
HIT.wh 软件学院

系统分析与设计

课堂笔记

衍淞 王

2020-4-21

第 1 章 现代系统分析员涉及的领域

1.1 解决商业问题的分析员

➤ 分析员解决问题的方案

- 研究理解问题
- 核实解决问题的效益大于成本
- 设计一套可能的解决方案
- 决定一个最佳方案并进行推荐
- 详细说明所选方案的细节
- 实施方案
- 监控证实得到预期结果

1.2 解决业务问题的系统

➤ 信息系统定义

对信息进行收集、处理、存储并输出所需信息的一组相互关联的部件的集合。

➤ 部件的理解

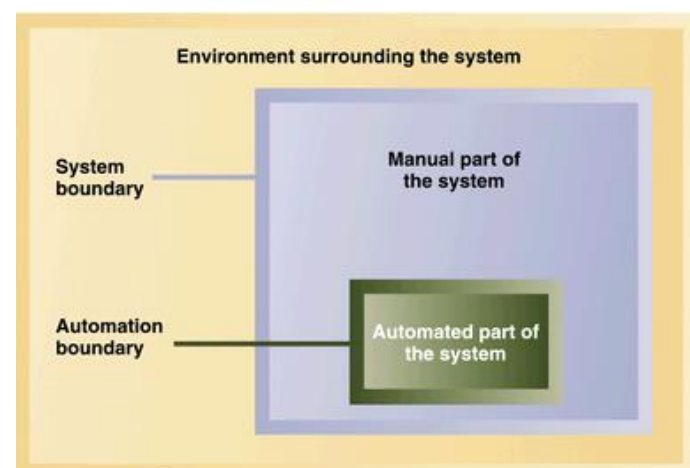
功能分解的子系统；不同类型的事件

➤ 边界

- **系统边界：**系统与外部环境之间输入输出所通过的分界
- **自动化边界：**系统内自动和手动的分界

➤ 信息系统的类型

- 事务处理系统 TPS
- 管理信息系统 MIS
- 主管信息系统 EIS
- 决策支持系统 DSS
- 通信与办公系统



1.3 系统分析员需要的技能

➤ 技术知识与技能

- 了解计算机基础知识
- 了解开发系统的多种工具和技术
- 了解系统开发的专门技术

➤ 商务知识与技能

- 对商业的全面熟悉
- 对特定行业的熟悉
- 对一个具体公司的熟悉

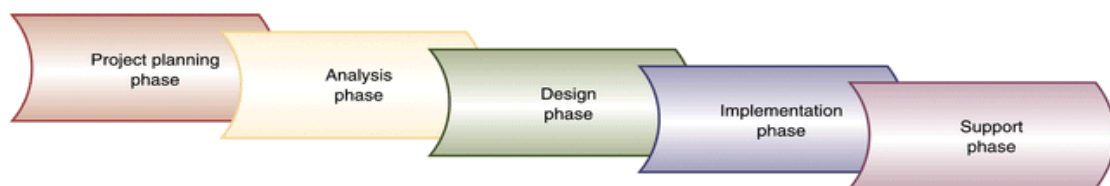
➤ 人的知识与技能

➤ 诚实与道德

第 2 章 项目经理级的分析员

2.1 系统开发生命期 – SDLC

2.1.1 系统开发生命期 – SDLC

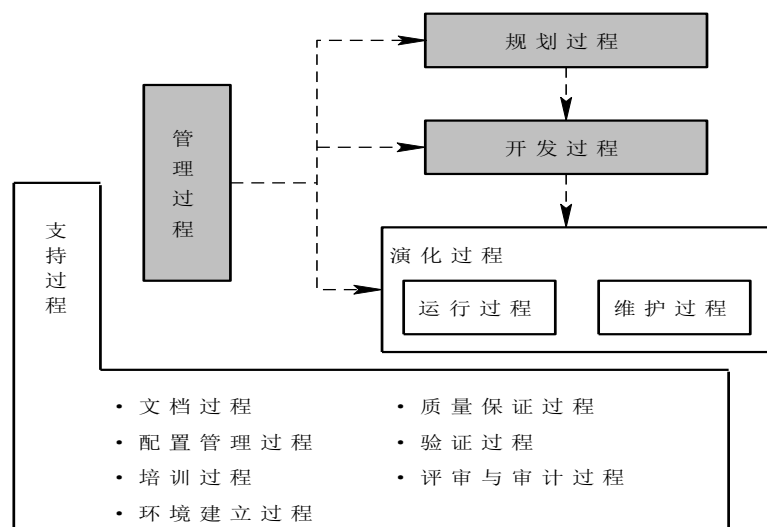


- 工程计划过程(之前没有的)
- 分析过程
- 设计过程
- 实现过程
- 支持过程

2.1.2 信息系统生存周期

➤ 定义

信息系统生存周期(Information System Life Cycle)是指从提出信息系统建设的设想开始, 历经规划, 开发, 演化等过程, 一直到被其它信息系统所替代的全过程。在信息系统生存周期中存在规划、开发、演化、管理和支持五个过程, 其中演化过程中又包括运行和维护两个子过程。



整个工程中贯穿着：

- 需求管理：系统必须完成需求要求的内容
- 成本管理：控制支出，项目必须要有利润
- 时间管理：控制项目的推进进度，在规定的的时间点完成任务

通过一些支持过程，去更好的实现这些管理

➤ 过程的具体定义

- 规划过程

规划过程(Planning Process)是信息系统生存周期中的第一个过程。在规划过程中，先提出信息系统建设的设想，对所要开发的信息系统进行规划和可行性分析，然后决定该信息系统**是否有必要开发**，并且制定**信息系统建设的总体规划**。

- 开发过程

开发过程(Development Process)是在信息系统规划的基础上，研制信息系统的全过程。信息系统开发要经过初始、细化、构建、移交等阶段，需要从事业务分析、需求分析、系统分析、系统设计、系统实现、测试等方面的工作，并经过**多次反复迭代**，最后形成可以交付用户使用的信息系统。

- 演化过程

演化过程(Evolution Process)是信息系统发挥作用的全过程。这个过程从信息系统提交使用开始，到信息系统不能继续适应企业目标、管理、技术的变化被终止为止。演化过程包括运行和维护两个子过程。**运行过程是信息系统应用于组织的业务、管理和决策，并发挥其作用的过程。维护过程则是信息系统要不断地适应环境和需求的变化，进行完善和版本更新的过程。**

- 管理过程

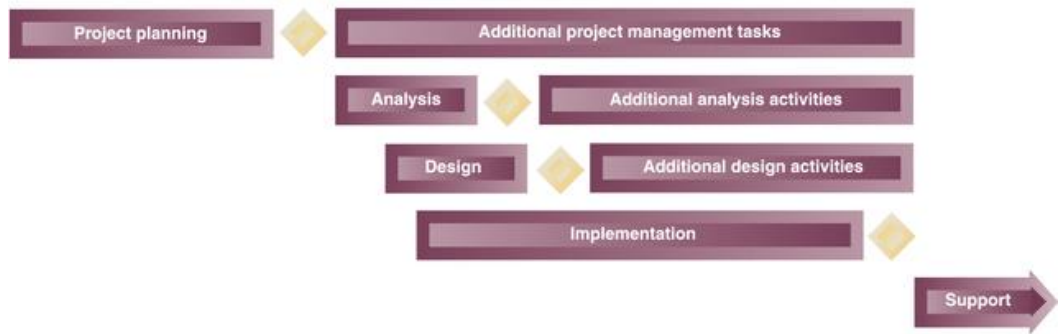
管理过程(Management Process)对信息系统实施有效的管理和控制。根据信息系统管理的内容划分，可以分为信息系统规划管理、开发管理、维护管理、运行管理等；根据信息系统管理的对象划分，可以分为信息系统人员管理、信息资源管理、项目管理、网络管理等。

- 支持过程（软件项目过程管理）

支持过程(Supporting Process)是在信息系统生存周期中，除了其它过程之外，起着辅助、支持作用的信息系统过程。支持过程包括一组过程，主要有文档过程、配置管理过程、质量保证过程、验证过程、评审和审计过程、培训过程、环境建立过程等。

➤ 系统开发活动的重叠

注意：运维不能和实现并行，计划不能和设计分析并行，必须提前搞



2.2 项目管理

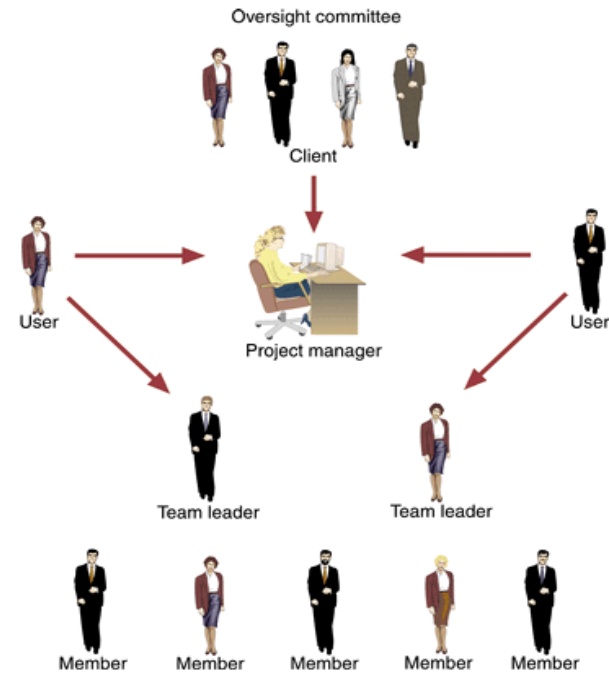
➤ 项目管理的含义（时间成本质量）

组织和指导其他人员在预先确定的进度和预算内实现计划

➤ 系统开发项目的参加人员

- 客户（提供资金弄软件）
- 监督委员会
- 用户（使用产品）
- 成员，小组领导
- 项目经理：要接受监督委员会的监督

➤ 从事项目管理工作的项目经理，必须有较好的沟通能力，协调客户、用户、程序员之间的关系



2.3 项目的启动

➤ 项目启动的三种方式

- 自顶向下

由老板提出，提出建设一个工程/系统的要求，具体的由老板提出

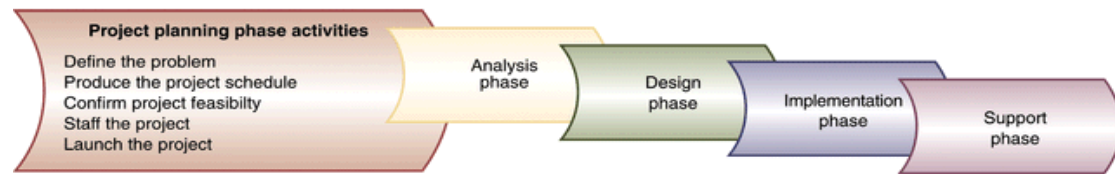
- 自下至上

根据市场的需求，提出的建设新工程的需求，比如疫情情况下的直播平台搭建

- 针对外界压力

建立系统应对外界的变化，由于政策、法律等因素的影响，必须建立新系统

2.4 项目计划阶段



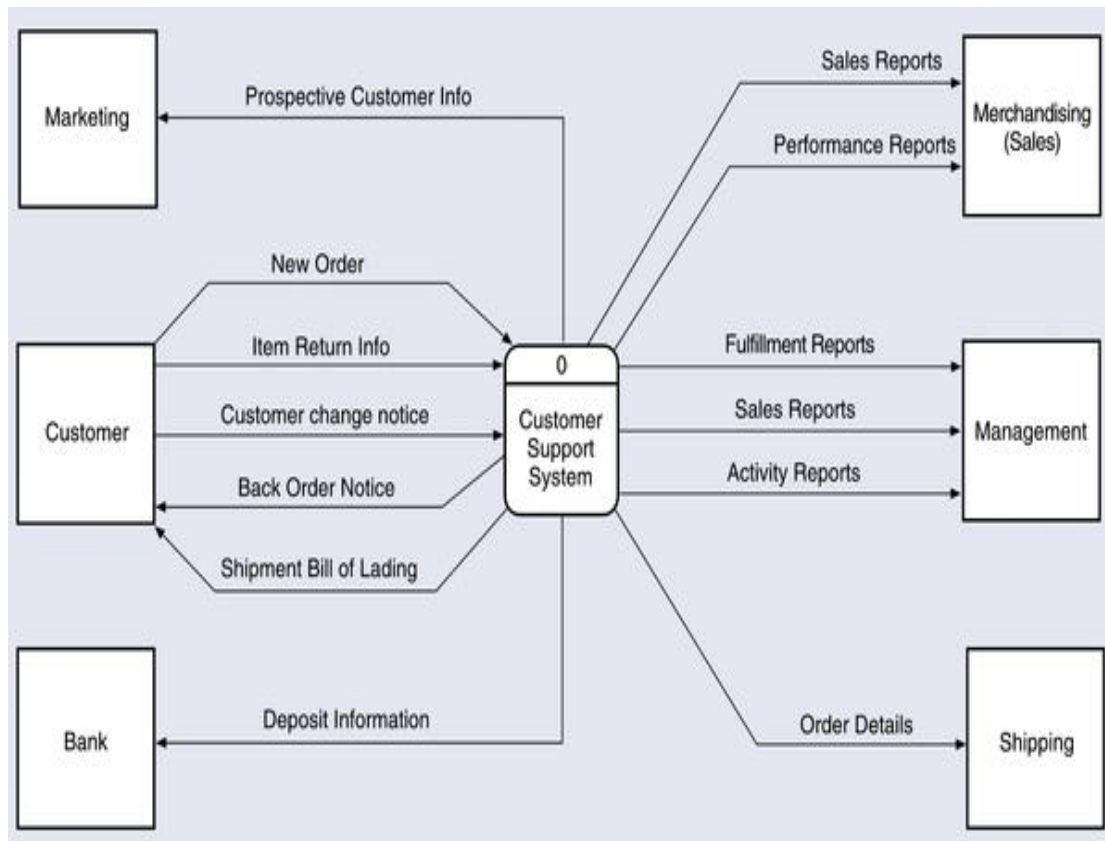
2.4.1 定义问题

➤ 定义问题的内容

- 检查项目启动的要求
- 需求说明
- 预期的商业收益
- 确定新系统的能力

➤ RMO 实例

- 启动的要求说明
 - 由公司的战略计划导致开发新的客户支持系统，即自顶向下
- 需求说明
 - 减少订单手工处理引起的错误
 - 通过快速订单处理加速订单完成
 - 通过 Internet 开辟一个新的销售渠道
 - 通过跟踪热销和滞销情况增加营业额
 - 通过扩大客户支持与信息增加客户信誉
- 确定新系统的能力
 - 支持联机的客户、订单、退还订单、退货处理
 - 支持传统的电话订购和邮购销售
 - 支持网络客户及目录产品的销售，包括网上的购买和订单跟踪
 - 维持足够的数据库和历史信息以支持市场分析
 - 为顾客提供交易的历史信息
 - 24 小时支持订单发货
 - 协调多个仓库的发货
- 联系图



2.4.2 确认项目可行性

➤ 经济可行性

• 成本/收益分析

- 成本：开发成本和运行成本
- 收益来源：减少成本和增加收入
- 财务计算：净现值、投资回收期、投资收益率

• 项目资金

➤ 组织和文化上的可行性

为特殊的客户开发项目时要注意文化差异，例如在给阿拉伯国家的客户设计 logo 不能有基督教的元素

➤ 技术可行性

➤ 进度可行性

➤ 资源可行性

2.4.3 制定项目进度表

➤ 定义

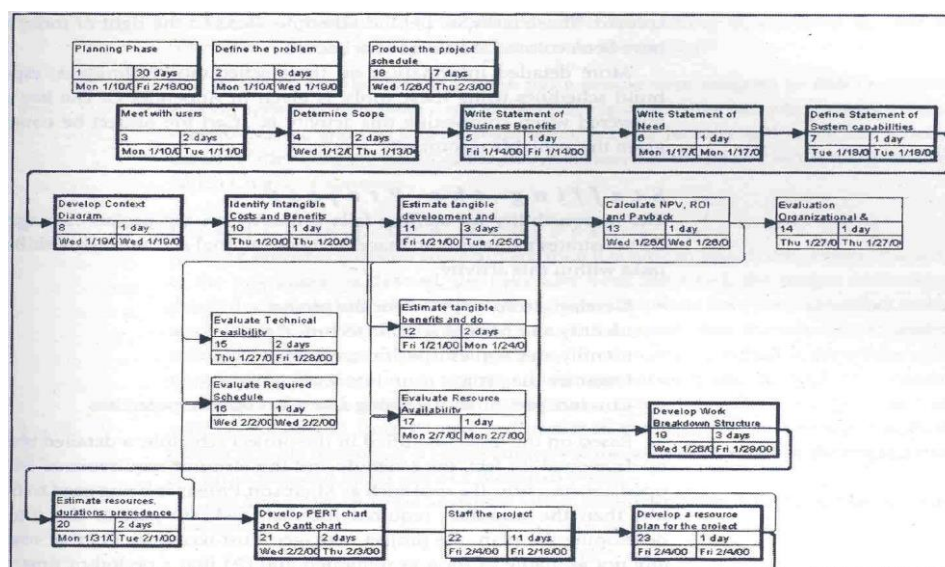
定义出开发的各个阶段，每个阶段要做什么活动，活动的任务目标是什么，整个计划越详细学好

➤ 进度表

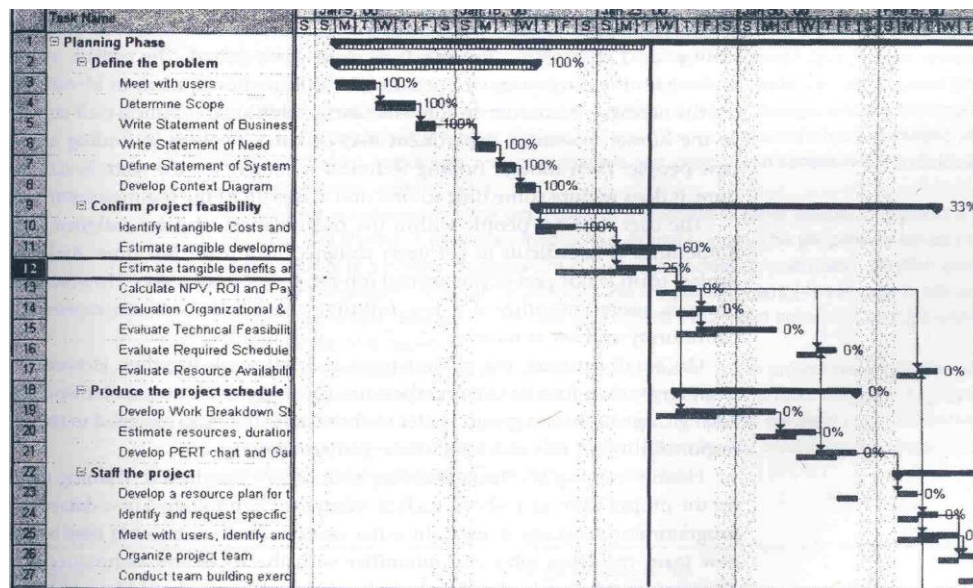
• 资源估算任务表

| | | |
|-------|-------------|---------|
| 1.0 | 项目计划阶段 | |
| 1.1 | 定义问题 | |
| 1.1.1 | 会见用户 | 1 4 2 2 |
| 1.1.2 | 确定作用域 | 1 3 2 2 |
| 1.1.3 | 编写商业收益说明 | 1 2 1 1 |
| 1.1.4 | 编写需求说明 | 1 2 1 1 |
| 1.1.5 | 定义系统能力说明 | 1 2 1 1 |
| 1.1.6 | 制定关联图 | 1 2 1 1 |
| 1.2 | 确认项目可行性 | |
| 1.2.1 | 识别无形成本和收益 | 1 1 1 2 |
| 1.2.2 | 估算有形开发和运行成本 | 1 3 2 2 |

• PERT 图



• Gantt 图



2.4.4 为项目提供人员

➤ 内容

- 制定详细的资源计划：工位、资金、服务器资源
- 确定并邀请专门技术人员
- 确定并邀请专门用户人员
- 把项目组分成多个工作小组
- 实施初步的培训和建组训练

2.4.5 启动项目

➤ 内容

- 监督委员会最后定案，资金释放
- 正常渠道的正式的通知

第 3 章 系统开发方法

3.1 方法、模型、工具和技术

➤ 方法

- 提供完成 SDLC 每一步的详细指导，包括具体的模型、工具和技术

➤ 模型

- 对现实世界某些重要方面的表示
- 系统构件的一些模型

-
- 流程图、数据流图(DFD)、实体—联系图(ERD)、结构图、用例图、类图、顺序图
 - 用于管理系统开发过程的一些模型
 - PERT 图、甘特图、组织层次图、财务分析模型-----NPV, ROI

➤ 工具

- 帮助生成项目中所需模型或其他组件的软件支持
 - 项目管理应用程序、制图/图形应用程序、集成开发环境(IDE)、数据库管理应用程序、代码生成工具
 - 计算机辅助系统工程(CASE)工具: 指的是根据设计图生成基础代码的一些工具
 - 反向工程工具: 把二进制代码反编译成汇编指令集

➤ 技术

- 帮助完成系统开发活动或任务的指导
 - 战略规划技术、项目管理技术、用户面谈技术、数据建模技术、关系型数据库设计技术、结构化分析技术、结构化设计技术、结构化编程技术、软件测试技术、面向对象分析和设计技术

3.2 系统开发的方法

3.2.1 传统方法

➤ 定义

- 通过查看过程和数据及其相互关系来定义系统需求、设计和构造系统, 把系统看作一个功能的集合。包括结构化方法和信息工程方法
- 通过不同的模块调用完成响应的功能

➤ 结构化方法

- 使用结构化编程、结构化分析和结构化设计技术。
- 结构化分析:

定义系统需要做什么、需要存储和使用哪些数据，系统需要什么样的输入和输出，以及系统的功能如何组织。

- **结构化模型：**

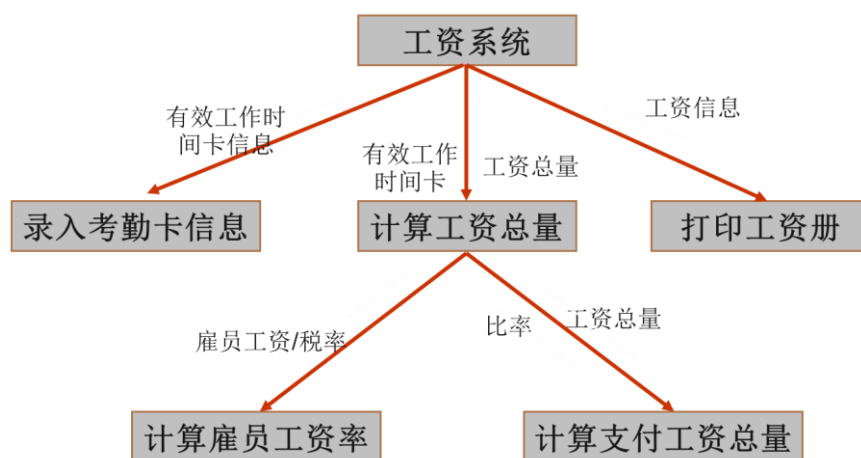
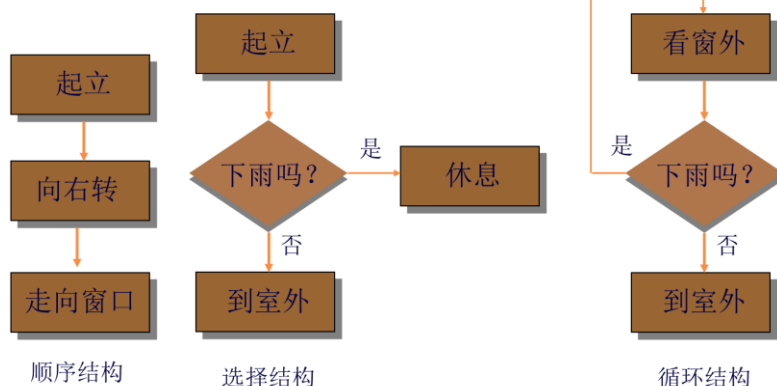
DFD, E-R 图, 结构图

- **结构化编程的三种结构**

顺序、选择、循环

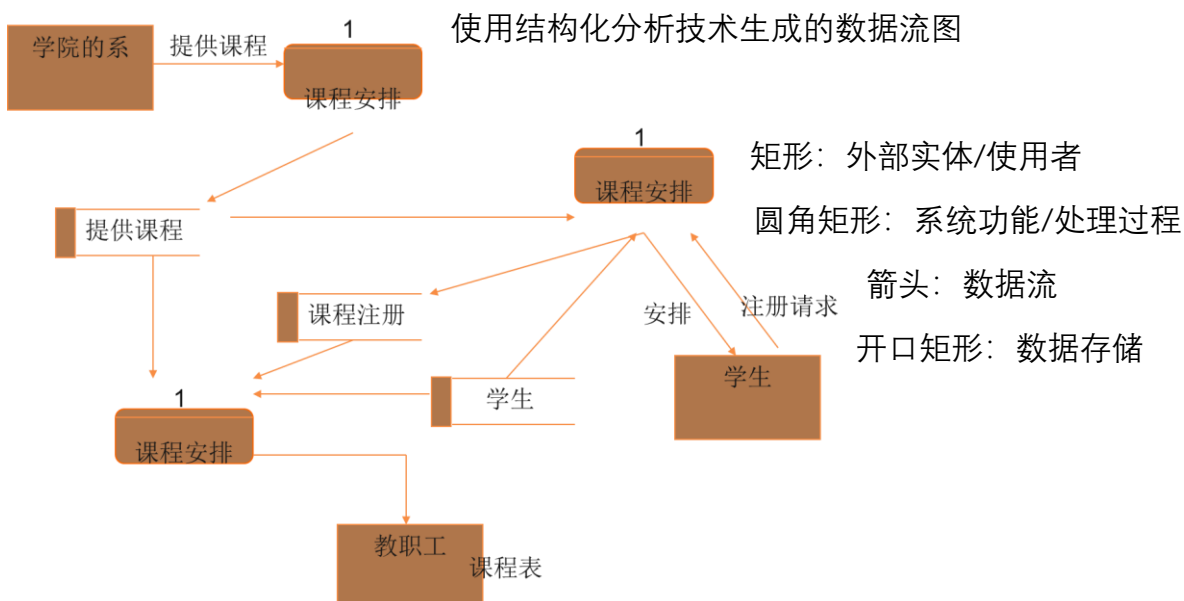
自顶向下的模块化程序设计中采用这三种结构实现具体功能模块

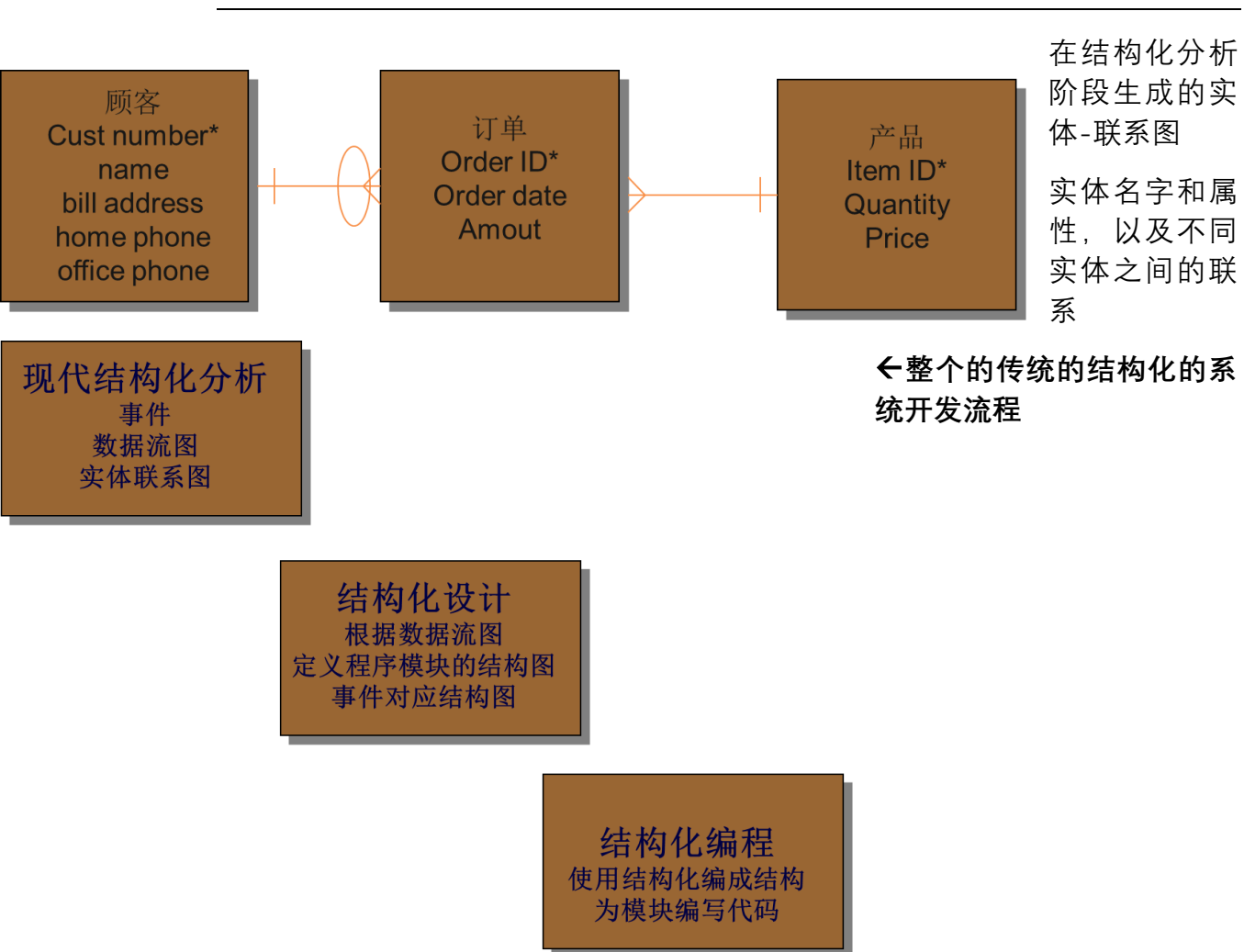
- **结构化设计技术**



- **结构化分析技术**

使用结构化分析技术生成的数据流图





➤ 信息工程方法

- 将系统的功能转化成数据流/信息流进行分析设计；
- 从全面的战略规划开始
- 使用集成的 CASE 工具

3.2.2 面向对象方法

➤ 定义

- 把信息系统看作一起工作来完成某项任务的相互作用的**对象集合**，通过**对象之间消息的传递和响应完成任务**

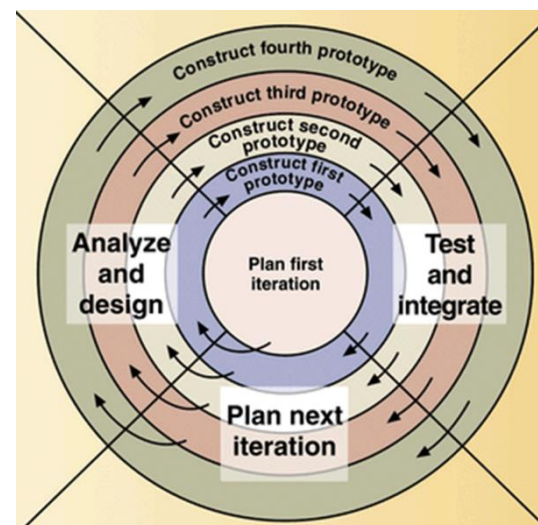
- 对象：计算机中可以对消息做出响应的事物

➤ 面向对象分析(OOA)

- 定义在系统中工作的所有类型的对象，并显示这些对象之间的相互关联

➤ 面向对象设计(OOD)

- 定义和系统中人机交互所必须的所有类型的对象，并对每一种类型的对



➤ 以人为重点的变体---把信息系统看作社会技术系统

完全是从人的角度去看待整个系统, 关注该系统的使用人群特点, 设计过程中更强调人机互动

➤ 基于开发速度的变体

- 快速开发的原因

- 市场对信息系统的大量需求
- 不断变化的技术和商业环境
- 开发过程缓慢: 返工、需求改变

- 定义

- 快速应用程序开发(RAD): 通过使用各种技术, 极大地加速系统开发过程

- 快速开发方法

- 原型化开发方法

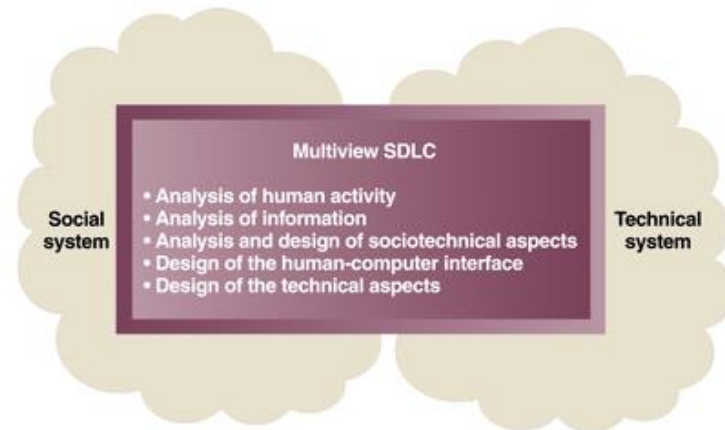
基于之前已经实现过的系统, 复用之前系统的组件/模块, 快速的开发出系统原型, 在原型的基础上进行设计和新的开发以及需求的确认

关键是实现软件的重用: 子系统级别重用、组件级别重用和对象级别的重用, 重用性最高的是执行代码重用(.class 文件等)

- 螺旋型开发方法

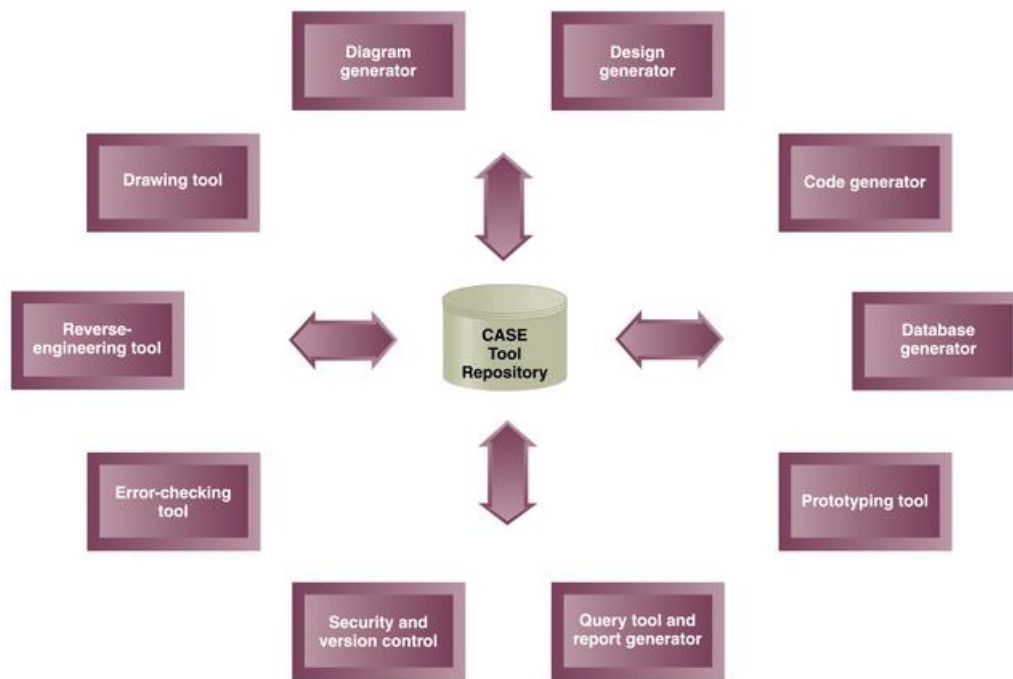
- 快速开发技术

- **风险管理技术:** 对未来的风险进行评估, 因为刚开始的开发是基于之前系统的复用, 并不完全符合需求
- **JAD:** 联合分析设计会议, 对复用的旧部件在新系统如何进行适用性的改变, 可能会出现什么样的问题
- 基于工具的开发
- **软件重用:** 对象框架、组件



3.4 计算机辅助系统工程(CASE)工具

CASE 工具存储库包含所有系统信息



3.5 分析和设计阶段细节

➤ 分析阶段

- 收集信息
 - 如何收集需求信息，对于有指向性功能的系统更多的去关注客户实际日常工作的实际需求
- 定义系统需求
 - 从收集到的大量的客户需求中识别出正确的系统需求
 - 对于模糊不清晰的系统需求要标注后再次找需求方确认
- 区分需求优先级
 - 哪些需求是必须做，哪些可做可不做
 - 找出工程中主要的施力方向
- 建立可行性原型和发现原型
 - 建立多个方案，用同一个评价体系去评价所有的方案后，选出最合适的
 - 应该使用可视化的模型去描述系统，即使用原型去描述系统，防止语言描述的二义性
- 生成和评价可选方案
 - 主要是定义评价的采分点

-
- 和管理人员复查推荐方案

第 4 章 调查系统需求

4.1 功能和技术需求

➤ 系统需求

- 对系统所提供的功能的作出详细定义
- 来源于计划阶段所确定的高层功能
- 系统需求分为**功能需求**和**技术需求**两类

例：计划阶段所定义的客户支持系统(css)的高层功能

客户支持系统(css)将满足下列目标：

1. 支持联机的客户、订单、退还订单、退货处理
 2. 支持传统的电话订购和邮购销售
 3. 支持网络客户及目录产品的销售，包括网上的购买和订单跟踪
 4. 维持足够的数据库和历史信息以支持市场分析
 5. 为顾客提供交易的历史信息
 6. 24 小时支持订单发货
 7. 协调多个仓库的发货
- 功能需求：系统必须支持的功能和过程
 - 例如：创建订单，增加订单项，计算订单总额
 - 技术需求：描述操作环境和性能目标的需求
 - 例如：win2000，B/S，Internet 和 Intranet，数据库，语言，接口等环境；系统响应时间，可靠性等性能。

4.2 系统需求的来源

➤ 系统相关者

指对系统的成功实施感兴趣的人，包括：系统用户、项目客户和技术人员

- 技术人员：对于客户提出的需求进行技术上的确认

- 项目客户：项目的出资方，处于对成本的考虑，可能会限制系统业务范围

- 系统用户：使用该系统的用户

➤ 系统用户的分类

- 水平方向和垂直方向的用户
- 业务操作用户
- 查询用户
- 管理用户 借还书系统相关者→
- 主管用户



4.3 系统需求的识别

➤ 开发系统需求的方法

- 以新系统的逻辑需求模型为中心

识别现有系统的
处理过程




为新系统开发需求
建立模型


➤ 需求调查时的三个主要问题

- 要做什么：
 - 商业过程和运作是什么
 - 理解商业功能
- 怎样做：
 - 商业过程如何完成
 - 转向新系统的方法
- 需要哪些信息：
 - 信息需求是什么
 - 定义新系统必须提供的具体信息

➤ 信息收集的各种方法

- 

Rocky Mountain Outfitters—Customer Order Form



Name and address of person placing order.
 (Please verify your mailing address and make correction below.)
 Order Date / /

Name _____

Address _____ Apt. No. _____

City _____ State _____ Zip _____

Phone: Day () _____ Evening () _____

Gift Order or Ship To: (Use only if different from address at left.)

Name _____

Address _____ Apt. No. _____

City _____ State _____ Zip _____

☐ Gift ☐ Address for this Shipment Only ☐ Permanent Change of Address

Gift Card Message _____

Delivery Phone () _____

| Item No. | Description | Style | Color | Size | Sleeve Length | Qty | Monogram | Style | Price Each | Total |
|----------|-------------|-------|-------|------|---------------|-----|----------|-------|------------|-------|
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |

Method of Payment

Check/Money Order ☐ Gift Certificate(s) ☐ AMOUNT ENCLOSED \$ _____

American Express ☐ MasterCard ☐ VISA ☐ Other ☐

Account Number MO YR

Expiration Date

MERCHANDISE TOTAL _____

Regular FedEx shipping \$4.50 per U.S. delivery address
 (Items are sent within 24 hours for delivery in 2 to 4 days)

Please add \$4.50 per each additional U.S. delivery address

FedEx Standard Overnight Service

Any additional freight charges

International Shipping (see shipping information on back) _____

Signature _____

- ## Checklist for Conducting an Interview
- Before**
 - Establish the objective for the interview
 - Determine correct user(s) to be involved
 - Determine project team members to participate
 - Build a list of questions and issues to be discussed
 - Review related documents and materials
 - Set the time and location
 - Inform all participants of objective, time, and locations
 - During**
 - Dress appropriately
 - Arrive on time
 - Look for exception and error conditions
 - Probe for details
 - Take thorough notes
 - Identify and document unanswered items or open questions
 - After**
 - Review notes for accuracy, completeness, and understanding
 - Transfer information to appropriate models and documents
 - Identify areas needing further clarification
 - Send thank-you notes if appropriate

| ID | Issue Title | Date Identified | Target End Date | Responsible Project Person | User Contact | Comments |
|----|-------------------------|-----------------|-----------------|----------------------------|------------------|--|
| 1 | Partial Shipments | 6-12-2003 | 7-15-2003 | Jim Williams | Jason Nadold | Ship partials or wait for full shipment? |
| 2 | Returns and Commissions | 7-01-2003 | 9-01-2003 | Jim Williams | William McDougal | Are commissions recouped on returns? |
| 3 | Extra Commissions | 7-01-2003 | 8-01-2003 | Mary Ellen Green | William McDougal | How to handle commissions on special promotions? |

- 观察商业过程和工作流
- 建立原型
- 主持 JAD 会议
 - 将先期识别出的需求信息，通过原型的形式展示讨论，避免需求不符的情况

4.4 结构化遍历

► 定义

- 简称遍历，是对调查结果和根据这些结果建立的原型进行复查
- 一般来说，结构化遍历有几个大的时间点：
 - 首先是系统每完成一个大的功能模块都要针对这个功能模块进行结构化遍历；
 - 其次是系统完成后，整体的结构化遍历
- 遍历之前要：确定遍历的内容和时间，以及遍历的参加者

- 通知参与本次遍历的人员本次遍历的是哪些功能模块，以及会议的地点时间

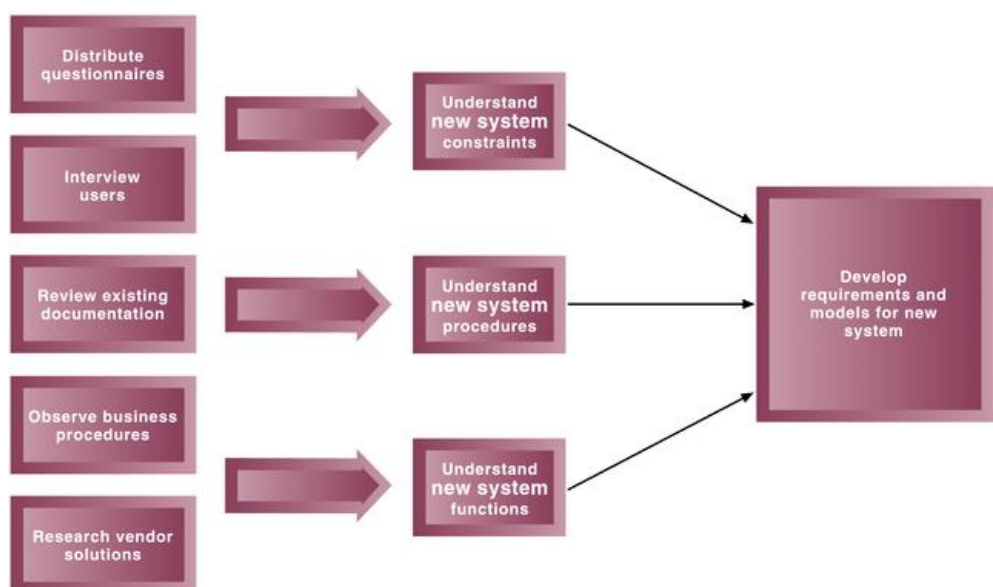
➤ 如何遍历

- 准备：本次遍历的功能模块的原型要建立完成
- 执行：会议中原型的演示，形成记录文档
- 后续工作：根据记录文档，进行需求分析和建模

4.5 业务流程重组 BPR

BPR 的好处是，**提高整个业务流程的效率**，具体就是将原有的业务流程按照信息系统的方式进行业务效率优化

在理解原有系统/流程的基础上，重组成新的信息系统，进而形成新系统的模型

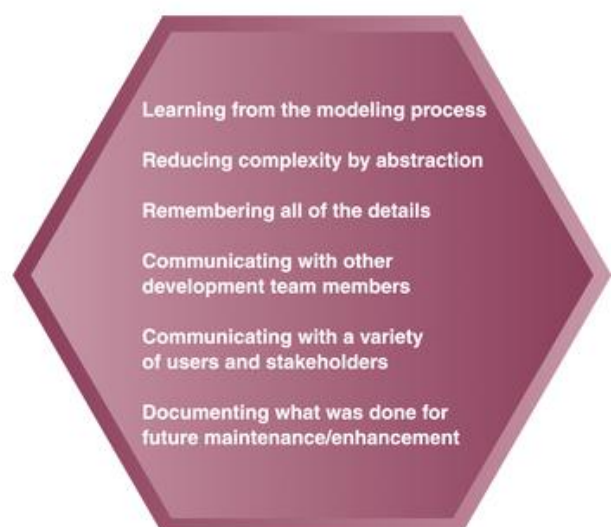


第 5 章 系统需求建模：事件和事实

5.1 模型与建模

➤ 建模的意义

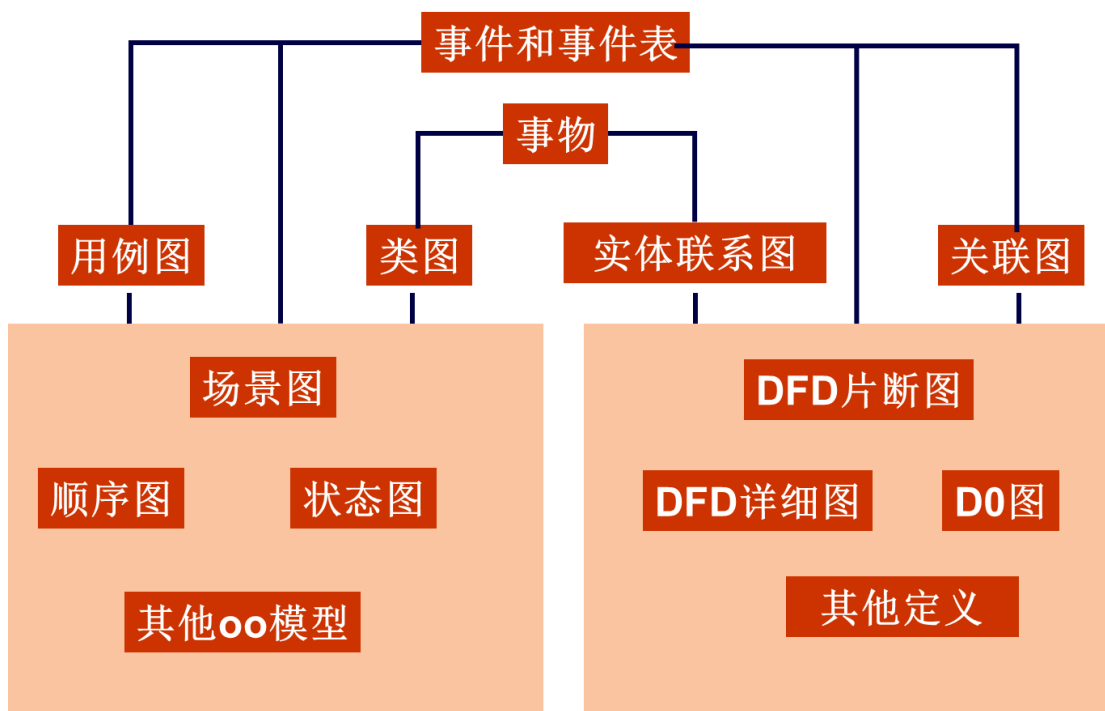
- 从建模过程中学习，更深刻的理解系统
- 通过抽象降低复杂度
- 记住所有细节，掌握更多的系统细节
- 与其他开发团队成员进行更高效的沟通，避免二义性
- 与各种用户和利益相关者进行沟通
- 记录为将来的维护/增强所做的工作



➤ 模型的类型

- 数学模型：方程、数学函数等
- 描述模型：程序设计语言描述等
- 图形模型：CAD 设计图等

➤ 系统分析和设计中使用的模型



- 关联图：把系统当做唯一的处理过程，绘制系统的交互过程
- 实体联系图即 ER
- 左侧的模型包括用例图和类图等模型都是面向对象分析方法用到的模型，右侧为传统的结构化模型

5.2 事件与系统需求

➤ 事件

- 事件是可以描述的、值得记录的、在某一特定时间和地点发生的事情
- 系统的所有处理过程都是由事件驱动或触发的
- 事件对定义系统需求的意义
 - 整体地看待系统，分析系统与外界的接口
 - 划分系统的一种方法：系统相关事件和系统无关事件，只有系统相关事件才会触发系统动作，因此只需要分析系统相关事件

➤ 系统相关事件的类型

- 系统外部事件：通常由外部实体或动作参与者触发
 - 外部实体想要通过该系统得到某种结果或者反馈，比如用户从图书馆系统中查找馆藏数量
 - 外部实体需要一些信息
 - 更改的数据需要更新
- 系统临时事件：由于到达某一时刻所发生的事件
 - 需要内部输出即需要在系统内部输出一些东西，例如：

管理报告（摘要或例外）、运营报告（详细交易）、内部声明和文件（包括工资单）
 - 需要向外部主动发出一些输出，例如：声明，状态报告，账单，提醒
- 状态事件
 - 当系统内部发生了需要处理的情况时所引发的事件，例如系统崩溃、系统数据丢失等触发的事件

Temporal Events to Look for Include:

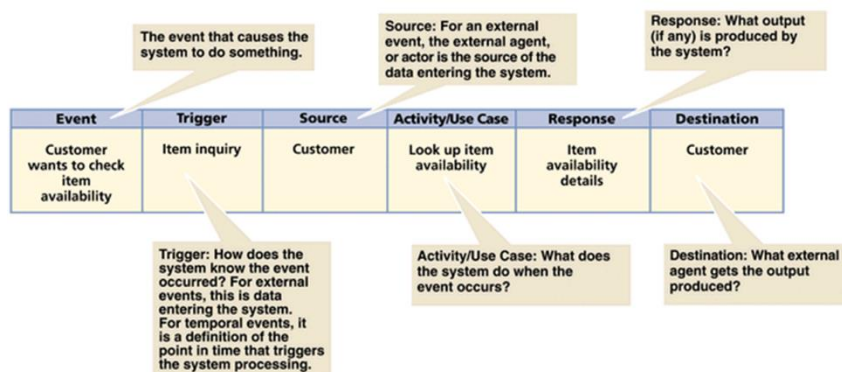
- √ Internal outputs needed
 - √ Management reports (summary or exception)
 - √ Operational reports (detailed transactions)
 - √ Internal statements and documents (including payroll)
- √ External outputs needed
 - √ Statements, status reports, bills, reminders

➤ 识别事件的方法

- 事件/条件和响应
- 事件时序：从一个客观的场景出发，按照一定的事件发生的时序列出系统的所有事件
- 在系统分析过程中，不会考虑技术依赖事件和系统控制
 - 在系统分析过程中，是基于理想的技术假设
 - 例如，像用户输入用户名密码错误，系统要做出怎样的动作，这样的事件属于技术依赖事件和系统控制

➤ 事件的构成(五要素)

- 触发器 Trigger
 - 事件的触发条件是什么
- 来源 Source
 - 事件的来源是什么
- 动作 Activity/Use case
 - 事件被触发后系统的动作是什么



- 响应 Response
 - 系统对于事件的响应动作
- 目的地 Destination
 - 事件的终结点在哪

注意：

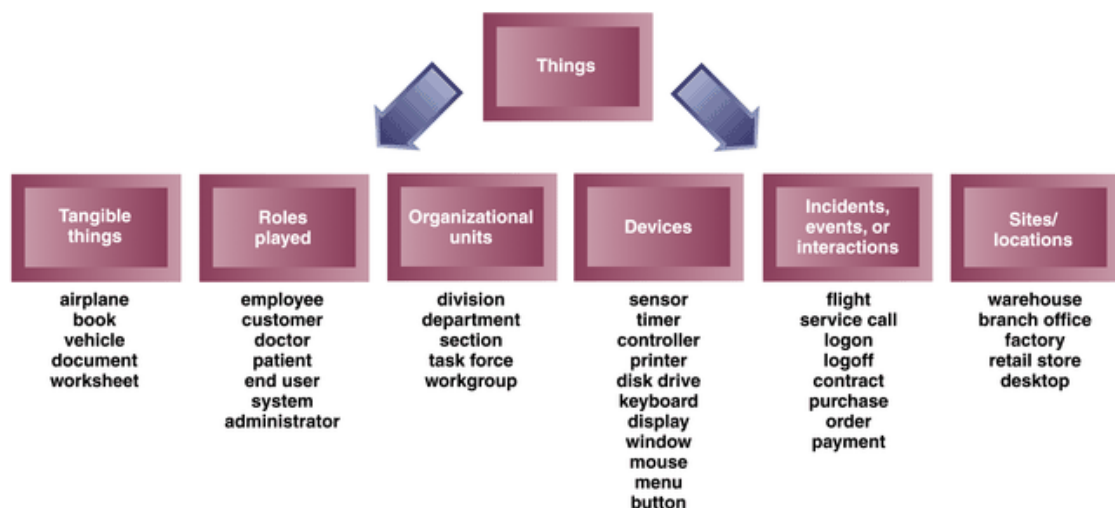
对于临时事件，其事件来源是系统内部而目的地是用户

5.3 事物与系统需求

➤ 事物

- 传统开发方法中的数据
- OO 开发中相互交互的对象

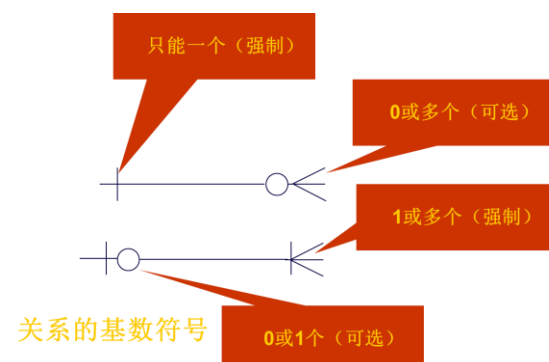
➤ 事物的类型



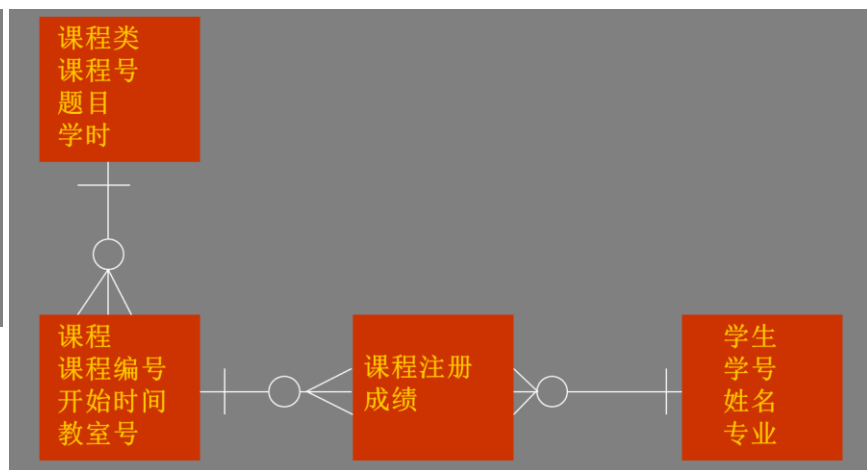
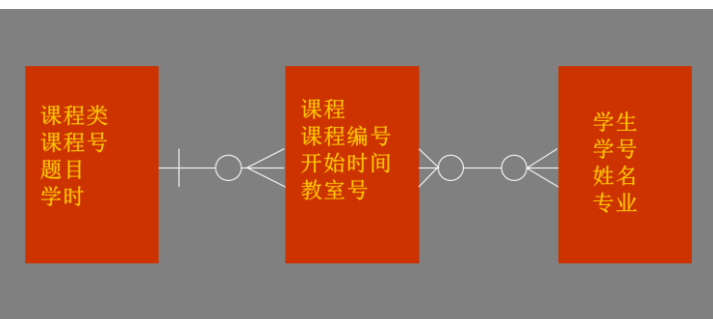
- 客观存在的事物：书籍、文件……
- 角色类型：管理员、读者……
- 组织事物：部门、小组……
- 设备事物：传感器、计时器……
- 事件类型事物
- 位置/地点事物

➤ 事物之间的关系

- 基数/重数 (Cardinality/multiplicity)



- 发生在事物间关联的数目
- 基数的取值范围
 - 0...*, 1, 1...*
 - 通过引入**关联实体**把多对多关系进行转化



- 二元关系、一元关系、三元关系、n 元关系
 - 指明有几个事物参与联系

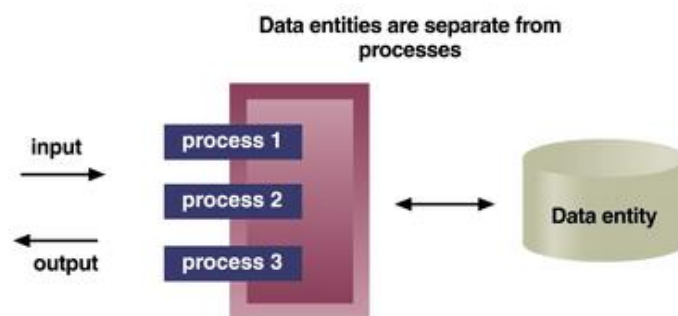
➤ 事物的属性

- 属性
- 标识符/关键字
- 复合属性

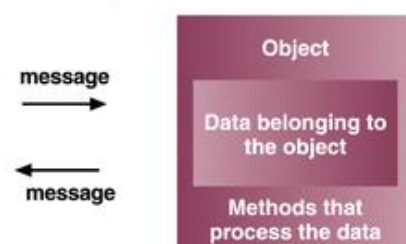
➤ 对象

- 类 class
- 方法 method
- 封装 encapsulate
- 对象和数据实体的比较
 - 数据实体与流程(方法/操作)分离, 一个数据实体对应一个数据库表
 - 对象封装了数据和对数据的操作方法

| All customers have these attributes: | Each customer has a value for each attribute: | | |
|--------------------------------------|---|----------|----------|
| Customer ID | 101 | 102 | 103 |
| First name | John | Mary | Bill |
| Last name | Smith | Jones | Casper |
| Home phone | 555-9182 | 423-1298 | 874-1297 |
| Work phone | 555-3425 | 423-3419 | 874-8546 |

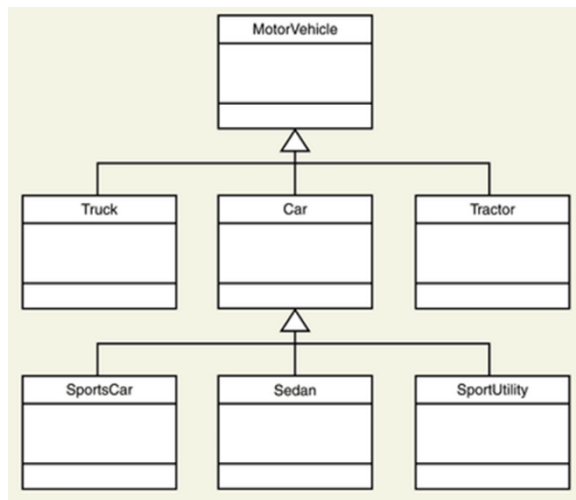


Objects encapsulate data and the methods that process the data into one unit

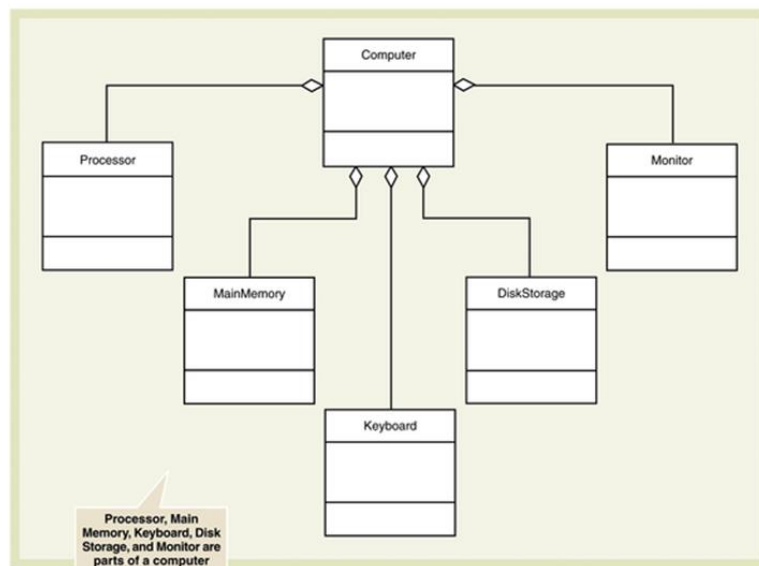


5.4 类图

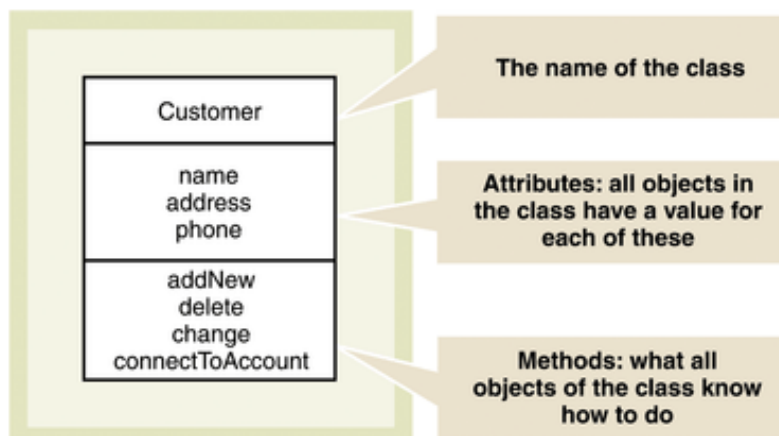
➤ 泛化



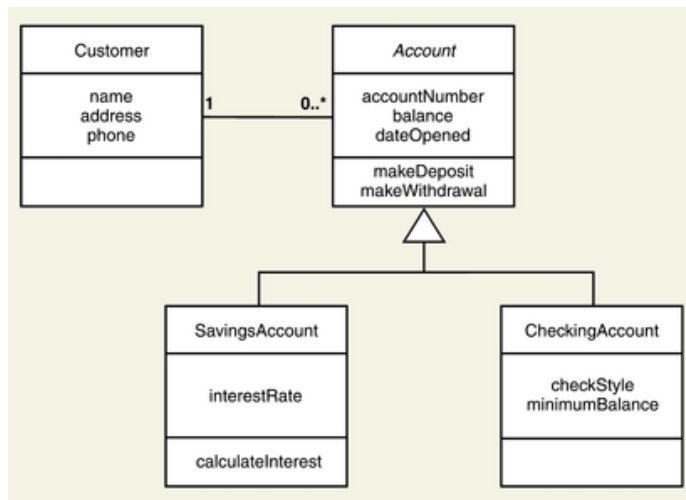
➤ 继承



➤ 类图符号的表示

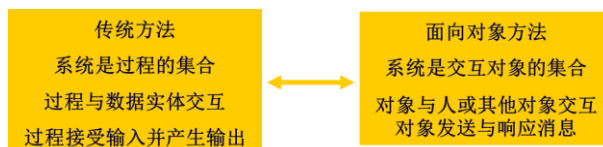


➤ 类图中的基数表示



第 6 章 传统的需求描述方法

➤ 传统观点和面向对象观点



本质上是一样的

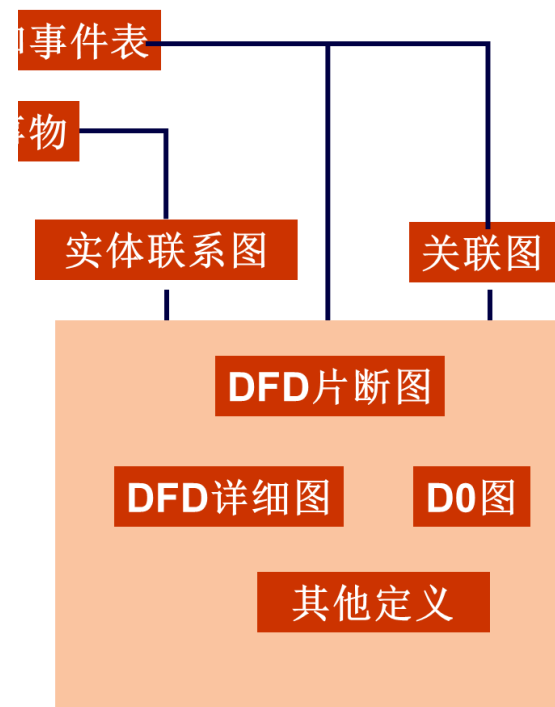
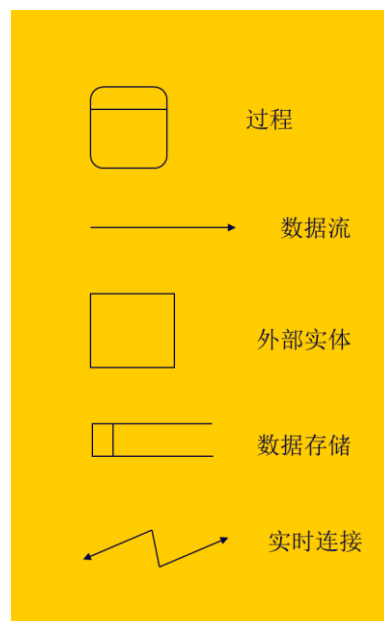
6.1 数据流程图

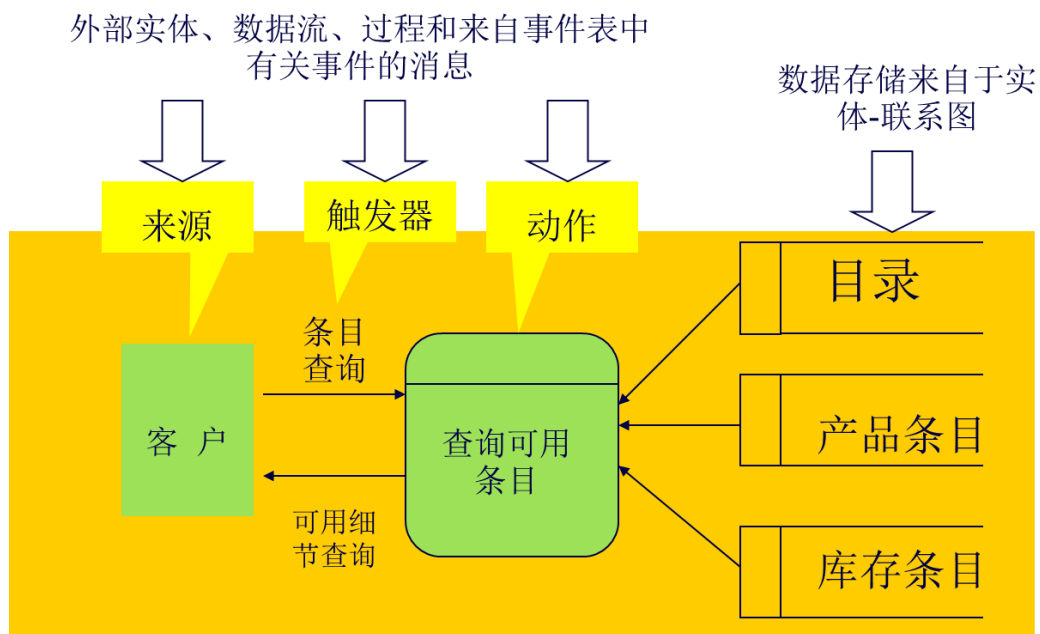
➤ 定义

数据流程图（DFD）是一种图像化的系统模型，展示信息系统的主要需求：输入、输出、过程和数据存储；

➤ 组成

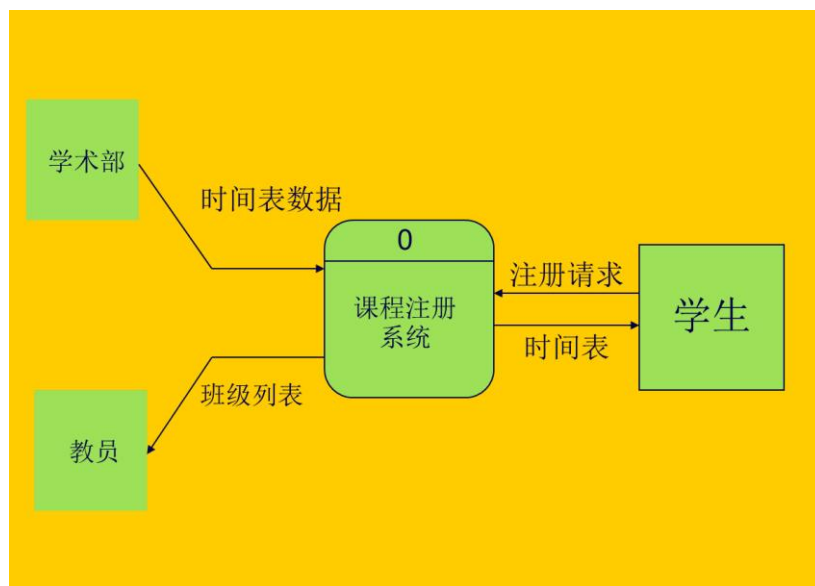
- 外部实体
- 数据流
- 数据存储
- 过程
- 实时连接





➤ 关联图

- 关联图是指描述系统高层结构的 DFD，所有的外部实体和进出系统的数据流都画在一张图中，并且整个系统被表示成一个过程；
- 系统内部在单个过程符号中概括所有处理活动的 DFD；

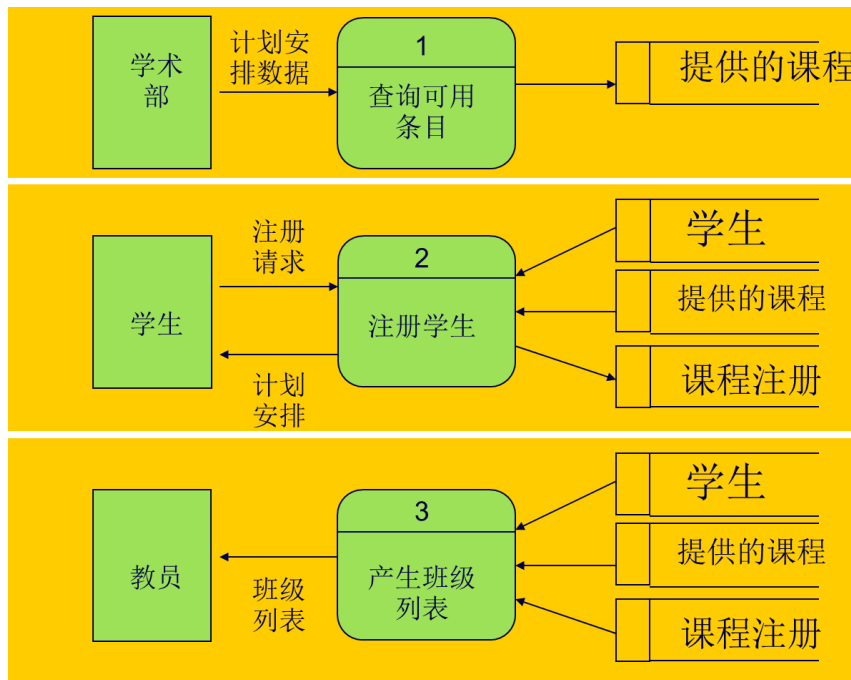


对于关联图中，和系统交互的每个事件都可以画出一个关联图

6.2 DFD 片段

➤ 定义

- 用于一个过程符号表示系统响应一个事件的 DFD



都是从关联图中和系统的交互事件对应来的

课程注册系统的DFD片段

➤ 事件划分的系统模型（0 层图）

- 一个为系统需求建立模型的 DFD，建模过程中对应于系统或子系统中每个事件使用单个过程；
- 0 层图：事件划分的系统模型的同义词；
- 是针对每个事件的，涉及到数据存储，是一种流程图片段；

➤ 逻辑 DFD 和物理 DFD

DFD 可以是一个物理的系统模型，也可以是一个逻辑的系统模型，当然也可以是二者的结合；

➤ 评估 DFD 质量

- 复杂性最小化：DFD 图看起来不能太复杂
- 信息超量：一个 DFD 图中的数据数量不能太多，根据 7 ± 2 原则
 - 接口最小化
- 数据流一致性：底层和顶层的 DFD 图在输入和输出上保持一致
 - 平衡、黑洞(顶层的 DFD 中有的输入底层 DFD 中缺失了)、奇迹(底层 DFD 中出现了顶层 DFD 没有的输入)

➤ 详细记录 DFD 部件

- 结构化英语：利用伪代码描述一个活动

```

借书 (读者 ID, 图书 ID) {
1、验证读者信息 (读者 ID);
2、检查读者可借图书信息 (读者 ID);
3、检查图书信息 (图书 ID);
4、借书 (图书 ID, 读者 ID) {
    填写借书单 (读者 ID, 图书 ID, 借书时间);
    更新读者信息 (读者 ID);
    更新图书信息 (图书 ID);
}
}

```

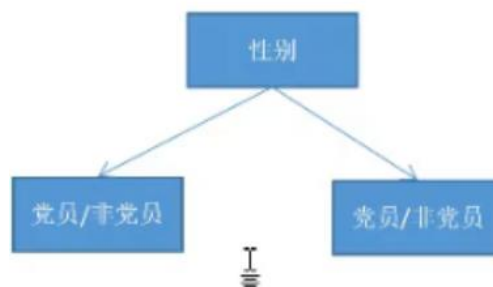
- 决策表

学生分类 () {

| 男 | | 女 | |
|----|-----|----|-----|
| 党员 | 非党员 | 党员 | 非党员 |
| | | | |
| | | | |
| | | | |

}

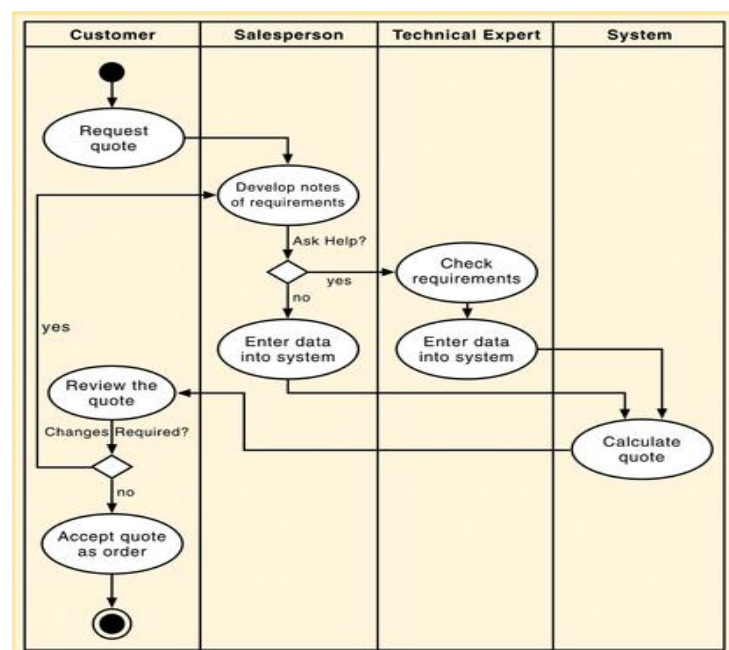
- 决策树



- 数据流定义：数据流内容和内部结构的文本描述；
- 数据元素定义
- 数据存储定义

➤ 信息工程模型(了解)

- IE 系统开发生命周期
- IE 和结构化开发的比较
- 过程分解和依赖模型
- 考虑网络节点和通信
- workflow 建模



第 7 章 面向对象的需求描述方法

7.1 面向对象的需求

7.1.1 类图

识别组成新系统的对象并进行分类。

- 提供系统组成部分的定义
- 包含系统程序和数据库两方面的信息
- 包括与问题域相关的类和实施类

7.1.2 用例图

识别不同的用户角色对系统的使用。

➤ 组成

• 用例：

描述系统在对事件做出响应时所采取的行动。表示系统为用户完成的一个单一用途或功能。

• 参与者：

系统用户扮演的一个角色

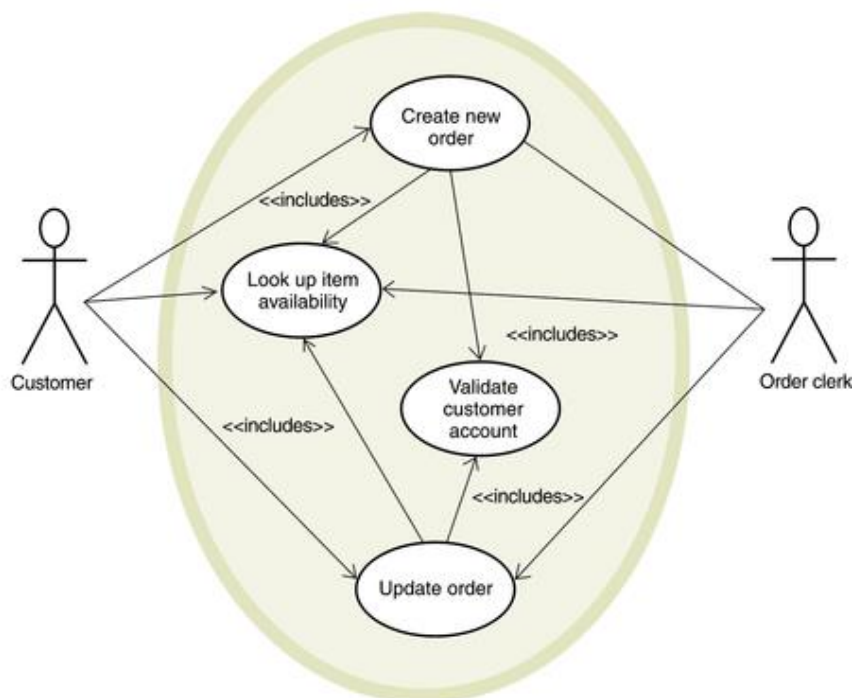
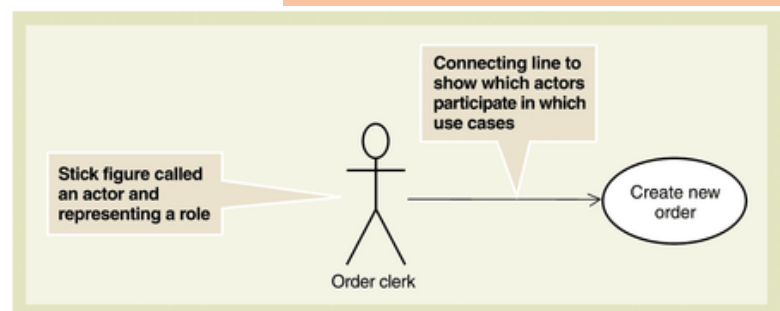
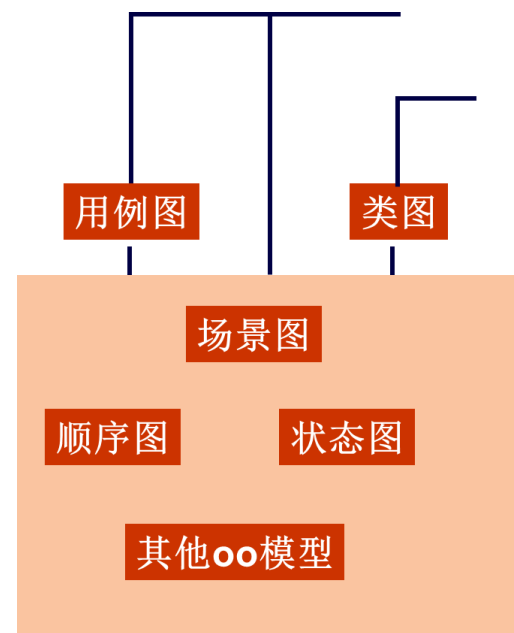
➤ 场景

- 在用例中的一个特定的活动顺序
- 一个用例可能有多个不同的场景

➤ 用例之间的包含/使用关系

➤ 开发用例图步骤

- 列出系统事件表
- 确定所有使用系统的参与者
- 开发场景

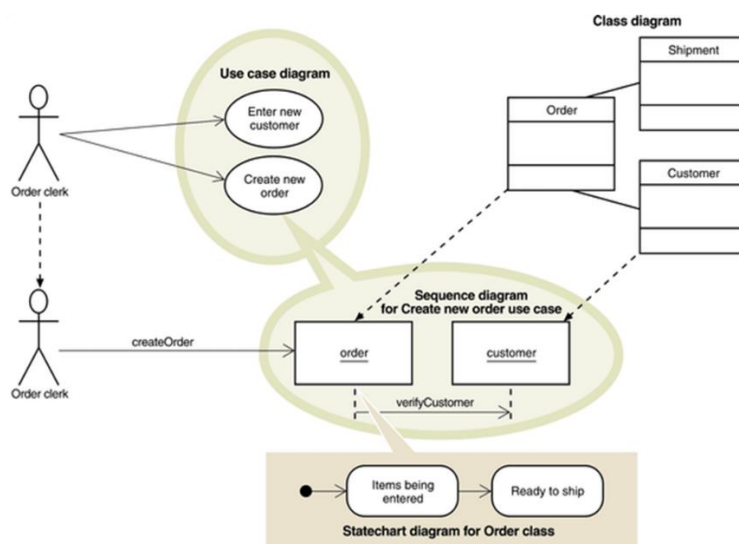


- 确定通用的内部用例（可以进行用例的复用）
- 划分用例

7.1.3 交互图

识别一起执行用例的对象间的协调

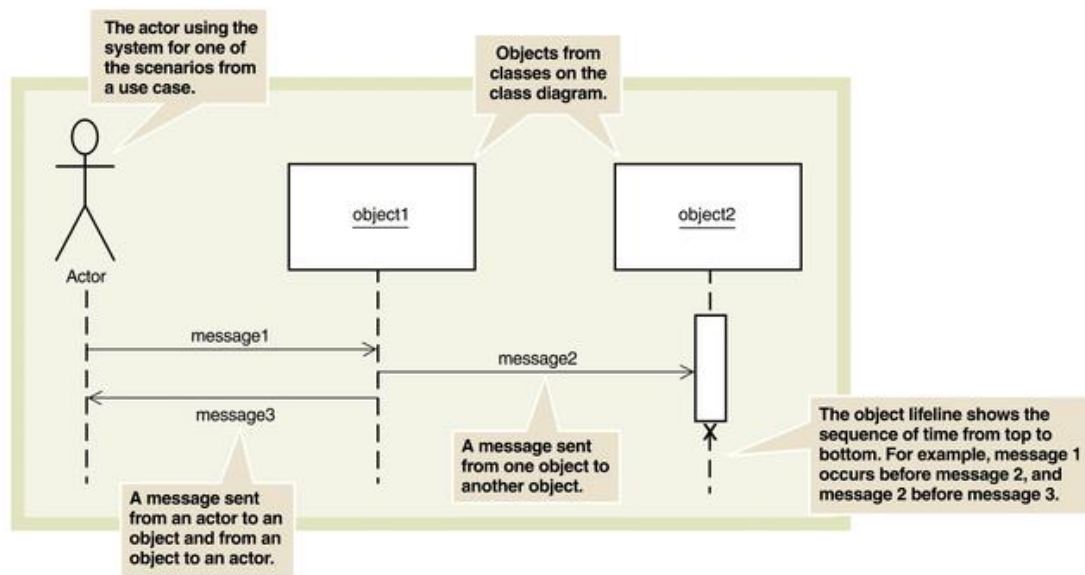
● 用例图、类图和交互图之间的关系



- 时序图中的参与者是用例图中的 actor，若该事件有多个参与者，可以添加时序图的 actor；
- 时序图中的实体一定要对应类图中的类，不要添加 UI 类、system 类或者 database 类等一些业务无关类；
- 时序图的绘制根据事件表，从来源参与者开始，由于触发器被触发，和一些对应的实体进行数据交互，最后数据流返回到目的参与者；

➤ 顺序图——识别对象间的消息的顺序

- 定义
 - 表示一个用例或场景中发生事件流时的对象之间的交互
 - 表示对象之间的交互和一组消息的顺序
- 组成



- 参与者：参与者来自于用例图中系统某个场景的用例
- 对象：对象是来自于类图中的实体类
- 生命线
- 消息：由方向箭头和消息描述器两部分组成

消息描述器的语法

[true/false 条件] 返回值 := 消息名 (参数列表)

示例：

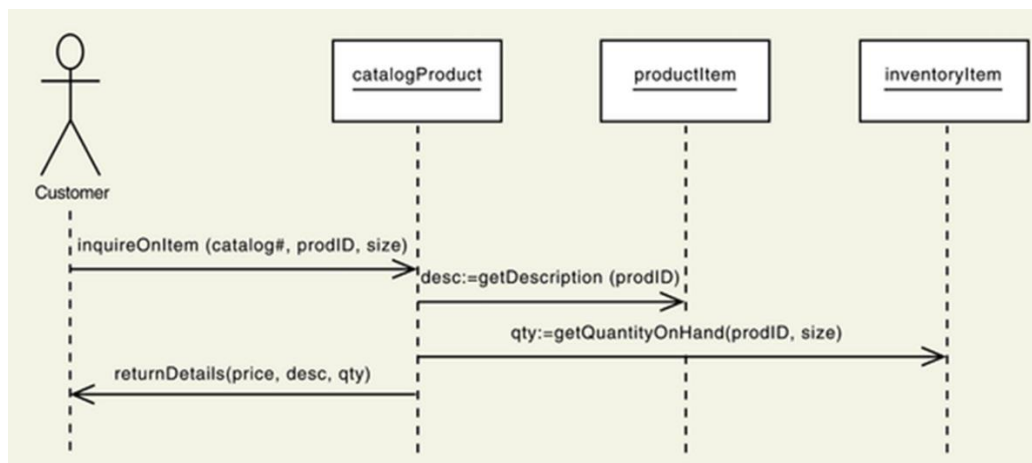
RingTelephone

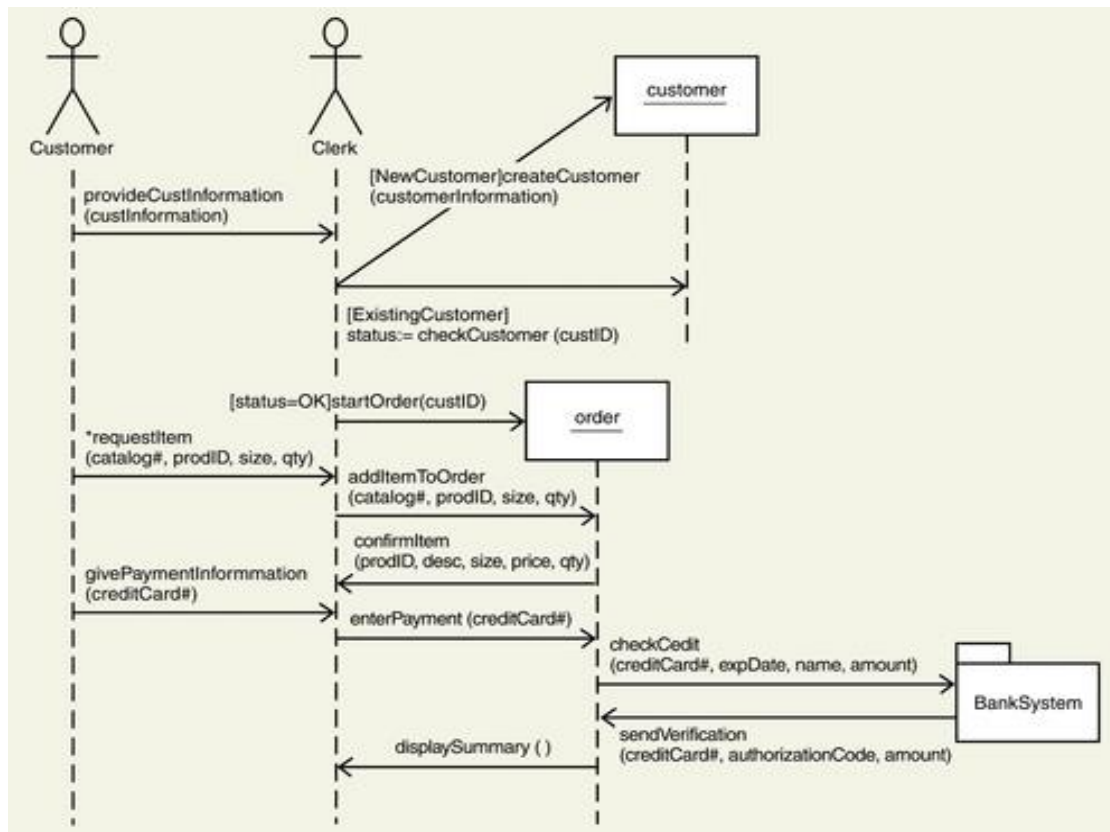
ItemInquiry ()

CreateOrderItem (ItemId,qty)

[FirstItem]OrderNumber := CreateOrder()

• 顺序图实例





使用返回值的形式则可以不用返回的消息箭头，可以合并一个两个对象之间的交互

● 顺序图开发步骤

- 确定对象和参与者
- 确定消息
- 确定消息的条件
- 确定消息的顺序
- 确定消息的形式化语法
- 进一步完整

➤ 协作图——识别协调的对象

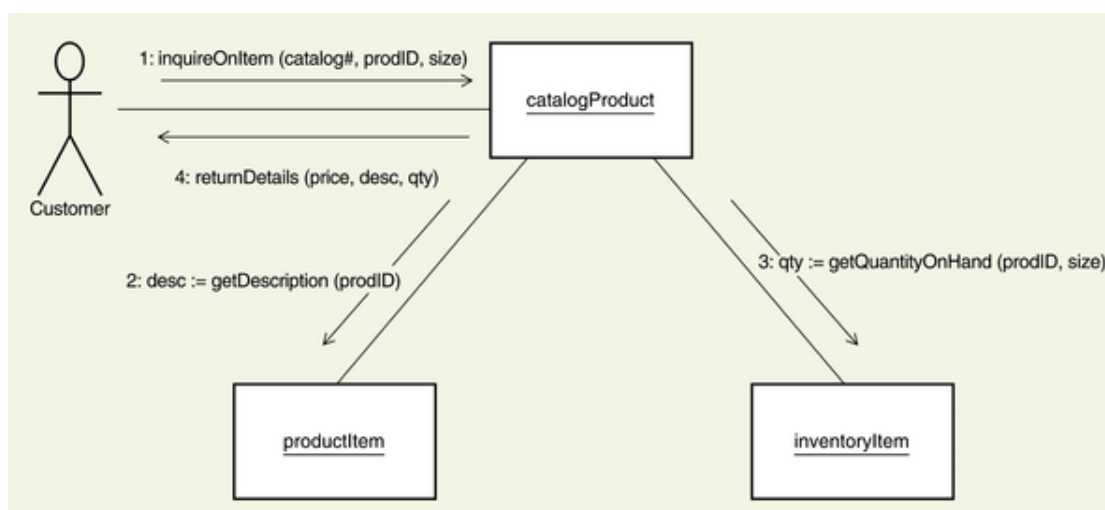
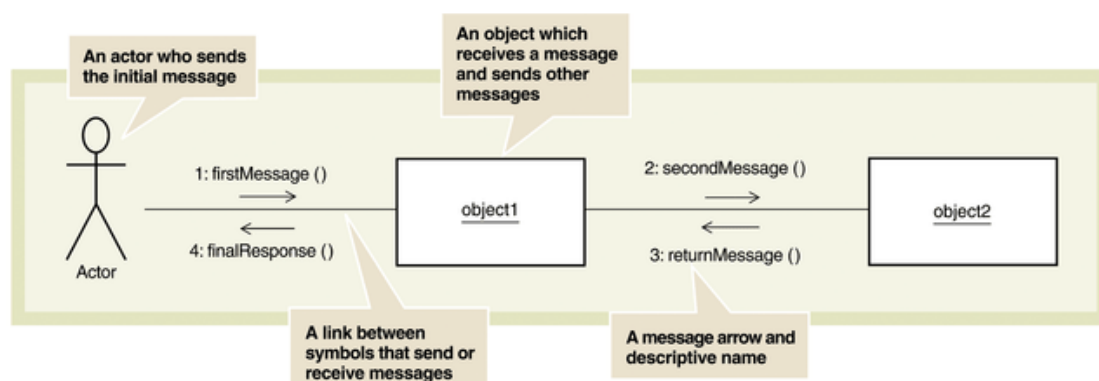
● 定义 和时序图/顺序图是等价的

- 用于快速浏览一个用例或场景中相互协作的所有对象，重点在于对象之间的协作。
- 符号：参与者、对象、消息、连接符号
- 消息的语法： 用顺序序号替代生命线表示时序

[true/false 条件] 顺序编号：返回值 := 消息名（参数列表）

- 示例

清晰的描述一个事件的所有参与者/实体及其之间的数据交互



7.1.4 状态图

识别对象的状态和转换

➤ 对象状态 Object State

- 状态指对象存在的条件
 - 满足某些标准、执行一些行为、等待一个事件
- 状态是对象的半永久条件
- 外部事件引起对象转移到新的状态

➤ 对象转换 Object Transitions

使对象离开一个状态并转变到一个新状态的机制

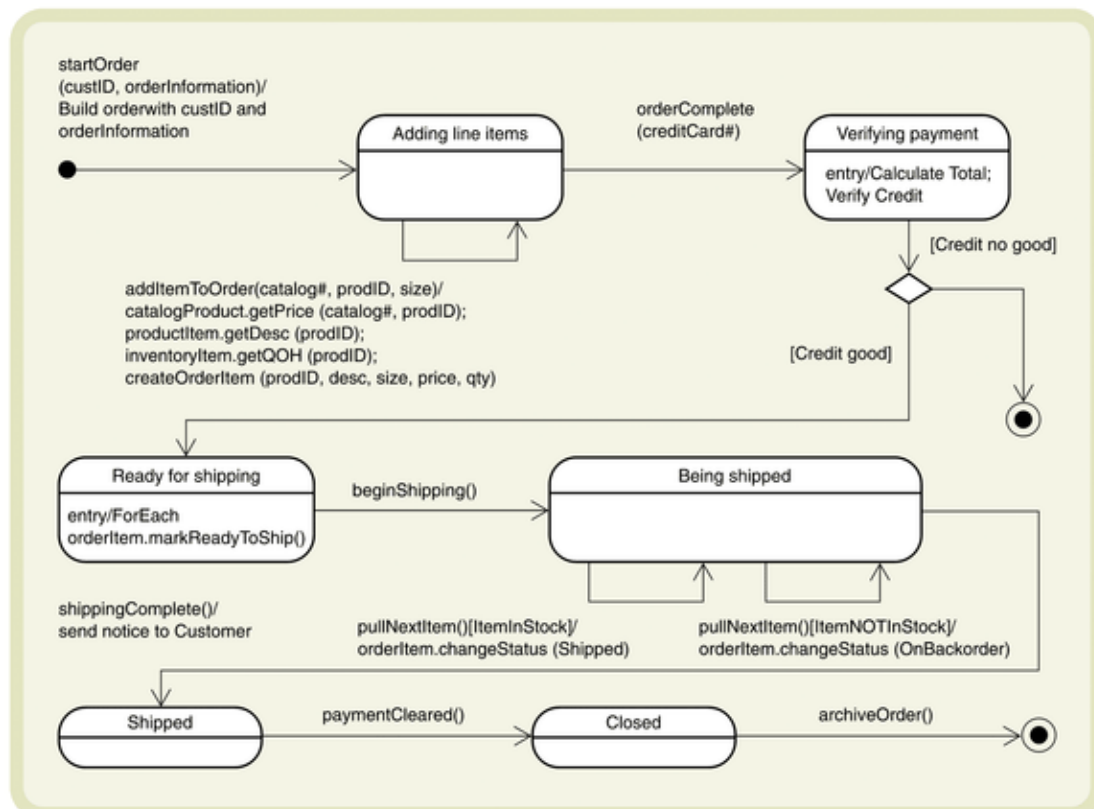
- 转换标签

转换名 (参数, ...) [保护条件] / 动作表达式

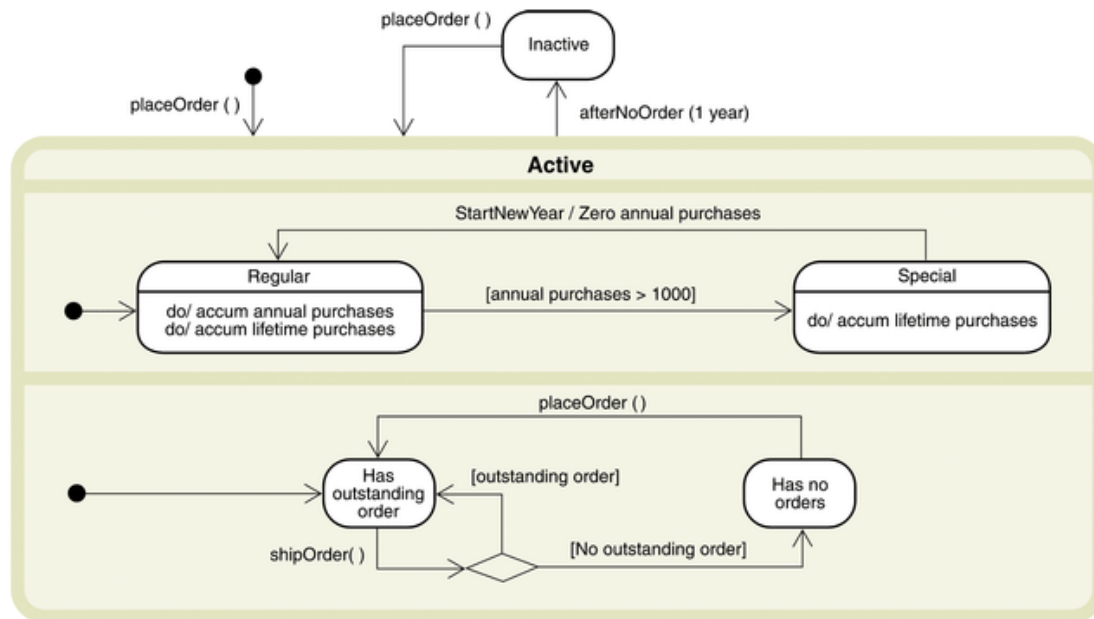
➤ 状态图的开发

- 检查类图，选择需要开发状态图的类
- 检查该类的所有顺序图，开发状态
- 开发转换，并进行完善
- 检查和测试状态图

➤ 状态图示例



- 并发行为



- 两个不同的对象均可以从一个事件被引起，则可以采用并发上下的形式描述两个对象在时空上的并发执行；
- 也可以是一个对象由于引发的不同条件可能转成不同的两个状态，也可以并发描述

第 8 章 环境、方案与决策

8.1 评估环境

➤ 确定目标处理环境

- 指要确定新系统的使用的计算机设备、操作系统以及网络配置
- 有集中式系统、分布式系统两大类

➤ Internet 、 Itranet 和 Extranet

- 优势：
 - 可访问性、低通信费、广泛使用的标准
- 不利因素
 - 安全性、可靠性、吞吐量、易变的标准

➤ 确定开发环境和系统软件环境

- 语言环境，使用何种编程语言
- 使用何种 CASE 工具和方法论
- 与其他系统的接口

- 操作系统环境
- 数据库管理系统环境，使用什么数据库

8.2 决定系统的范围和自动化水平

➤ 分析范围

范围扩充（scope creep）

范围表：将系统功能的需求程度分等级，定义产品实现功能的边界范围。

➤ 分析自动化水平

给每项功能定义不同的自动化解决方案，定义每项功能其自动化的水平。

➤ 评估确定系统范围和自动化水平

8.3 系统实施方案分析

1. 托管：

直接购买符合本系统需求的第三方服务/产品，例如阿里云服务等，无需购买其软件，只是付费使用其功能；

2. 软件包和 ERP：

如果有第三方产品满足需求，则直接购买其现成的软件产品；

3. 客户化开发：

如果购买的产品和需求有偏差，则根据需求进行针对性的二次开发；

4. 自行开发：自行从头实现系统需求；

8.4 选择实施方案

➤ 确定选择标准

- 通用需求标准评价

| General requirements criteria | Weight (5=high, 1=low) | Alternative 1 In-house | | Alternative 2 Package #1 + modify | |
|-----------------------------------|------------------------|------------------------|----------|-----------------------------------|----------|
| | | Raw | Extended | Raw | Extended |
| Availability of experienced staff | 4 | 3 | 12 | 3 | 12 |
| Development cost | 3 | 5 | 15 | 5 | 15 |
| Expected value of benefits | 5 | 5 | 25 | 3 | 15 |

上图中，有“开发团队的开发经验”、“项目开发成本”、“预期获得收益”三个维度，根据我们将要进行的项目的需求预期，对每个维度赋予权重，纵向的两列是两个不同的方案，根据选定的维度对两个方案进行打分，例如该方案需要开发团队经验的程度或者该方案需要的成本，对原始得分 Raw 乘以权重 Weight 后得出最终的 Extended。将纵向的 Extended 求和后，是该方案在通用需求标准下的得分。

- 功能需求标准评价

| Functional requirements criteria | Weight (5=high, 1=low) | Alternative 1 In-house | | Alternative 2 Package #1 + modify | |
|----------------------------------|------------------------|------------------------|----------|-----------------------------------|----------|
| | | Raw | Extended | Raw | Extended |
| Make inquiry on items | 4 | 5 | 20 | 4 | 16 |
| Create customer order | 5 | 5 | 25 | 5 | 25 |

- 技术需求标准评价

| Technical requirements criteria | Weight (5=high, 1=low) | Alternative 1 In-house | | Alternative 2 Package #1 + modify | | Alternative 3 Package #2 + modify | | Alternative 4 Custom development | |
|---------------------------------|------------------------|------------------------|------------|-----------------------------------|------------|-----------------------------------|------------|----------------------------------|------------|
| | | Raw | Extended | Raw | Extended | Raw | Extended | Raw | Extended |
| Robustness | 5 | ? | *18 | 3 | 15 | 4 | 20 | ? | *18 |
| Programming errors | 4 | ? | *16 | 4 | 16 | 4 | 16 | ? | *16 |
| Quality of code | 4 | ? | *18 | 4 | 16 | 5 | 20 | ? | *18 |
| Documentation | 3 | 5 | 15 | 3 | 9 | 4 | 12 | 4 | 12 |
| Easy installation | 3 | 5 | 15 | 5 | 15 | 4 | 12 | 4 | 12 |
| Flexibility | 3 | 4 | 12 | 3 | 9 | 4 | 12 | 5 | 15 |
| Structure | 3 | 4 | 12 | 4 | 12 | 4 | 12 | 4 | 12 |
| User-friendliness | 4 | 5 | 20 | 3 | 12 | 4 | 16 | 5 | 20 |
| Total | | | 126 | | 104 | | 120 | | 123 |

三项评价标准进行总求和后，得出总得分即该方案的总评价；

➤ 做出选择

- 根据评价标准得出的方案评分进行客观的推荐；
- 有可能在做完需求的分析以后，发现已有系统/产品满足功能，则直接购买即可，不一定得出方案的下一步就是自行实现系统；

➤ 生成 RFP (Request for Proposal 系统生成建议书)

- 需求者的描述和背景
- 提交建议的格式和方法

- 系统需求描述
- 评估标准

➤ 递交结果和做出决定

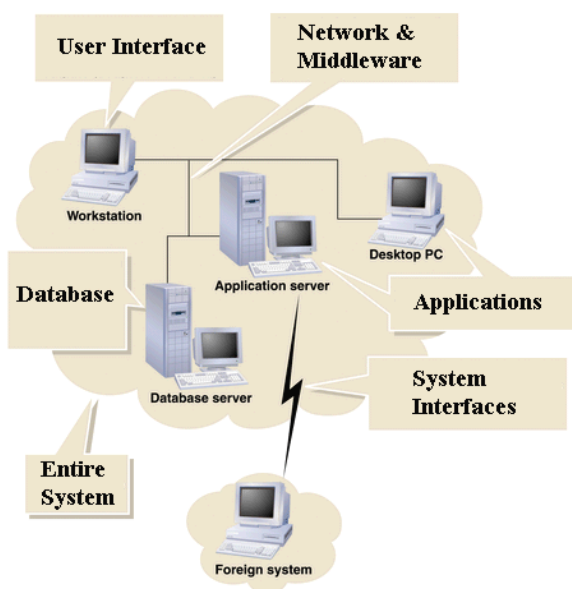
- 形成最终的推荐文档上交项目组进行讨论；
- 项目组讨论完成并且完善后，上交筹划(监督)指导委员会；

第 9 章 系统设计

9.1 设计要素

从分析阶段的需求建模，形成模型文档。到设计阶段根据分析阶段的产物进行：系统结构设计、输入输出设计、人机交互界面设计和数据库设计

➤ 设计阶段的主要活动

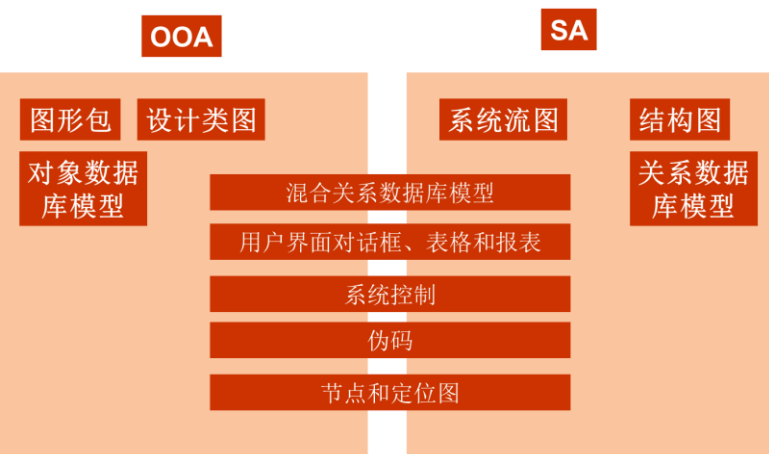


- 人机接口；
- 网络结构；
- 应用程序结构；
- 系统接口；
- 数据库设计；
- 系统控制设计

有面向对象和结构化两种系统设计模

式

系统设计的工作模型



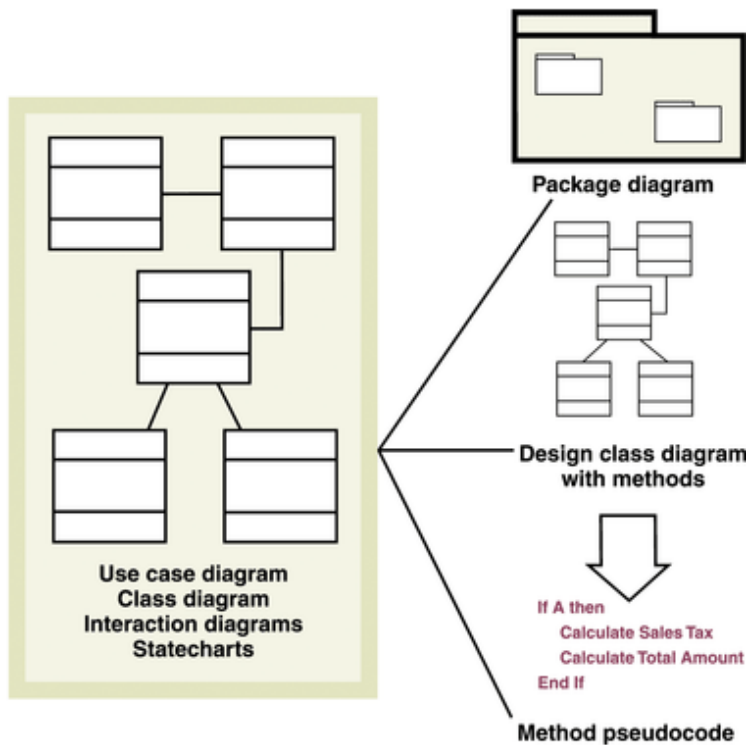
左侧为面向对象，右侧为结构化程序设计

除了两个部分特有的工作外，有结构化和面向对象共通的工作如左图所示

9.2 传统结构化应用结构设计（删去）

9.3 面向对象的应用结构设计

➤ 面向对象的设计模型



根据系统分析阶段产生的：用例图、类图、时序图、状态图等，设计出包图结构、设计类图，其中对于设计类图中的每个方法要形成方法的伪码结构。

➤ 包图

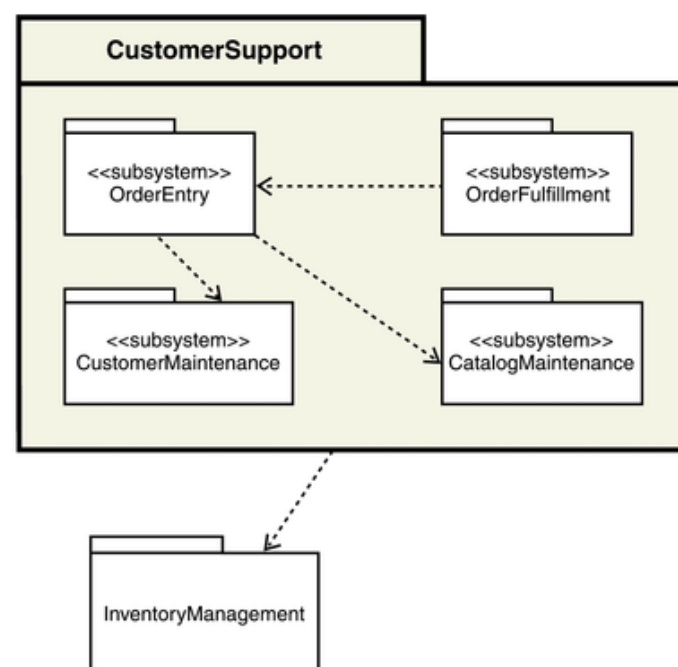
程序的每个功能所用的到类之间的关系是怎样的。

• 设计类包图

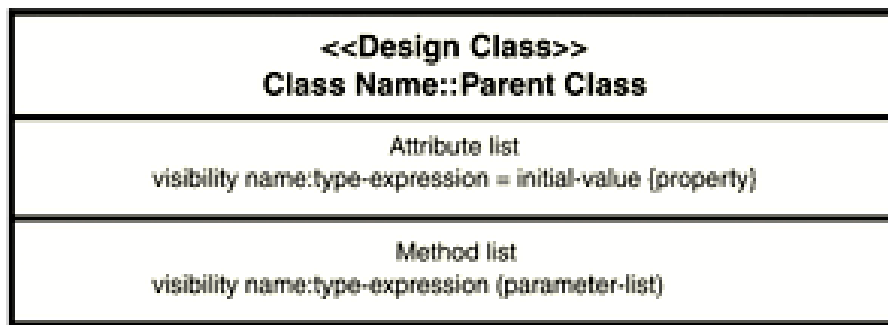
指明具体的类包括在哪个包的包图，并且标明类之间的依赖关系。

根据系统的功能模块，把系统划成各种各样的包，包之间会相互依存。

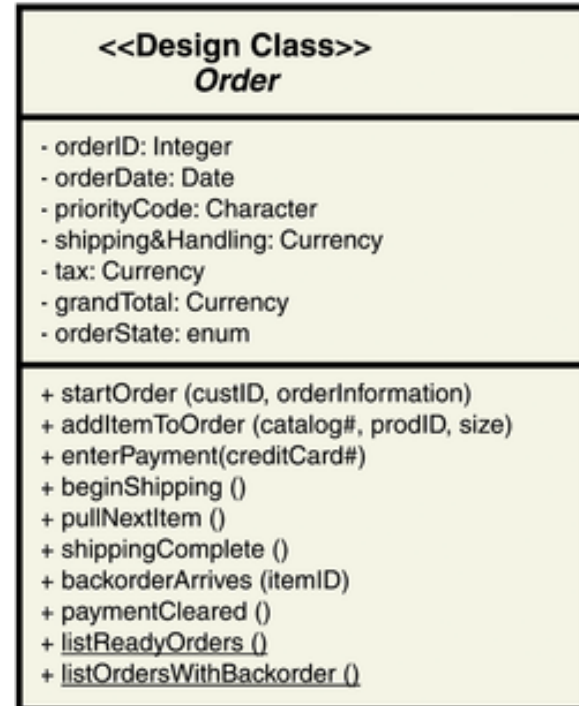
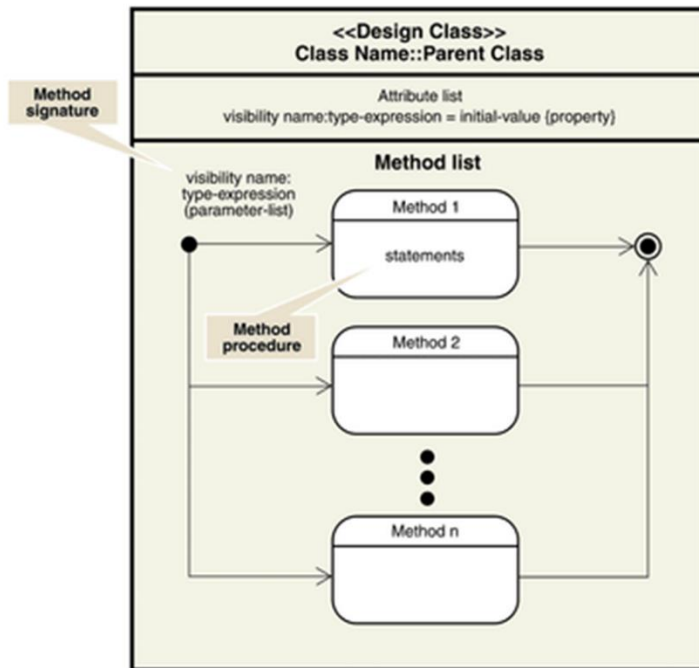
还需要指明包里面存在哪些类。



➤ 设计类图—详细的类图



- 声明所用属性的类型，即定义属性类型；
- 定义属性的可见性；
- 定义方法，包括返回值和参数以及可见性；
- 可以通过伪码表示方法的内部执行过程；



- 根据时序图获得完整的方法描述和参数，可以借机修正时序图；

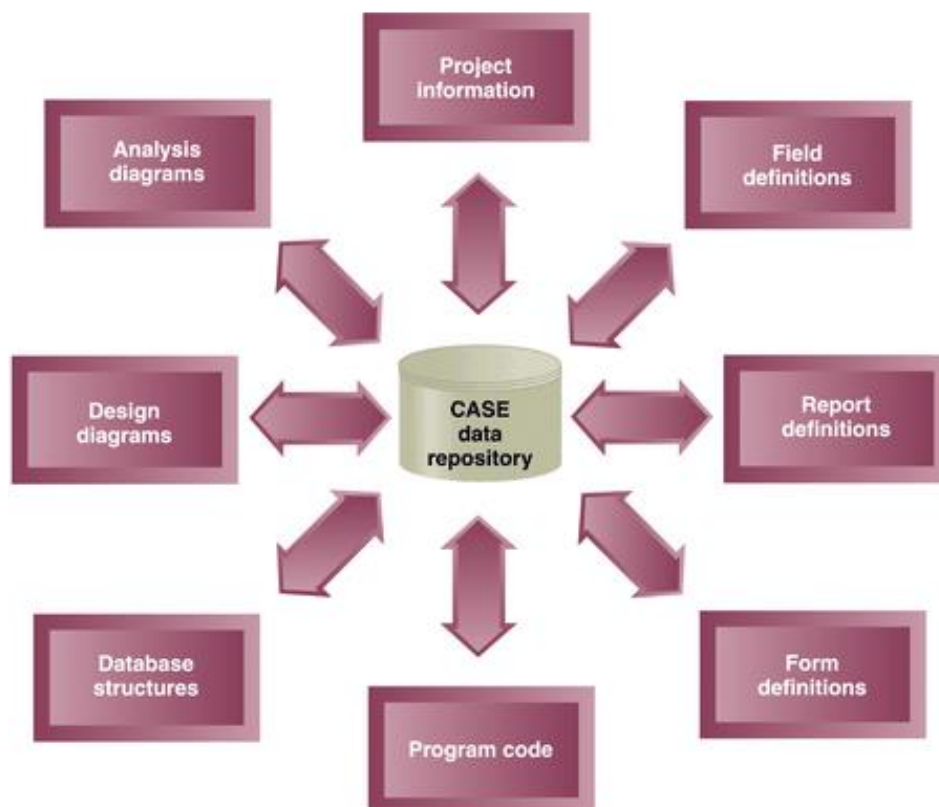
➤ 设计类图的开发

- 决定需要设计的类，建立属性列表
- 找到属于这个类的所有方法，考查顺序图
- 详细描述方法逻辑，考查状态图

9.4 项目协调

- 协调项目组成员

- 协调信息

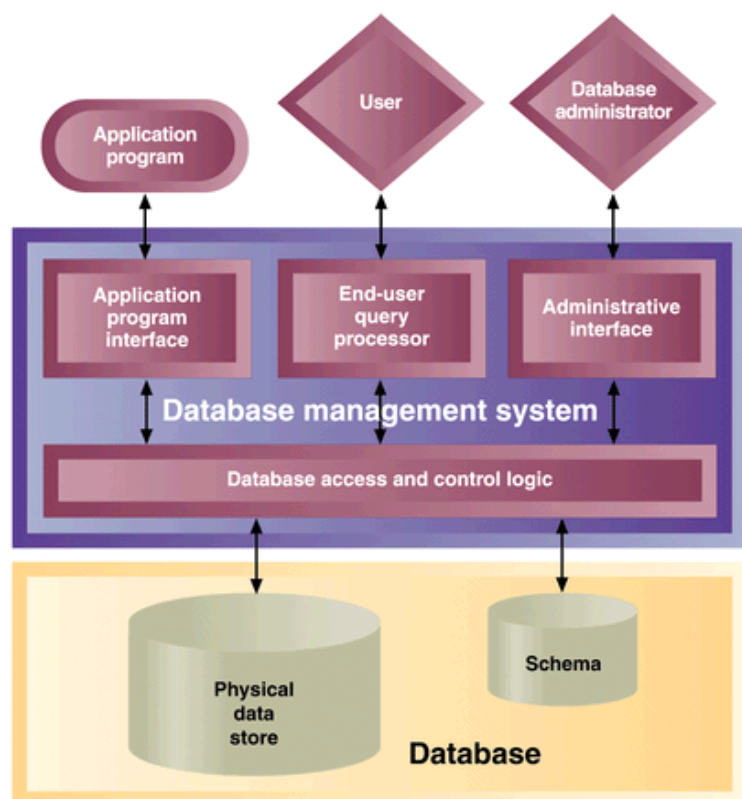


第 10 章 数据库设计

10.1 数据库与数据库管理系统

大概分为：层次、网状、关系和面向对象四种数据库

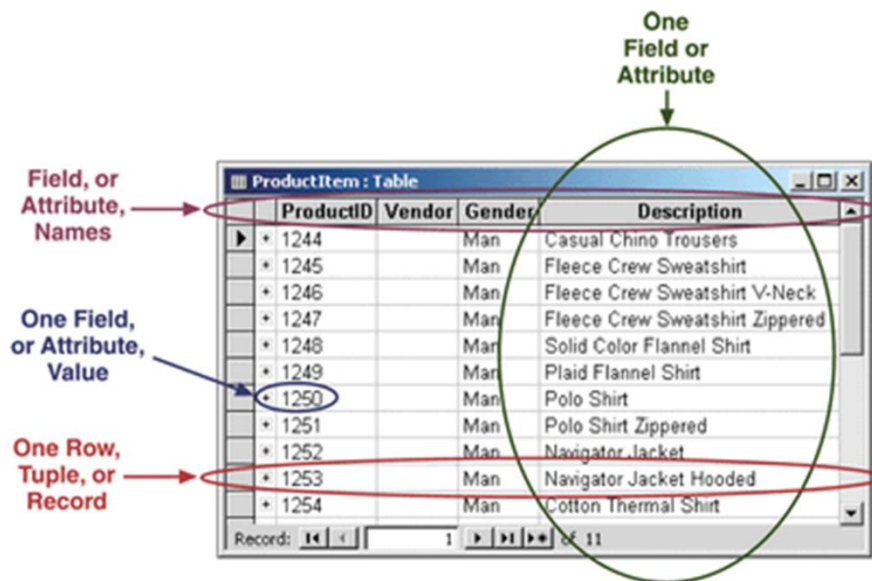
目前是关系性数据库较多, 对象数据库还在发展中



10.2 关系数据库模型

➤ 关系数据库管理系统的组成

- 表/关系
- 行/元组记录
- 字段/属性
- 字段值/属性值/数据元素
- 关键字
- 主键
- 外键



➤ 设计关系数据库

- 为每个实体（类）建立一个表
- 为每个表选择一个主键
- 增加外键以表示一对多关系
- 建立新表表示多对多关系
- 定义参照完整性
- 评价模式进行必要的改进
- 为每个字段选择适当的数据类型和取值范围

➤ 关系数据库设计实例

- 列出表名和表中的属性

| Table | Attributes |
|------------------|--|
| Catalog | Season, Year, Description, EffectiveDate, EndDate |
| Customer | Name, BillingAddress, ShippingAddress, DayTelephone Number, NightTelephone Number |
| InventoryItem | Size, Color, Options, QuantityOnHand, AverageCost, ReorderQuantity |
| Order | Date, PriorityCode, ShippingAndHandling, Tax, GrandTotal, DateReceived, ProcessorClerk, PhoneClerk, CallStartTime, LengthOfCall, EmailAddress, ReplyMethod |
| OrderItem | Quantity, Price, BackOrderStatus |
| OrderTransaction | Date, TransactionType, Amount, PaymentMethod |
| Package | Description, SalePrice |
| ProductItem | Vendor, Gender, Description, Season, NormalPrice, SpecialPrice |
| ReturnItem | Quantity, Price, Reason, Condition, Disposal |
| Shipment | TrackingNumber, DateSent, TimeSent, ShippingCost, DateArrived, TimeArrived |
| Shipper | Name, Address, ContactName, Telephone |

- 根据列出的表，选出主键（红色）

| | |
|-------------------------|--|
| Catalog | Number ,Season,Year,Deccription,EffectiveDate,End Date |
| Customer | AccountNumber ,Name,BillingAddress,ShippingAddress,DayTelephoneNumber,NightTelephoneNumber |
| InventoryItem | Number ,Size,Color,Options,QuantityOnHand,AverageCost,ReorderQuantity |
| Order | Number ,Date,PriorityCode,ShippingAndHandling,Tax,GrandTotal,DateReceived,ProcessorClerk,PhoneClerk,CallStartTime,LengthOfCall,EmailAddress,ReplyMethod |
| OrderItem | Number ,Quantity,Price,BackOrderStatus |
| OrderTransaction | Number ,Date,TransactionType,Amount,PaymentMethod |
| Package | Number ,Description,SalePrice |
| ProductItem | Number ,Vendor,Gender,Description,Season,NormalPrice,SpecialPrice |
| ReturnItem | Number ,Quantity,Price,Reason,Condition,Disposal |
| Shipment | TrackingNumber ,DateSent,TimeSent,ShippingCost,DateArrived,TimeArrived |
| Shipper | Number ,Name,Address,ContactName,Telephone |

- 根据列出的表。选出外键（蓝色）

| | |
|-------------------------|---|
| Catalog | Number ,Season,Year,Deccription,EffectiveDate,End Date |
| Customer | AccountNumber ,Name,BillingAddress,ShippingAddress,DayTelephoneNumber,NightTelephoneNumber |
| InventoryItem | Number , ProductItemNumber ,Size,Color,Options,QuantityOnHand,AverageCost,ReorderQuantity |
| Order | Number , AccountNumber ,Date,PriorityCode,ShippingAndHandling,Tax,GrandTotal,DateReceived,ProcessorClerk,PhoneClerk,CallStartTime,LengthOfCall,EmailAddress,ReplyMethod |
| OrderItem | Number , OrderNumber , InventoryItemNumber , ShipmentNumber ,Quantity,Price,BackOrderStatus |
| OrderTransaction | Number , OrderNumber ,Date,TransactionType,Amount,PaymentMethod |
| Package | Number , CatalogNumber ,Description,SalePrice |
| ProductItem | Number , PackageNumber ,Vendor,Gender,Description,Season,NormalPrice,SpecialPrice |
| ReturnItem | Number , OrderNumber , InventoryItemNumber ,Quantity,Price,Reason,Condition,Disposal |
| Shipment | TrackingNumber , ShipperNumber ,DateSent,TimeSent,ShippingCost,DateArrived,TimeArrived |
| Shipper | Number ,Name,Address,ContactName,Telephone |

- 对于多对多关系要列出单独的表，表示多对多的关系

| | |
|------------------|---|
| Catalog | Number,Season,Year,Deccription,EffectiveDate,EndDate |
| CatalogPackage | CatalogNumber,PackageNumber |
| CatalogProduct | CatalogNumber,ProductNumber,Price |
| Customer | AccountNumber,Name,BillingAddress,ShippingAddress,DayTelephoneNumber,NightTelephoneNumber |
| InventoryItem | Number,ProductItemNumber,Size,Color,Options,QuantityOnHand,AverageCost,ReorderQuantity |
| Order | Number,AccountNumber,Date,PriorityCode,ShippingAndHandling,Tax,GrandTotal,DateReceivied,ProcessorClerk,PhoneClerk,CallStartTime,LengthOfCall,EmailAddress,ReplyMethod |
| OrderItem | Number,OrderNumber,InventoryItemNumber,ShipmentNumber,Quantity,Price,BackOrderStatus |
| OrderTransaction | Number,OrderNumber,Date,TransactionType,Amount,PaymentMethod |
| Package | Number,CatalogNumber,Description,SalePrice |
| PackageProduct | PackageNumber,ProductNumber |
| ProductItem | Number,PackageNumber,Vendor,Gender,Description,Season,NormalPrice,SpecialPrice |
| ReturnItem | Number,OrderNumber,InventoryItemNumber,Quantity,Price,Reason,Condition,Disposal |
| Shipment | TrackingNumber,ShipperNumber,DateSent,TimeSent,ShippingCost,DateArrived,TimeArrived |
| Shipper | Number,Name,Address,ContactName,Telephone |

➤ 评价模式质量

- 高质量的数据模型的特点
 - 表中每行以及主键都是唯一的
 - 冗余数据较少
 - 容易实现未来数据模型的改变
- 行和关键字的唯一性
- 数据模型的灵活性

➤ 数据库规范化

规范化：通过最小数据冗余来保证数据库模式质量的过程

- 第一范式 (1NF)：没有重复字段和字段组
- 第二范式 (2NF)：所有非主键字段都函数依赖于主键（去掉部分依赖）
- 第三范式 (3NF)：没有非主键字段函数依赖于其他非主键字段（去掉传递依赖）