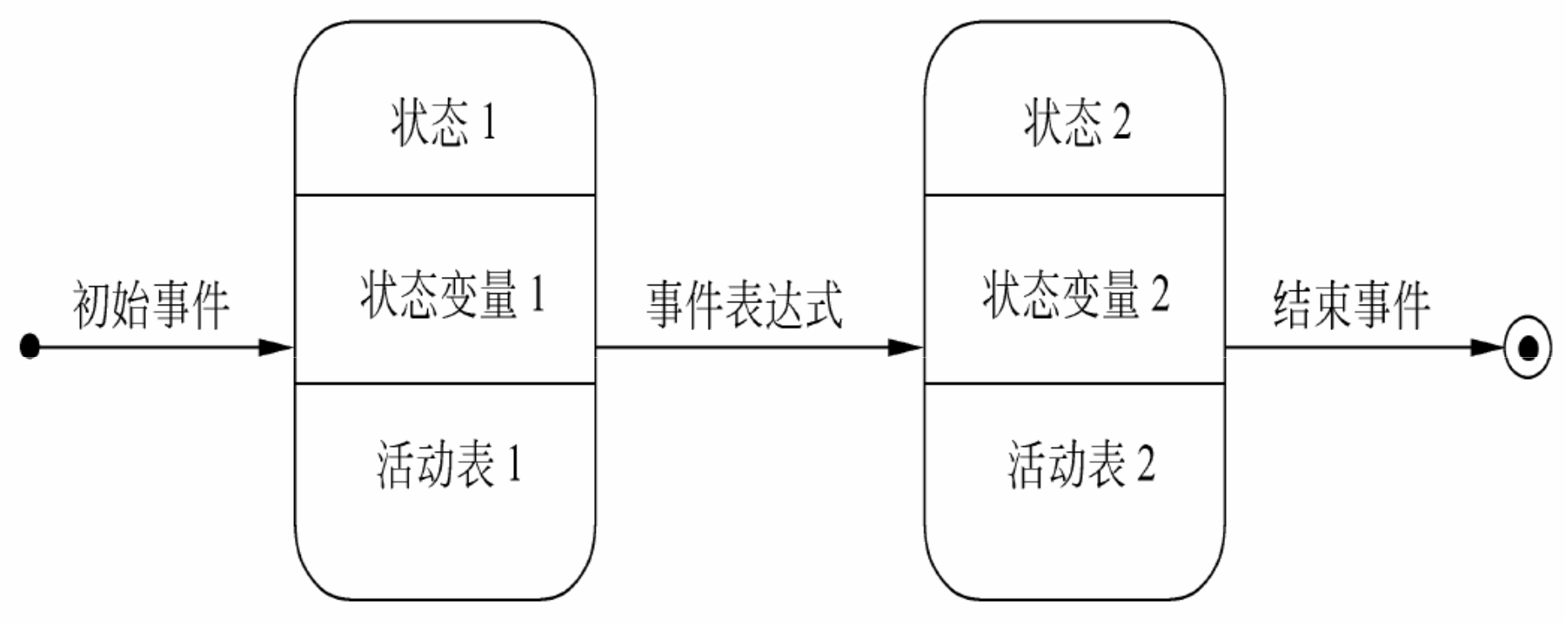

从一种状态到另一种状态的迁移


系统状态

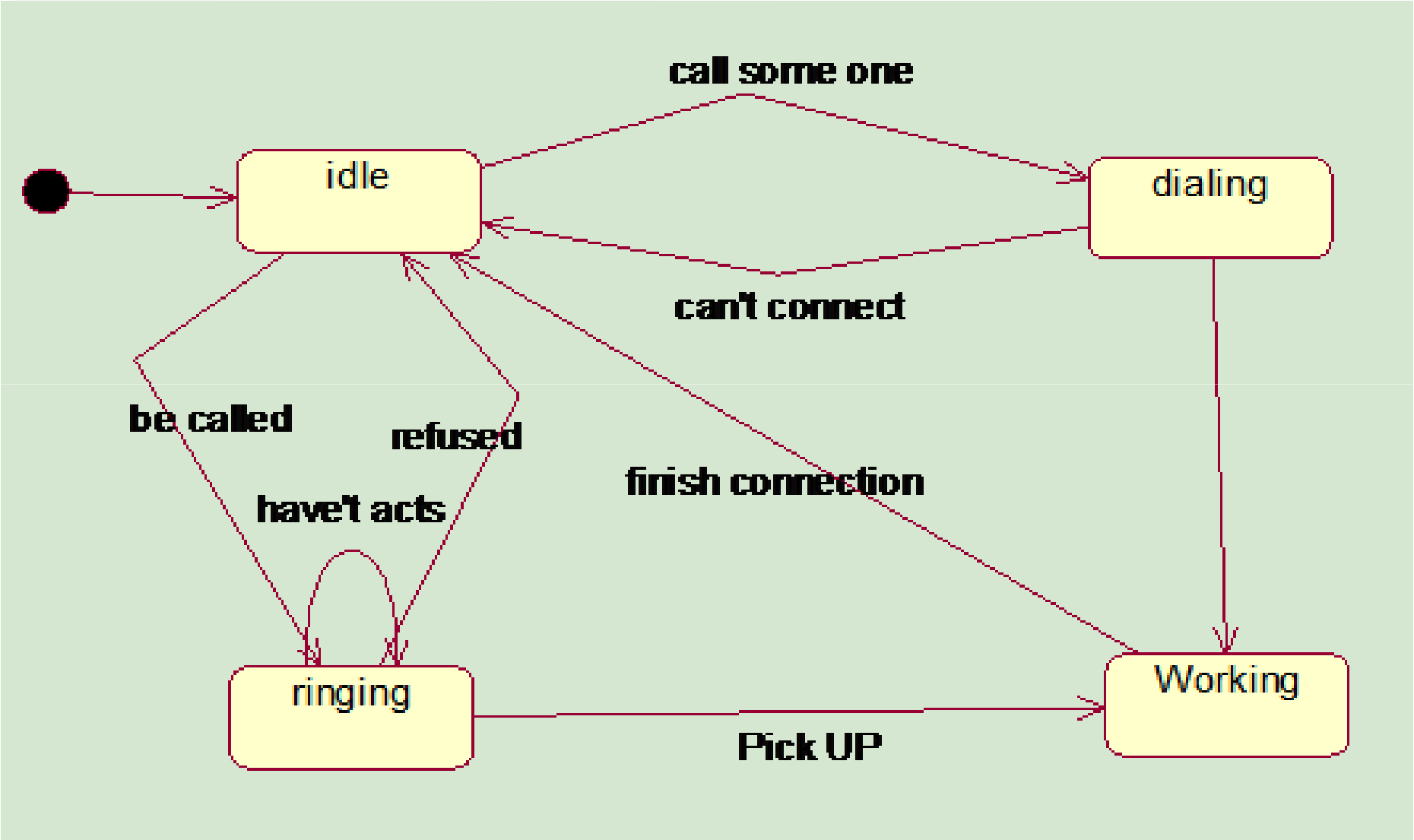

实心圆，表示初态


同心圆，表示终态

状态转换图——（2）构建



状态转换图——（3）示例 Back



数据字典——（1）概念

- 数据流图上所有成分的定义和解释的文字集合就是数据字典。
  - 数据流图只能给出系统逻辑功能的一个总体框架，但缺乏详细、具体的内容。
  - 数据字典能对数据流图的各种成分起注释、说明的作用，给这些成分赋以实际的内容。
  - 数据流图与数据字典是密不可分的，两者结合起来构成软件的逻辑模型(分析模型)

数据字典——（1）概念

- 数据字典的用途
  - 分析阶段的交流工具
  - 包含控制信息
  - 数据库设计的基础
- 数据字典要求
  - 1）完整性
  - 2）一致性
  - 3）可用性

数据字典——（2）构成

- 数据字典由字典条目组成，每个条目描述**DFD**中的一个元素
- 数据字典条目包括：数据流、文件、数据项(组成数据流和文件的数据)、加工、源或汇点

数据字典——（3）构成

- 不同的开发组织或团队可以根据项目的需要定义字典条目的描述内容
- 字典条目中的描述内容主要包括
  - **DFD**元素的基本信息(名称、别名、简述、注解)
  - 定义(数据类型、数据组成)
  - 使用特点(取值范围、使用频率、激发条件)
  - 控制信息(来源、去向、访问权限)等

数据字典——（3）构成

- 数据字典的描述符号

符　号	名　称	举　　例
=	定义为	x=… 表示x由…组成
+	与	a+b　表示a和b
[…，…]	或	［a，b］表示a或b
[…|…]	或	［a|b］表示a或b
{…}	重复	{a}　表示a重复0或多次
$\{\ldots\}_m^n$	重复	$\{a\}_3^8$ 表示a重复3到8次
(…)	可选	(a)　表示a重复0或1次
"…"	基本数据元素	"a"　表a是基本数据

数据流条目的描述内容

- 名称：数据流名(可以是中文名或英文名)
- 别名：名称的另一个名字
- 简述：对数据流的简单说明
- 数据流组成：描述数据流由哪些数据项组成
- 数据流来源：描述数据流从哪个加工或源流出
- 数据流去向：描述数据流流入哪个加工或汇点
- 数据量：系统中该数据流的总量
  - 如考务处理系统中“报名单”的总量是100000张
  - 或者单位时间处理的数据流数量，如80000张/天
- 峰值：某时段处理的最大数量
  - 如每天上午9：00至11：00处理60000张表单
- 注解：对该数据流的其它补充说明

31

数据流组成

- 数据流组成是数据流条目的核心，它列出组成该数据流的各数据项，例如：
  - 培训报名单＝姓名＋单位＋课程
  - 运动员报名单＝队名＋姓名＋性别＋｛参赛项目｝
- 当一个数据流的组成比较复杂时，可以将其分解成几个数据流，例如：
  - 课程＝课程名＋任课教师＋教材＋时间地点
  - 时间地点＝｛星期几＋第几节＋教室｝

数据字典——（3）构成

- 数据流组成示例——发票

单位名称

商品名	数量	单价	金额
总金额			

日期　　营业员

发票＝单位名称＋｛商品名＋数量＋单价＋金额｝$^{5}_{1}$
＋总金额＋日期＋营业员

数据字典——（3）构成

- 数据流组成示例——发票
  - 发票＝单位名称＋｛商品名＋数量＋单价＋金额｝$_1^5$ ＋总金额＋日期＋营业员
  - 单位名称={汉字}$_1^{10}$ 或1{汉字}10
  - 商品名={汉字}$_1^8$ 或1{汉字}8
  - 数量=1..999
  - 单价=0.00..10000.00
  - 金额=0.00..100000.00
  - 日期=年＋月+日

加工条目的描述内容

- 名称：加工名
- 别名：同数据流条目
- 加工号: 加工在**DFD**中的编号
- 简述：对加工的功能的简要说明
- 输入数据流：描述加工的输入数据流，包括读哪些文件名
- 输出数据流：描述加工的输出数据流，包括写哪些文件名
- 加工逻辑：简要描述加工逻辑，或者对加工规约的索引
- 异常处理：描述加工处理过程中可能出现的异常情况，及其处理方式
- 加工激发条件：描述执行加工的条件，如，“身份认证正确”，“收到报名单”
- 执行频率：描述加工的执行频率，如，每月执行一次，每天**0**点执行
- 注解：对加工的其它补充说明

加工逻辑的描述方法

- 结构化语言（**PDL：Program Design Language**）：介于自然语言和形式语言之间的一种半形式语言
- 判定表 **(Decision Table)**：适用于加工逻辑包含多个条件，而不同的条件组合需做不同的动作
- 判定树**(Decision Tree)**：判定表的变种，它本质上与判定表是相同的，只是表示形式不同

结构化语言（PDL）

- 没有严格的语法
- 加工规约分为若干个段落，每个段落可分为内外两层：
  - 外层有严格的语法来描述它的控制结构
    - 如结构化英语中可使用**if_then_else**、**while_do**、**repeat_until**、**for_do**、**case**等结构
  - 内层可以用自然语言来描述
- 允许使用嵌套结构
- 尽可能精确、无二义、简明扼要、易理解

“审批发货单”的结构化英语描述

IF 发货单金额超过$500 THEN
    IF 欠款超过了60天 THEN
        在偿还欠款前不予批准
    ELSE （欠款未超期）
        发批准书，发货单
    ENDIF
ELSE （发货单金额未超过$500）
    IF 欠款超过60天 THEN
        发批准书，发货单及赊欠报告
    ELSE （欠款未超期）
        发批准书，发货单
    ENDIF
ENDIF

判定表（Decision Table）

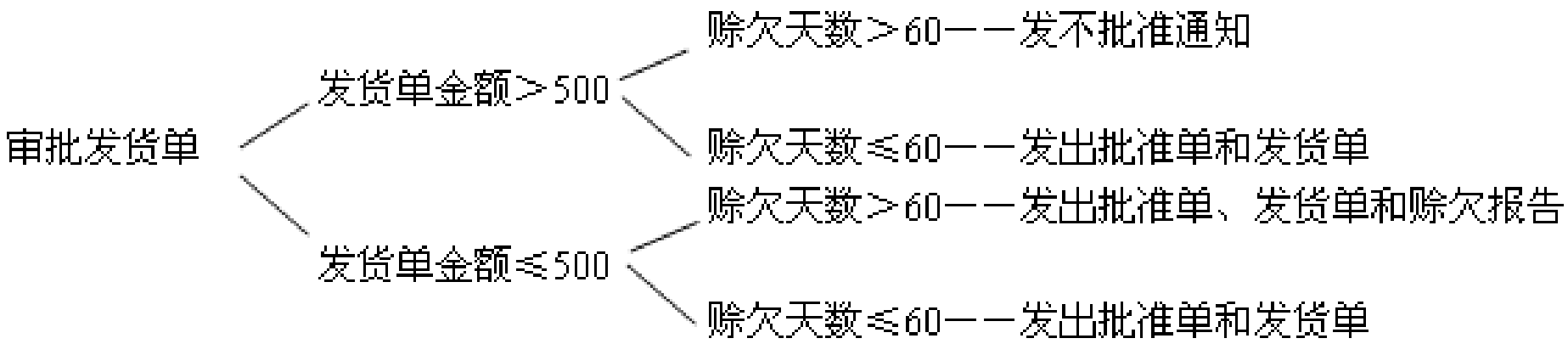
- 如果数据流图的加工需要依赖于多个逻辑条件的取值，使用判定表来描述比较合适。
- 一张判定表通常由四部分构成：
  - 所有条件：左上部
  - 所有可能操作：左下部
  - 各种条件组合：右上部
  - 各种组合条件应有操作：右下部

“审批发货单”的判定表描述

		1	2	3	4
条件	发货单金额	＞$500	＞$500	≤$500	≤$500
	赊欠情况	＞60天	≤60天	＞60天	≤60天
操作	不发出批准书	√			
	发出批准书		√	√	√
	发出发货单		√	√	√
	发出赊欠报告			√	

判定树

- 本质上与判定表是相同的，只是表示形式不同
- 例如“审批发货单”加工逻辑的判定树描述入下：



决策表/判定表（Decision Table）

- Example
  - Regular customer
    - No discount
  - Silver customer
    - 8% off
  - Gold customer
    - 15% off
  - Special discount
    - X% off

其决策表为：

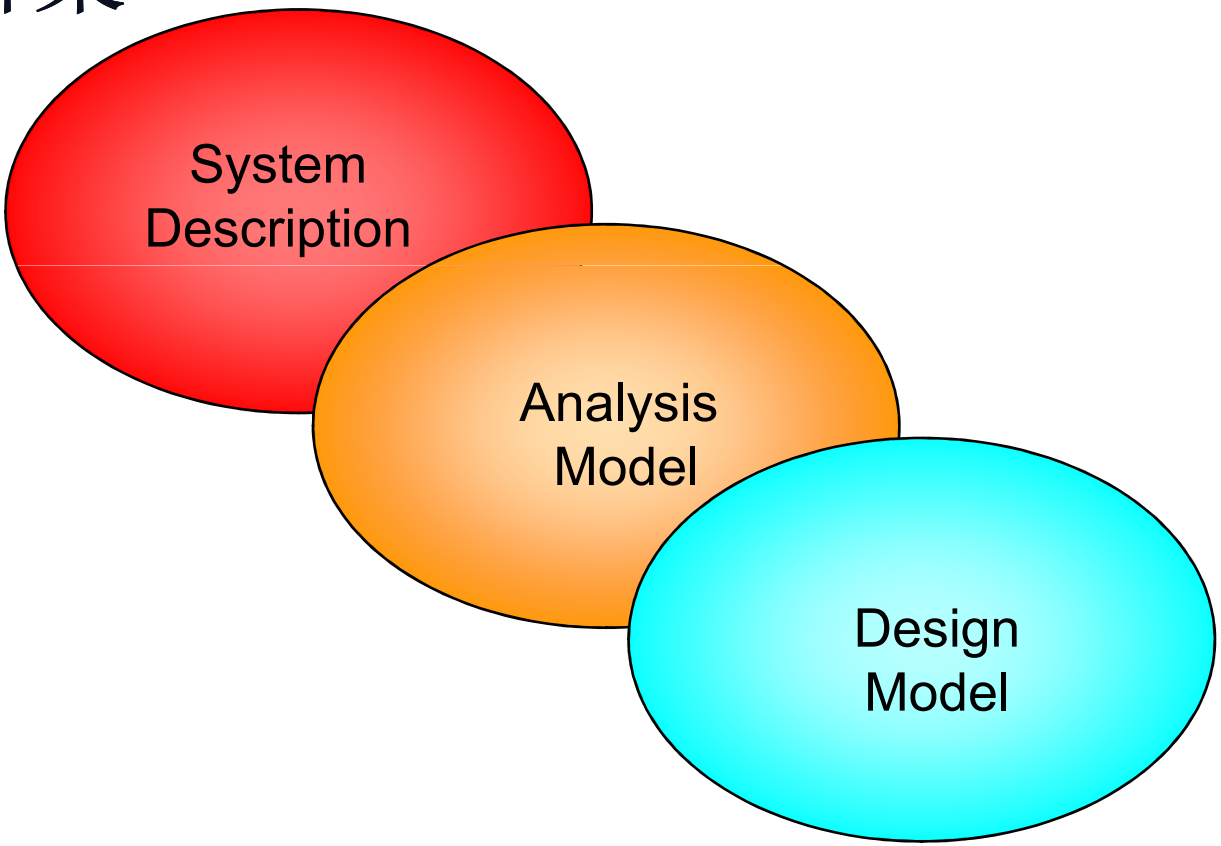
	Rules					
Conditions	**1**	**2**	**3**	**4**	**5**	**6**
regular customer	T	T				
silver customer			T	T		
gold customer					T	T
special discount	F	T	F	T	F	T
Rules						
no discount	✓					
apply 8 percent discount			✓	✓		
apply 15 percent discount					✓	✓
apply additional x percent discount		✓		✓		✓

数据字典——（4）实现 Back

- 可采用类似电子表格的工具来建立数据字典的电子文档，其好处是便于字典条目的检索，字典的管理和维护
- 也可由辅助绘制**DFD**的工具自动产生，有利于利用数据字典来检查**DFD**的一致性和完整性，并保持数据字典与**DFD**的一致
- 也可人工制作。

（二）面向过程的需求工程方法

- 需求分析的模型构成了客户和开发人员之间的桥梁



（二）面向过程的需求工程方法

- 1.需求获取
  - 起始过程（Inception）：与客户建立初步交流
  - 导出过程（Elicitation）：通过访谈与调查（会议，问卷，原型）获得需求的描述
- 2.需求分析
  - 精化过程（Elaboration）：通过分析建模，建立精确的技术模型，说明软件的功能、特征和约束。
- 3.需求处理
  - 协商过程（Negotiation）：
  - 形成规格说明（Specification）：
- 4.需求确认

（二）面向过程的需求工程方法

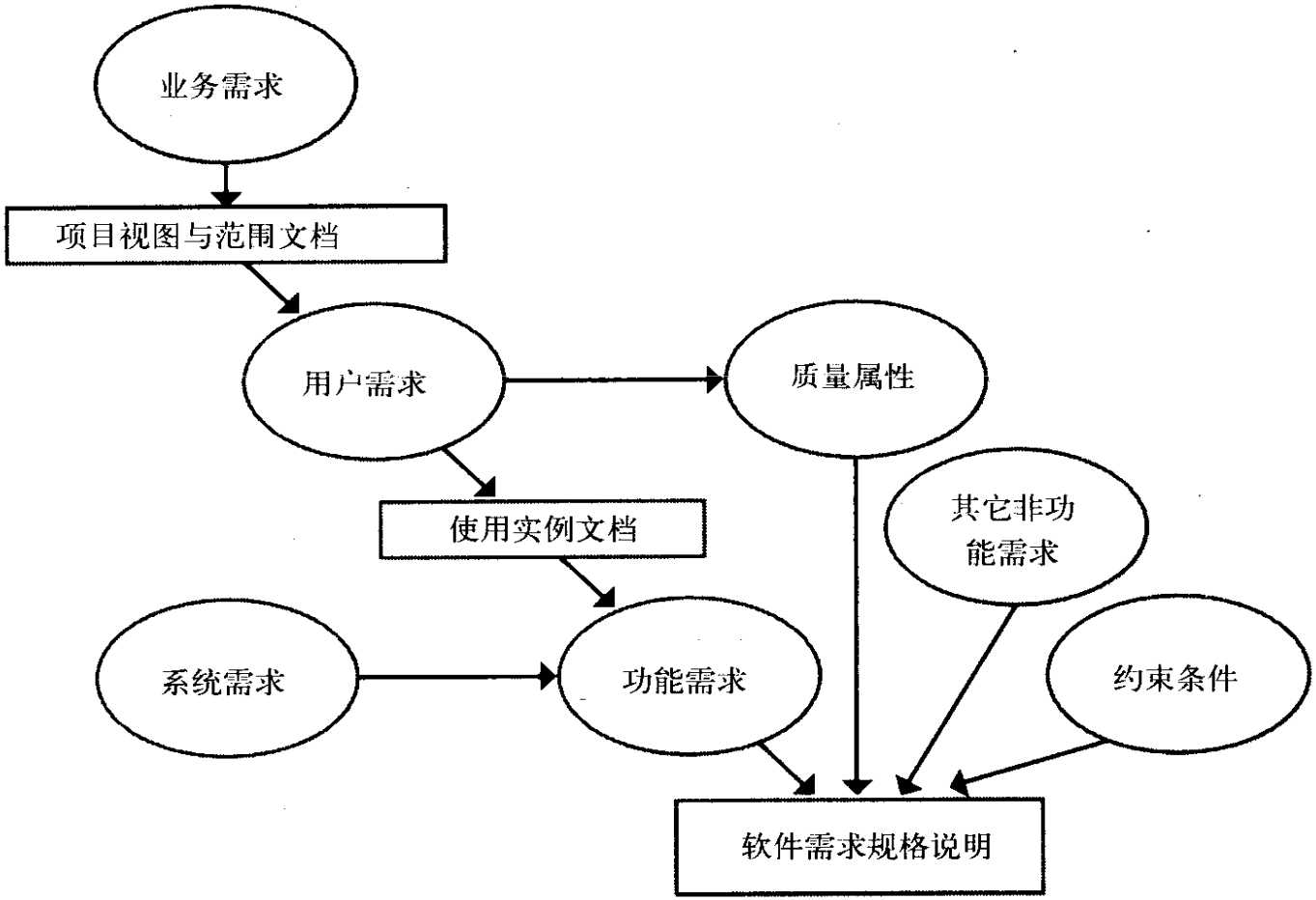
3.需求处理

Negotiating Requirements

- **Identify the key stakeholders**
  - **These are the people who will be involved in the negotiation**
- **Determine each of the stakeholders "win conditions"**
  - **Win conditions are not always obvious**
- **Negotiate**
  - **Work toward a set of requirements that lead to "win-win"**

（二）面向过程的需求工程方法

3.需求处理——编制软件需求规格说明



软件需求各组成部分之间的关系

需求规约 （示例）

- 引言：陈述软件目标，在基于计算机的系统语境内进行描述。
- 信息描述：给出软件必须解决问题的详细描述，记录信息内容和关系、流和结构。
- 功能描述：描述解决问题所需的每个功能。其中包括，为每个功能说明一个处理过程；叙述设计约束；叙述性能特征；用一个或多个图形来形象地表示软件的整体结构和软件功能与其他系统元素间的相互影响。
- 行为描述：描述作为外部事件和内部产生的控制特征的软件操作。
- 检验标准：描述检验系统成功的标志。即对系统进行什么样的测试，得到什么样的结果，就表示系统已经成功实现了。它是“确认测试”的基础。
- 参考书目：包含了对所有和该软件相关的文档的引用，其中包括其他的软件工程文档、技术参考文献、厂商文献以及标准。
- 附录：包含了规约的补充信息，表格数据、算法的详细描述、图表以及其他材料。

（二）面向过程的需求工程方法

- 1.需求获取
  - 起始过程（Inception）：与客户建立初步交流
  - 导出过程（Elicitation）：通过访谈与调查（会议，问卷，原型）获得需求的描述
- 2.需求分析
  - 精化过程（Elaboration）：通过分析建模，建立精确的技术模型，说明软件的功能、特征和约束。
- 3.需求处理
  - 协商过程（Negotiation）：
  - 形成规格说明（Specification）：
- 4.需求确认

需求确认（需求验证）

- 需求验证目的是要检验需求是否能够反映用户的意愿
- 评审人员评审时往往需要检查以下内容：
  1. 系统定义的目标是否与用户的要求一致；
  2. 系统需求分析阶段提供的文档资料是否齐全；文档中的描述是否完整、清晰、准确地反映了用户要求；
  3. 被开发项目的数据流与数据结构是否确定且充足；
  4. 主要功能是否已包括在规定的软件范围之内，是否都已充分说明；
  5. 设计的约束条件或限制条件是否符合实际；
  6. 开发的技术风险是什么；
  7. 是否详细制定了检验标准，它们能否对系统定义是否成功进行确认。

（二）面向过程的需求工程方法

- 1.需求获取
  - 起始过程（Inception）：与客户建立初步交流
  - 导出过程（Elicitation）：通过访谈与调查（会议，问卷，原型）获得需求的描述
- 2.需求分析
  - 精化过程（Elaboration）：通过分析建模，建立精确的技术模型，说明软件的功能、特征和约束。
- 3.需求处理
  - 协商过程（Negotiation）：
  - 形成规格说明（Specification）：
- 4.需求确认

（三）面向对象的需求工程方法

- 面向对象的需求工程中，分析和建模过程发生了变化。
- 从面向对象的角度出发，建立的模型有：
  - 功能模型
  - 对象模型
  - 动态模型
- 建模方法：**UML**

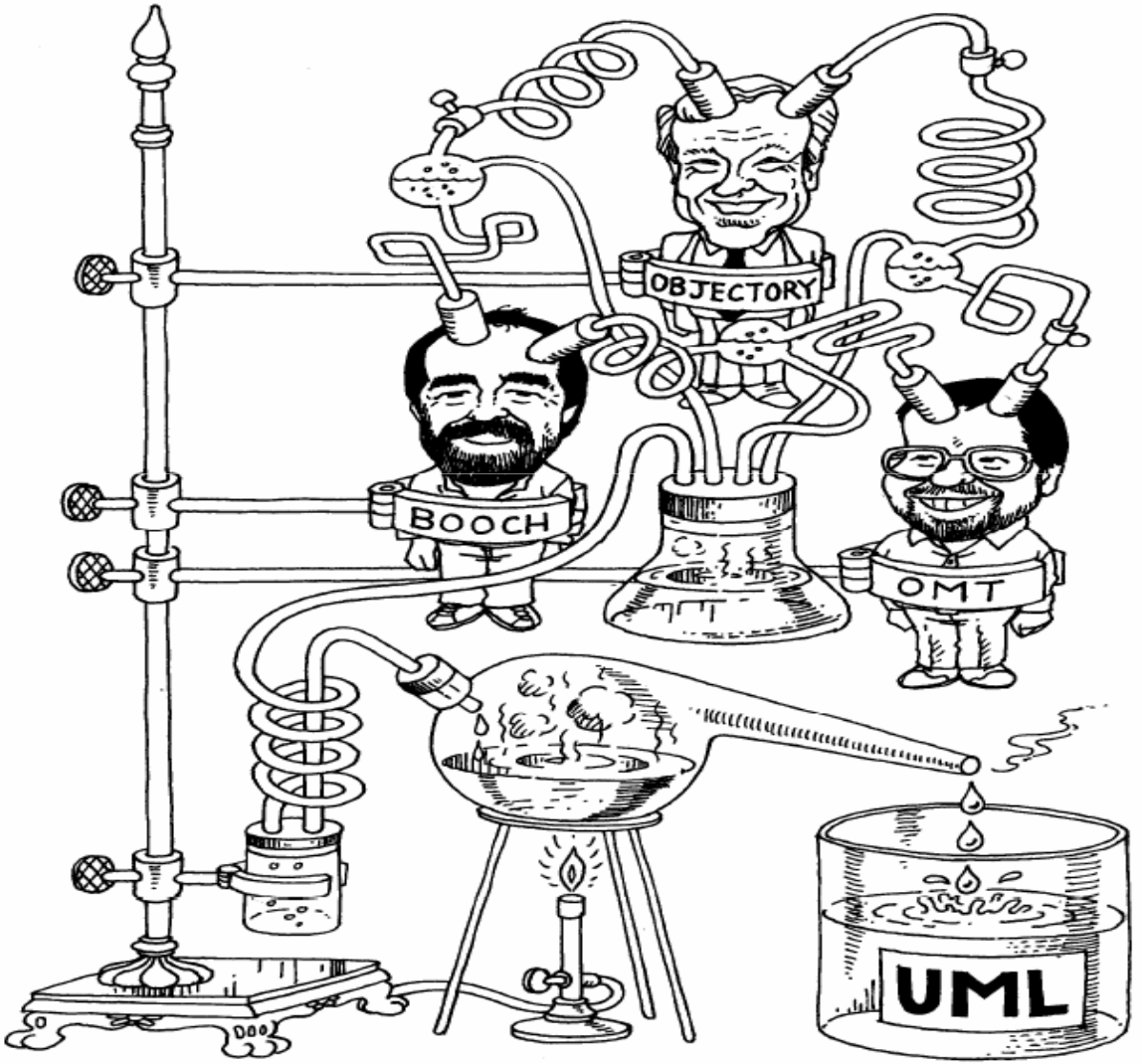
（三）面向对象的需求工程方法

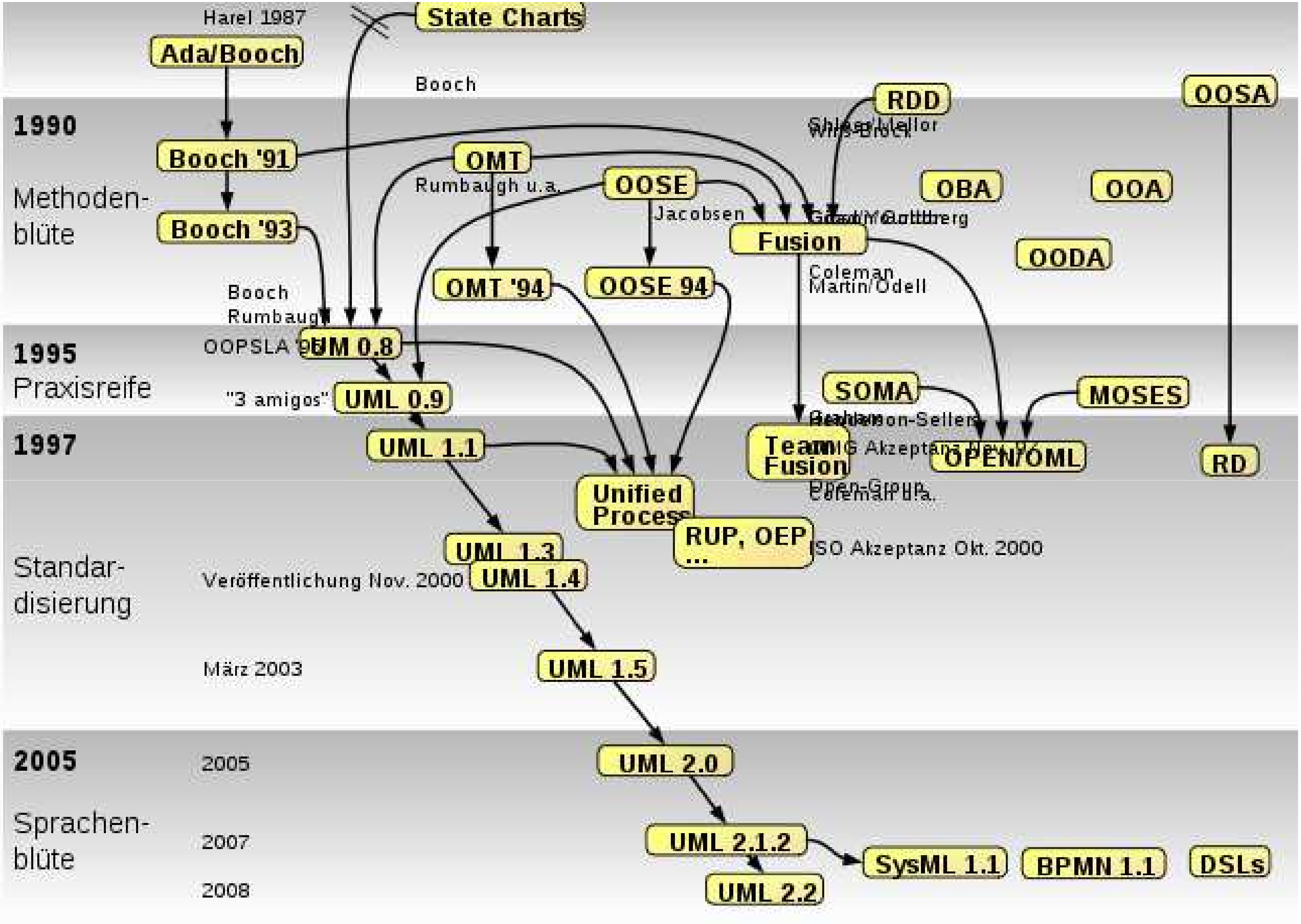
1. UML简介

- **Unified Modeling Language（统一建模语言）**是国际对象管理组织**OMG**制定的一个通用的、可视化建模语言标准
- 用于描述（**specify**）、可视化（**visualize**）、构造（**construct**）和记载（**document**）软件密集型系统的各种工件
- UML提供了一系列建模元素、概念、关系以及规则，应用于软件开发活动

Unified Modeling Language

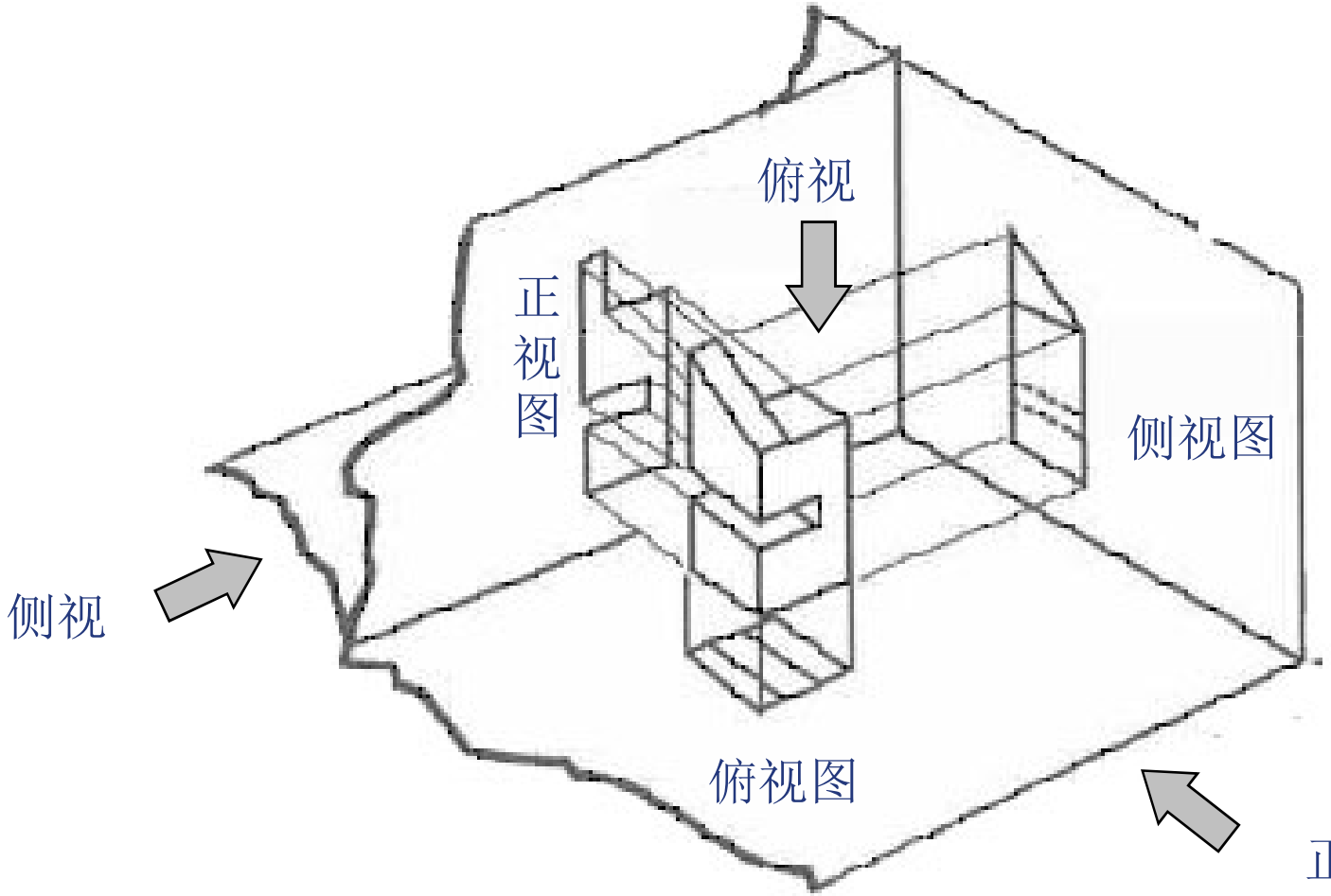
The UML is an international industry standard graphical notation for describing software analysis and designs.





Harel 1987
State Charts
Ada/Booch
Booch
1990
Booch '91
OMT
OOSE
RDD
OOSA
Methoden-
blüte
Booch '93
Rumbaugh u.a.
Jacobsen
OBA
OOA
Fusion
OODA
Booch
Rumbaugh
OMT '94
OOSE 94
Coleman
Martin/Odell
1995
Praxisreife
OOPSLA '95
UM 0.8
"3 amigos"
UML 0.9
SOMA
MOSES
1997
UML 1.1
Team
Fusion
OMG Akzeptanz Nov. 97
OPEN/OML
RD
Unified
Process
RUP, OEP
...
ISO Akzeptanz Okt. 2000
UML 1.3
Standar-
disierung
Veröffentlichung Nov. 2000
UML 1.4
März 2003
UML 1.5
2005
2005
UML 2.0
Sprachen-
blüte
2007
UML 2.1.2
SysML 1.1
BPMN 1.1
DSLs
2008
UML 2.2

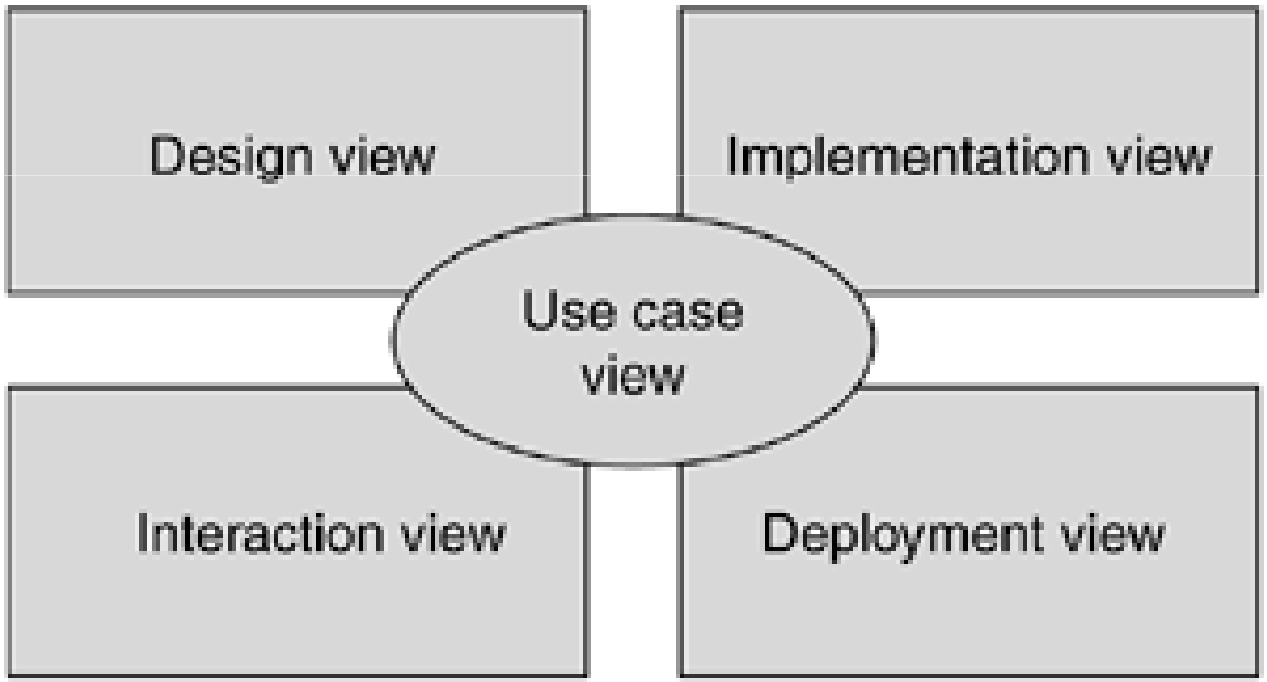
三维建筑模型的视图



（三）面向对象的需求工程方法

1. UML简介

- UML视图



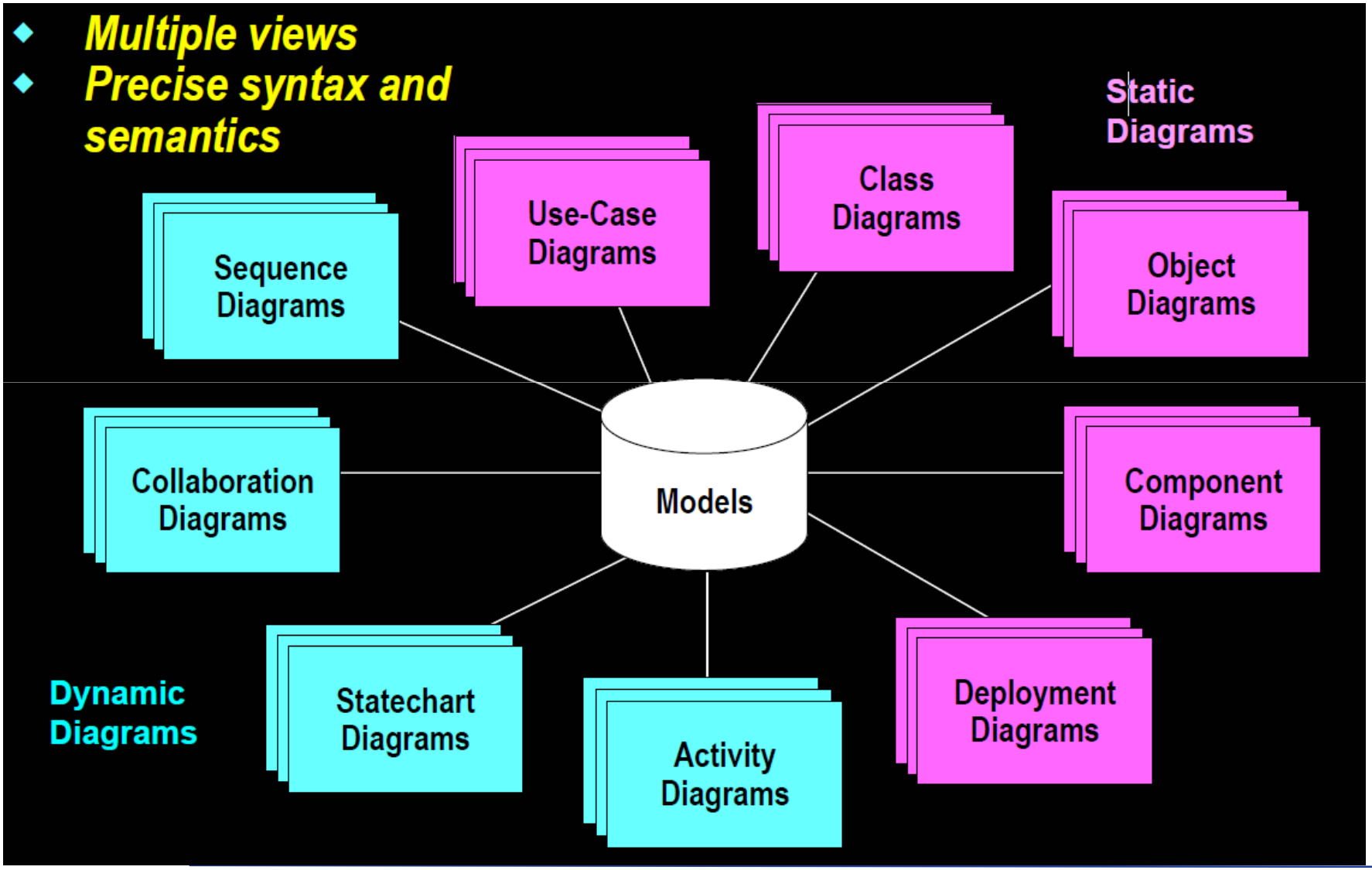
UML中的视图

- 用例视图
  - 从用户的角度描述系统所应具有的功能
- 设计视图（逻辑视图）
  - 用系统的静态结构和动态行为来展示系统内部的功能是如何实现的
  - 系统的静态结构通过类图和对象图，而动态行为使用交互图和活动图进行描述。
- 实现视图
  - 展示代码的组织和执行，描述系统的主要功能模块和个模块之间的关系，主要被开发人员使用。

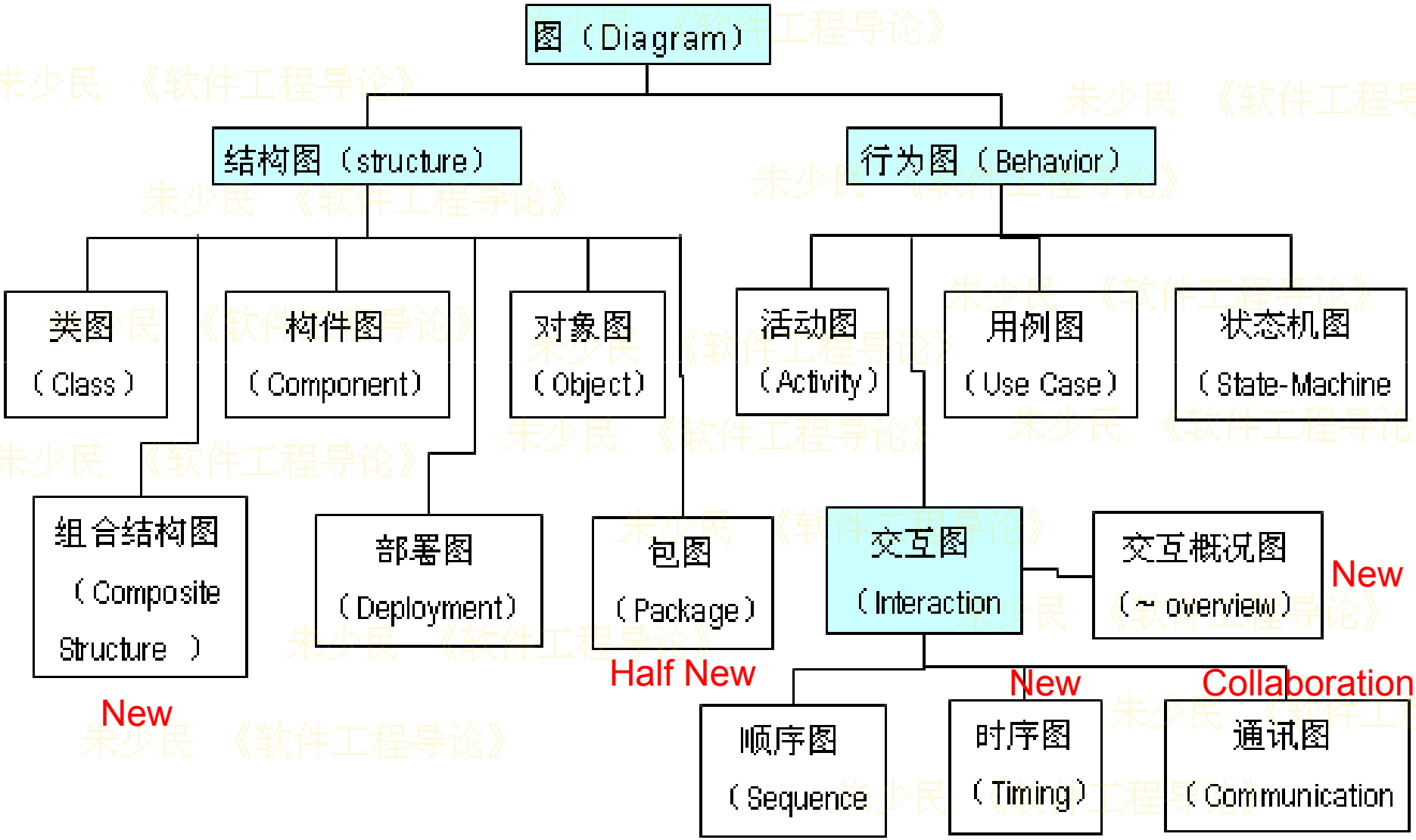
UML中的视图

- 交互视图（进程视图，并发视图）
  - 分析和设计系统如何有效利用资源、并行执行、处理来自外界的异步事件，处理线程的通信和同步。
  - 进程视图包括动态图（状态机、交互图、活动图）和实现图（交互图和部署图）
- 部署视图
  - 显示系统的具体部署

UML 1.X的9种图


Multiple views
Precise syntax and semantics
Static Diagrams
Use-Case Diagrams
Class Diagrams
Sequence Diagrams
Object Diagrams
Collaboration Diagrams
Models
Component Diagrams
Dynamic Diagrams
Statechart Diagrams
Activity Diagrams
Deployment Diagrams

UML 2.0的13种图


图（Diagram）
结构图（structure）
行为图（Behavior）
类图
（Class）
构件图
（Component）
对象图
（Object）
活动图
（Activity）
用例图
（Use Case）
状态机图
（State-Machine
组合结构图
（Composite
Structure ）
New
部署图
（Deployment）
包图
（Package）
Half New
交互图
（Interaction
交互概况图
（~ overview）
New
顺序图
（Sequence
New
时序图
（Timing）
Collaboration
通讯图
（Communication

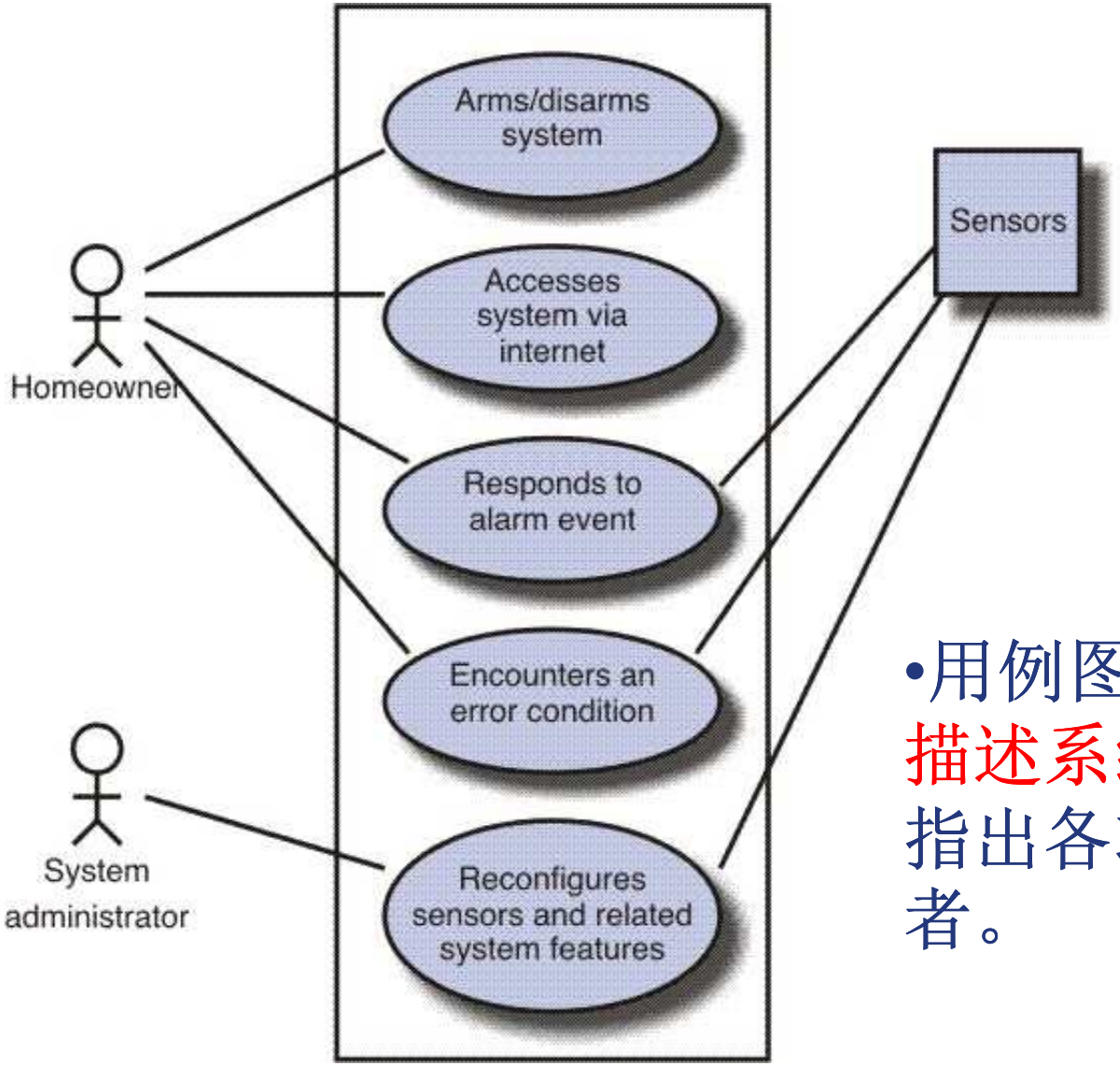
（三）面向对象的需求工程方法

- 1.需求获取
  - 起始过程（Inception）和导出过程（Elicitation）
  - 可通过用例图、活动图获得对场景的描述
- 2.需求分析
  - 精化过程（Elaboration）
  - 通过类图、包图、对象图对实体建模
  - 通过状态图、顺序图、协作图、活动图等对行为建模
- 3.需求处理
  - 协商过程（Negotiation）
  - 形成规格说明（Specification）
- 4.需求确认

（三）面向对象的需求工程方法

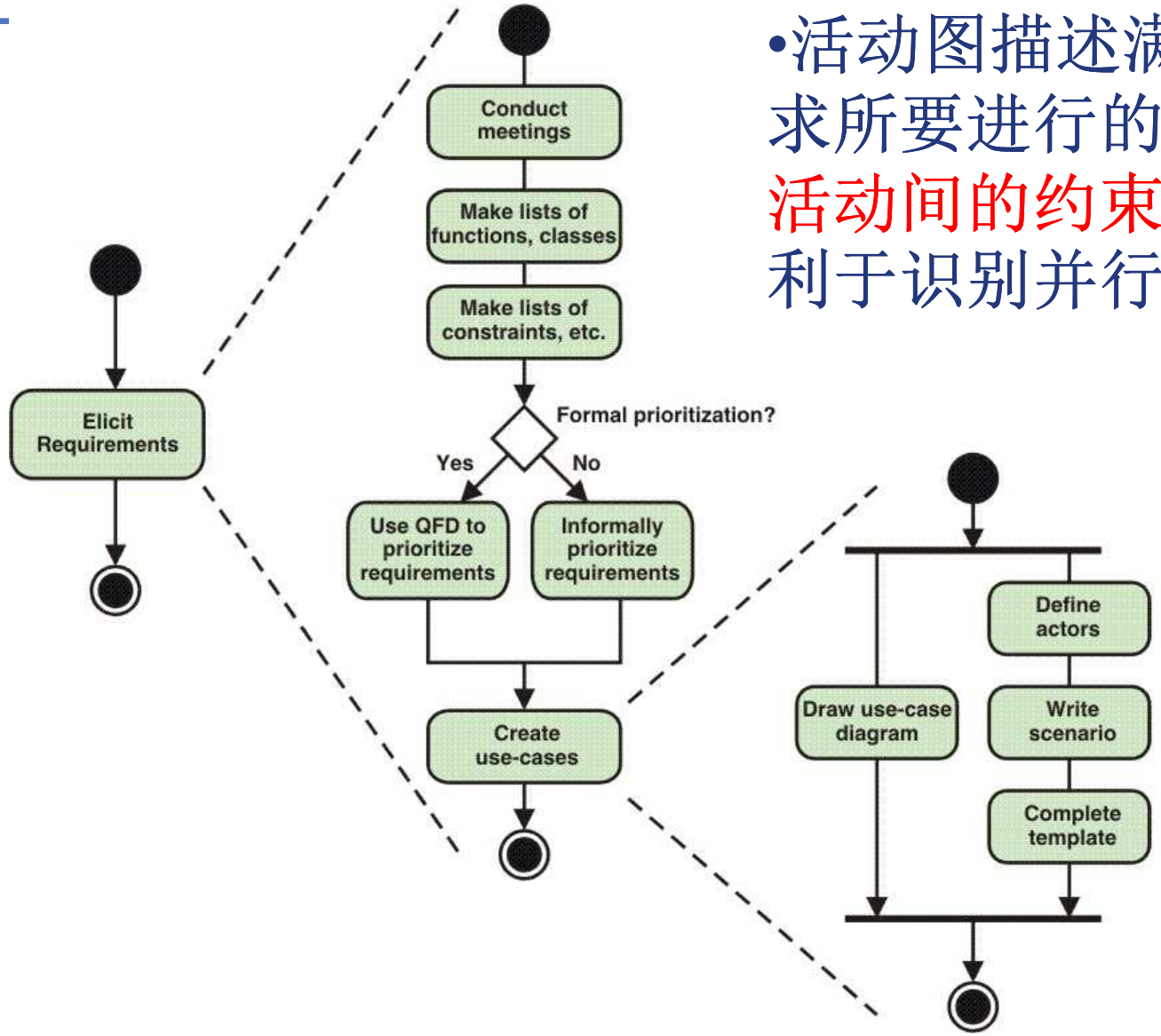
- 1.需求获取
  - 起始过程（Inception）和导出过程（Elicitation）
  - 1）识别stakeholders
  - 2）采用用例（use-case）驱动的分析方法，识别用例，画出用例图
  - 3）对用例进行详细解析，对其添加事件流描述，可使用活动图帮助描述。

用例图



- 用例图从用户角度描述系统功能，并指出各功能的操作者。

活动图



- 活动图描述满足用例要求所要进行的活动以及活动间的约束关系，有利于识别并行活动。

（三）面向对象的需求工程方法

2.需求分析

- 精化过程（**Elaboration**）
- 1）分析需求，发现实体类以及类之间的关系，确定它们的静态结构和动态行为。
  - 通过类图、包图、对象图对实体建模（静态结构建模）
  - 通过状态图、顺序图、协作图、活动图等对其动态行为建模
- 2）若需要，建立面向对象数据库模型。

发现对象类的方法

- 1）使用名词/动词分析发现类
  - 基于需求说明，找到名词或名词短语，作为可能的候选类和属性；找到动词或动词短语，作为可能的候选操作
- 2）使用CRC卡片发现类的关联
  - 利用CRC卡片，进行brain storm，发现并完善类之间的关联关系

发现对象类的实例

- 小王是一个爱书之人，家里各类书籍已过千册，而平时又时常有朋友外借，因此需要一个个人图书管理系统。该系统应该能够将书籍的基本信息按计算机类、非计算机类分别建档，实现按书名、作者、类别、出版社等关键字的组合查询功能。在使用该系统录入新书籍时，系统会自动按规则生成书号，可以修改信息，但一经创建就不允许删除。该系统还能够对书籍的外借情况进行记录，可对外借情况列表进行打印。另外，还希望能够对书籍的购买金额、册数按特定时间周期进行统计。

小王是一个爱书之人，家里各类书籍已过千册，而平时又时常有朋友外借，因此需要一个个人图书管理系统。该系统应该能够将书籍的基本信息按计算机类、非计算机类分别建档，实现按书名、作者、类别、出版社等关键字的组合查询功能。在使用该系统录入新书籍时系统会自动按规则生成书号，可以修改信息，但一经创建就不允许删除。该系统还应该能够对书籍的外借情况进行记录，可对外借情况列表打印。另外，还希望能够对书籍的购买金额、册数按特定时间周期进行统计

筛选备选类

- “小王”、“人”、“家里”很明显是系统外的概念，无须对其建模；
- 而“个人图书管理系统”、“系统”指的就是将要开发的系统，即系统本身，也无须对其进行建模；
- 很明显“书籍”是一个很重要的类，而“书名”、“作者”、“类别”、“出版社”、“书号”则都是用来描述书籍的基本信息的，因此应该作为“书籍”类的属性处理，而“规则”是指书号的生成规则，而书号则是书籍的一个属性，因此“规则”可以作为编写“书籍”类构造函数的指南。
- “基本信息”则是书名、作者、类别等描述书籍的基本信息统称，“关键字”则是代表其中之一，因此无需对其建模；
- “功能”、“新书籍”、“信息”、“记录”都是在描述需求时使用到的一些相关词语，并不是问题域的本质，因此先可以将其淘汰掉；

筛选修选类

- “计算机类”、“非计算机类”是该系统中图书的两大分类，因此应该对其建模，并改名为“计算机类书籍”和“非计算机类书籍”，以减少歧义；
- “外借情况”则是用来表示一次借阅行为，应该成为一个候选类，多个外借情况将组成“外借情况列表”，而外借情况中一个很重要的角色是“朋友”—借阅主体。虽然到本系统中并不需要建立“朋友”的资料库，但考虑到可能会需要列出某个朋友的借阅情况，因此还是将其列为候选类。为了能够更好地表述，将“外借情况”改名为“借阅记录”，而将“外借情况列表”改名为“借阅记录列表”；
- “购买金额”、“册数”都是统计的结果，都是一个数字，因此不用将其建模，而“特定时限”则是统计的范围，也无需将其建模；不过从这里的分析中，我们可以发现，在该需求描述中隐藏着一个关键类—书籍列表，也就是执行统计的主体

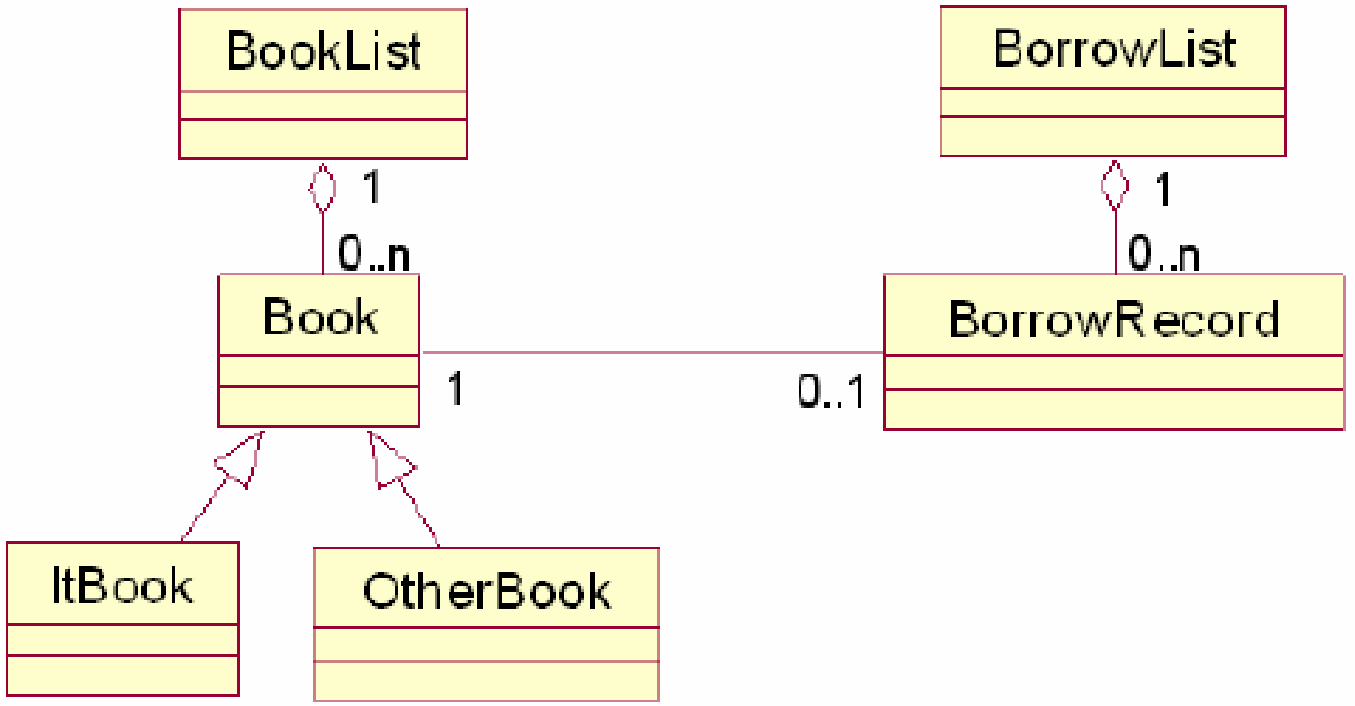
得到候选类

书籍　　计算机类书籍　　非计算机类书籍

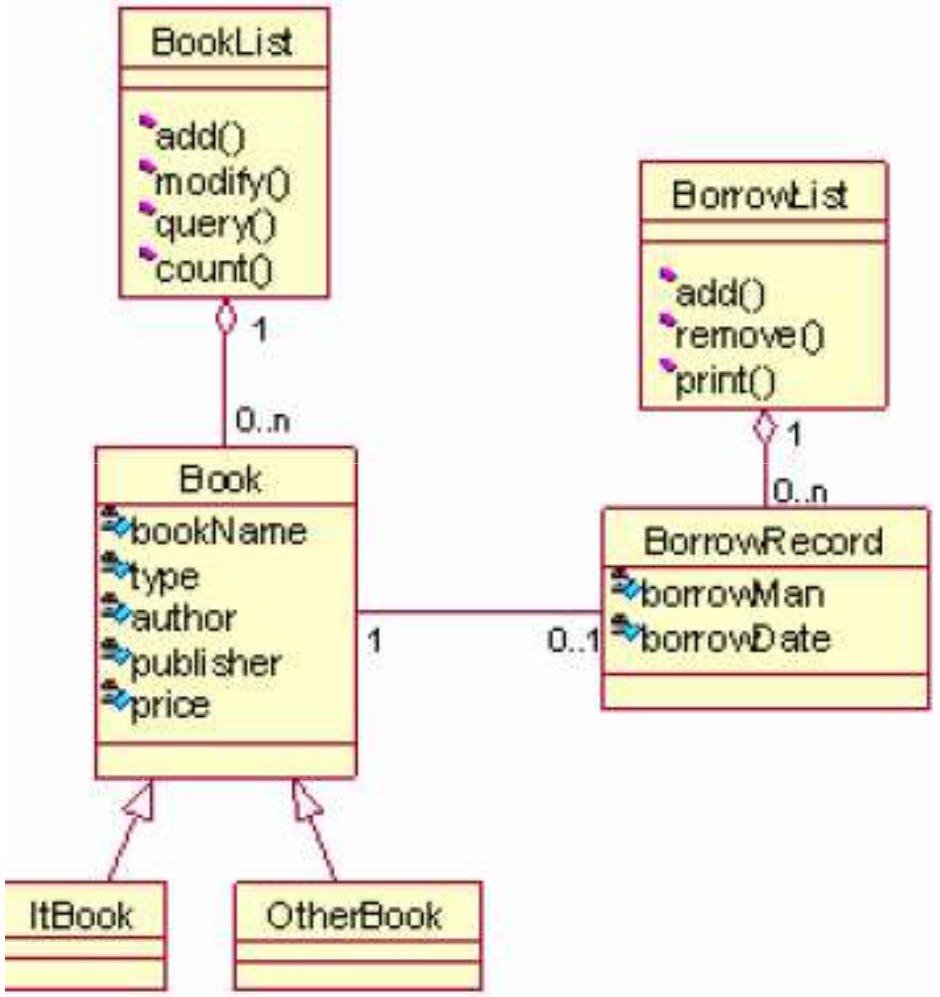
借阅记录　　借阅记录列表　　书籍列表

确定对象类

- 画出类图


BookList
BorrowList
1
0..n
1
0..n
Book
BorrowRecord
1
0..1
ItBook
OtherBook

细化对象类


BookList
add()
modify()
query()
count()
1
0..n
Book
bookName
type
author
publisher
price
1
0..1
BorrowList
add()
remove()
print()
1
0..n
BorrowRecord
borrowMan
borrowDate
ItBook
OtherBook

- 软工作业
-
- 教务处需要开发一款手机应用程序“教室在哪儿？”给全校师生使用。请根据本校具体情况进行需求分析,构建系统的类图及相关其他图。
-

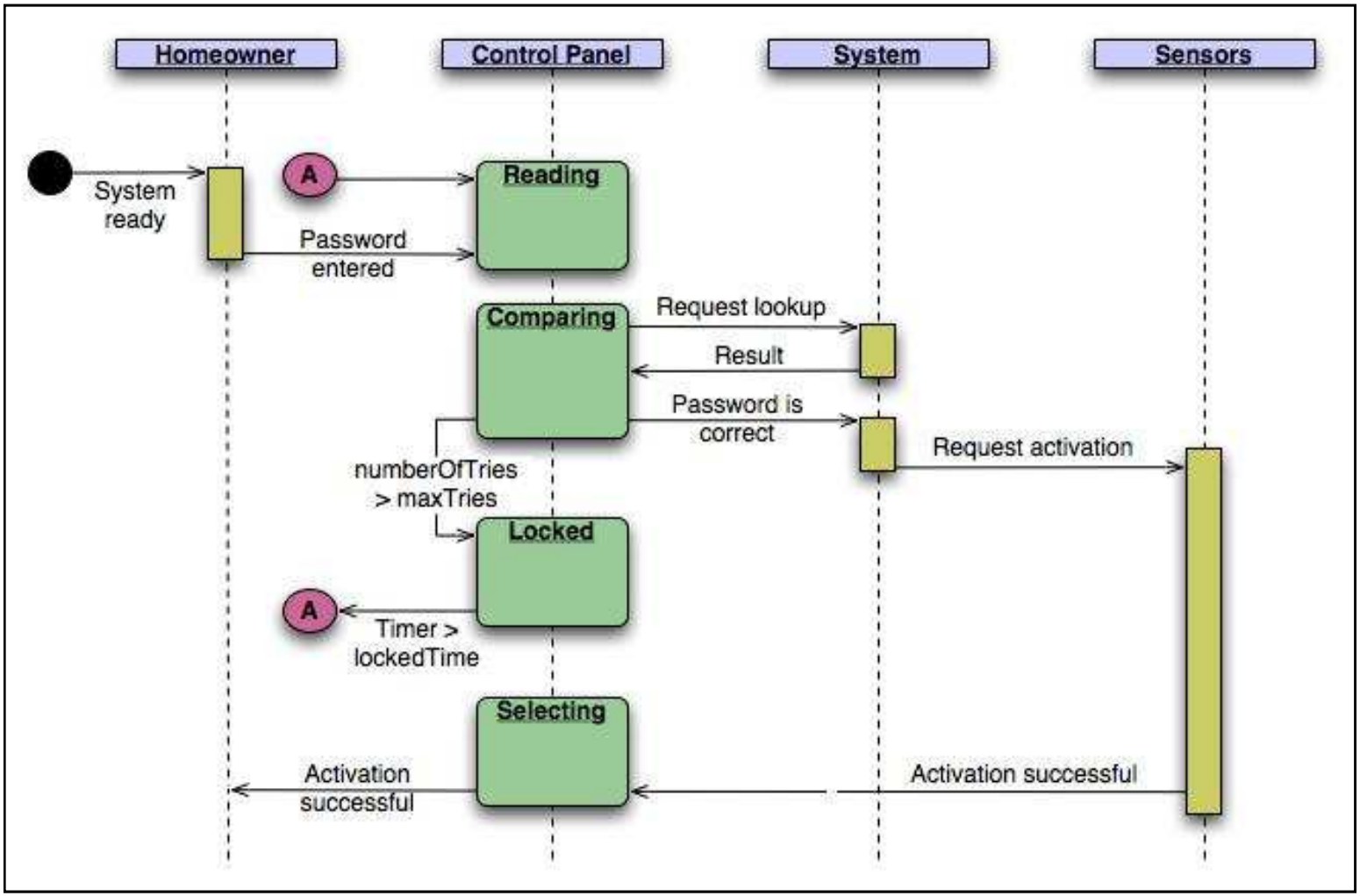
实体建模（静态结构建模）

- 类图描述系统中类的静态结构。不仅定义系统中的类，表示类之间的联系如关联、依赖、聚合等，也包括类的内部结构（类的属性和操作）。
- 对象图是类图的实例，几乎使用与类图完全相同的标识。
- 包由包或类组成，表示包与包之间的关系。包图用于描述系统的分层结构。

行为建模

- 状态图描述类的对象所有可能的状态以及事件发生时状态的转移条件。
  - 状态图是对类图的补充
  - 仅为那些有多个状态其行为受外界环境的影响并且发生改变的类画状态图
- 顺序图显示对象之间的动态合作关系，它强调对象之间消息发送的顺序，同时显示对象之间的交互
- 协作图描述对象间的协作关系，跟顺序图相似，显示对象间的动态合作关系。

顺序图


Homeowner
Control Panel
System
Sensors
System ready
A
Reading
Password entered
Comparing
Request lookup
Result
Password is correct
Request activation
numberOfTries > maxTries
Locked
A
Timer > lockedTime
Selecting
Activation successful
Activation successful

建立面向对象数据库模型

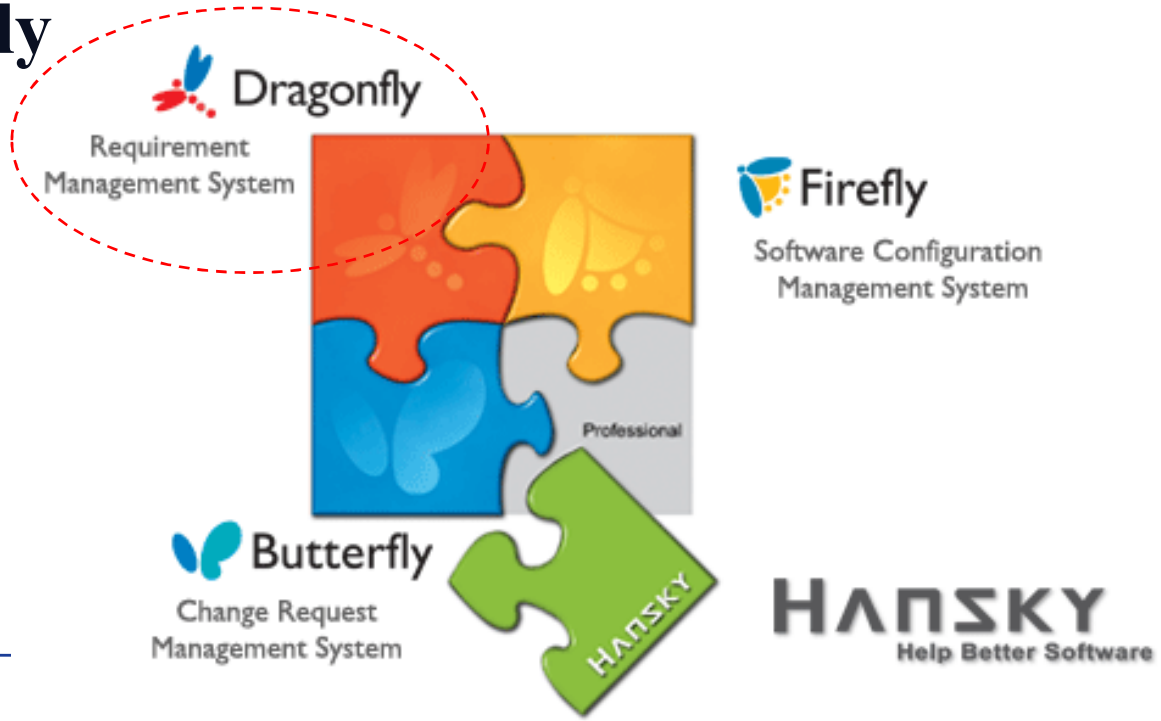
- 从某种意义上说**UML**类图是**E-R**图的超集，**ER**图只针对存储的数据，而类图在此基础上增加了行为建模的能力。
- 使用类图时，为了便于表示从数据库文件中引用和检索对象，需要定义一个代表持久性的父类，该类实现对数据库文件的读写、存储、检索、更新等操作
- 此父类用**UML**标准构造型**{persistent}**来表示

（三）面向对象的需求工程方法

- 1.需求获取
  - 起始过程（Inception）和导出过程（Elicitation）
  - 可通过用例图、活动图获得对场景的描述
- 2.需求分析
  - 精化过程（Elaboration）
  - 通过类图、包图、对象图对实体建模
  - 通过状态图、顺序图、协作图、活动图等对行为建模
- 3.需求处理
  - 协商过程（Negotiation）
  - 形成规格说明（Specification）
- 4.需求确认

（四）需求分析和管理工具

- IBM Rational RequisitePro
- Telelogic DOORS
- Borland CaliberRM
- CA BPWin
- Hansky Dragonfly



总结

- 本部分首先学习了需求工程要做什么，这么做的必要性，以及怎么去做（需求工程过程）。
- 需求开发过程通常包括：
  - **1.需求获取**
    - 起始过程（**Inception**）和导出过程（**Elicitation**）
    - 方法：访谈与调查（会议，问卷，原型），可通过用例图、活动图获得对场景的描述
  - **2.需求分析：精化过程（Elaboration）**
    - 面向过程：数据流图，实体关系图，数据字典，控制流图
    - 面向对象：通过类图、包图、对象图对实体建模，通过状态图、顺序图、协作图、活动图等对行为建模
  - **3.需求处理**
  - **4.需求确认**

Southeast University

Thank You !

lliao@seu.edu.cn