中南大学
CENTRAL SOUTH UNIVERSITY

软件工程

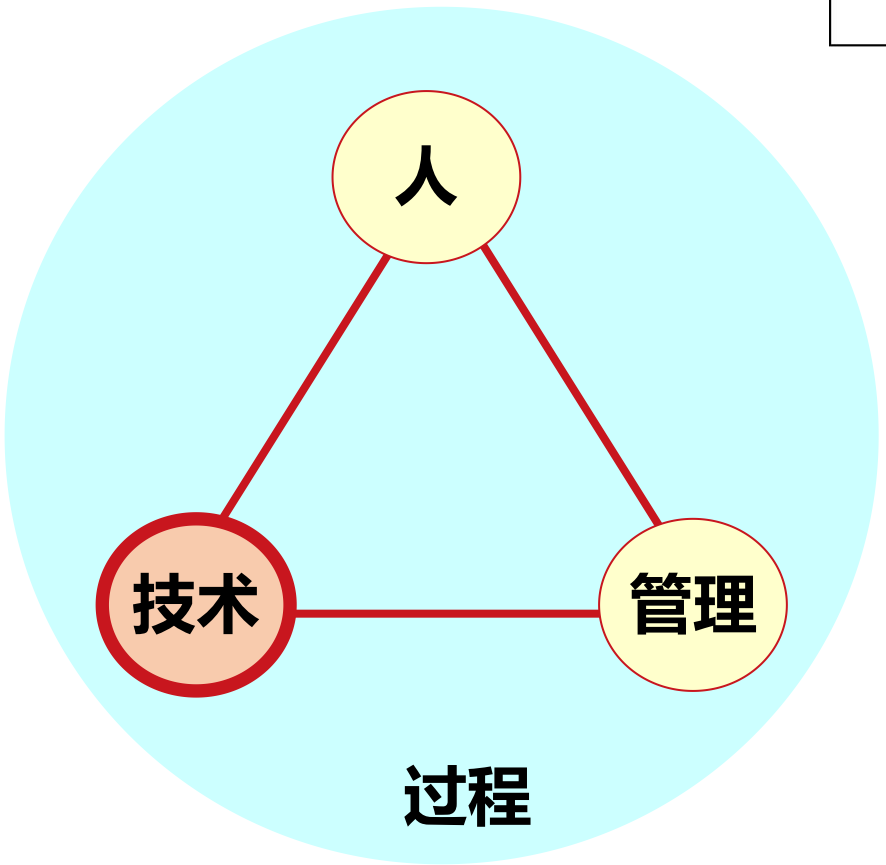
Software Engineering

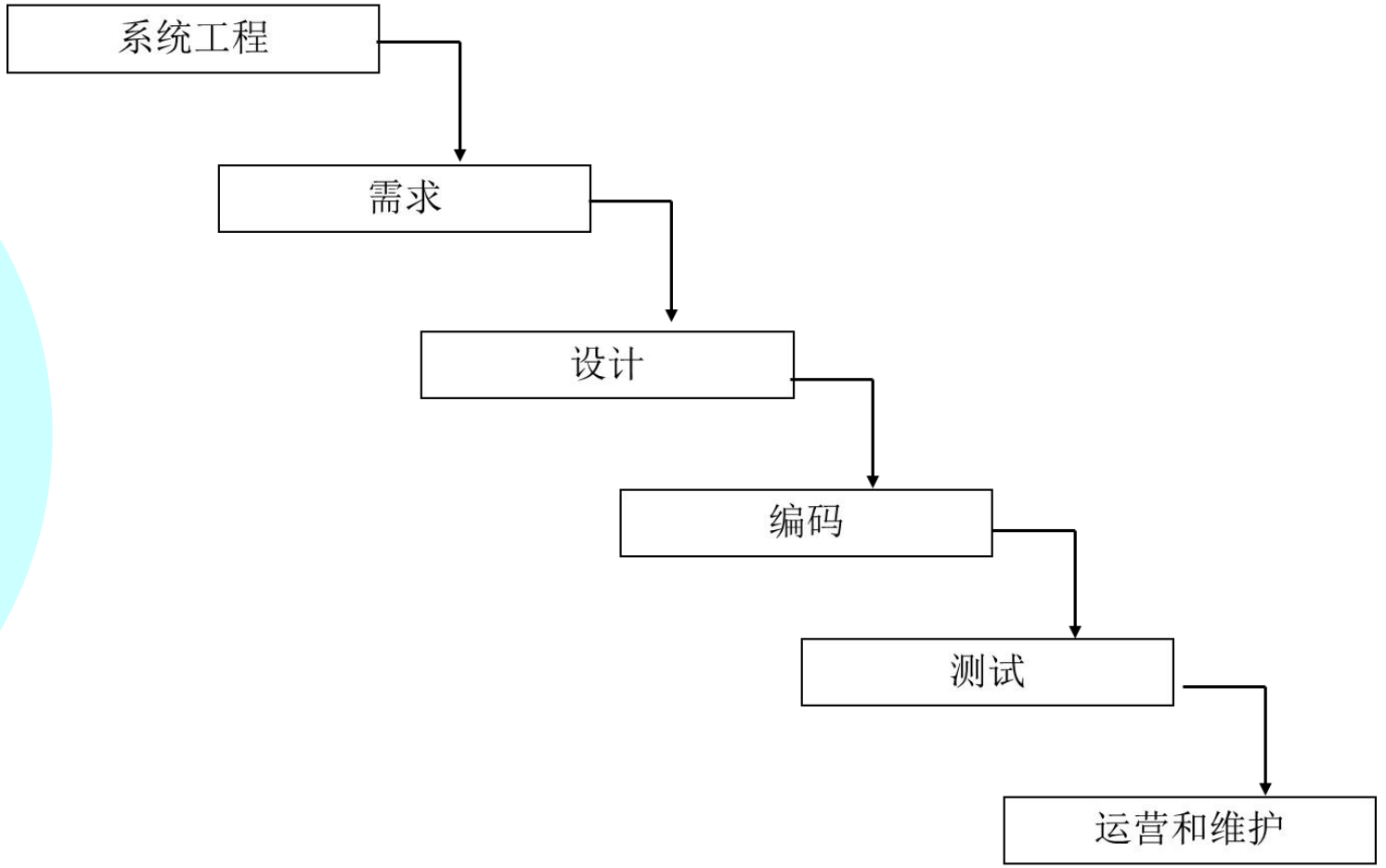
龙 军

jlong@csu.edu.cn 18673197878

计算机学院 | 大数据研究院

软件建模是软件工程的核心技术





第三章 软件工程模型和方法

大 纲

- 01-什么是模型
- 02-软件建模方法
- 03-面向对象方法概述

@第3章.教材

思考

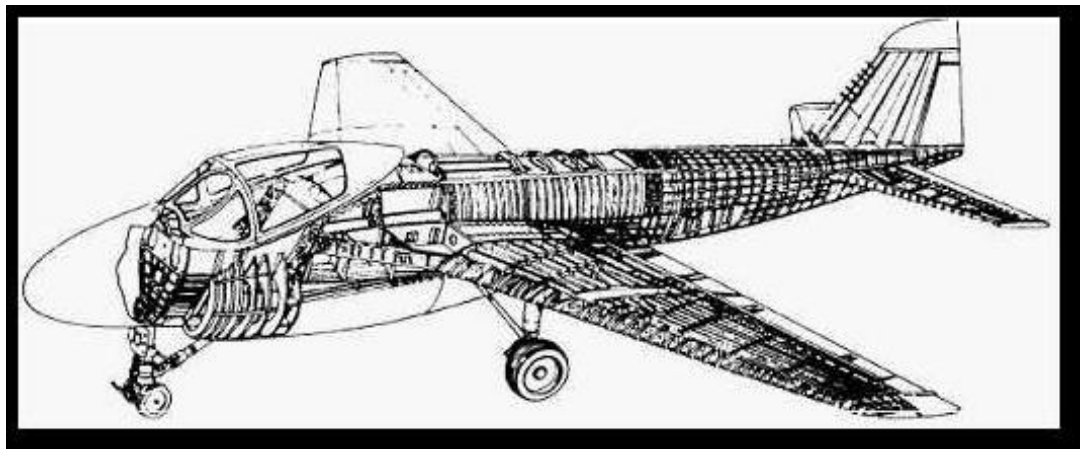
- 什么是模型?
- 为什么要建模?
- 软件模型有哪几种?

波音747

- 超过6百万个零件，仅机尾就有上百个零件
- 能承载上百个旅客
- 能不停地飞行上千公里

波音747

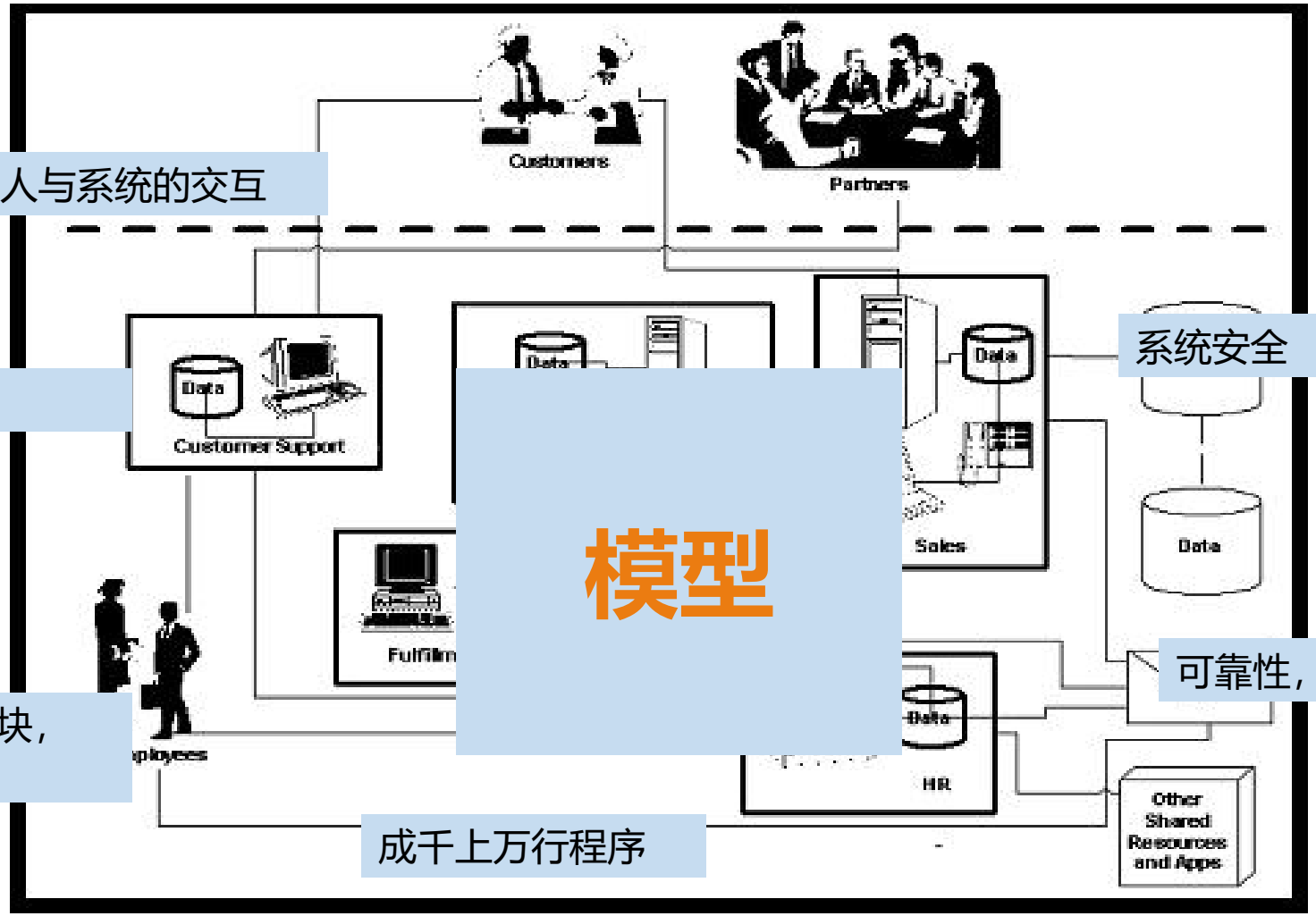
- 当波音在60年代开始生产747的时候，一共画了75，000张工程图



为什么?

因为他们正在生产一个极其复杂的系统，
而且不能出哪怕是一点点的差错

软件开发也同样复杂



人与系统的交互

交易，存储

子系统，模块，
分布式

模型

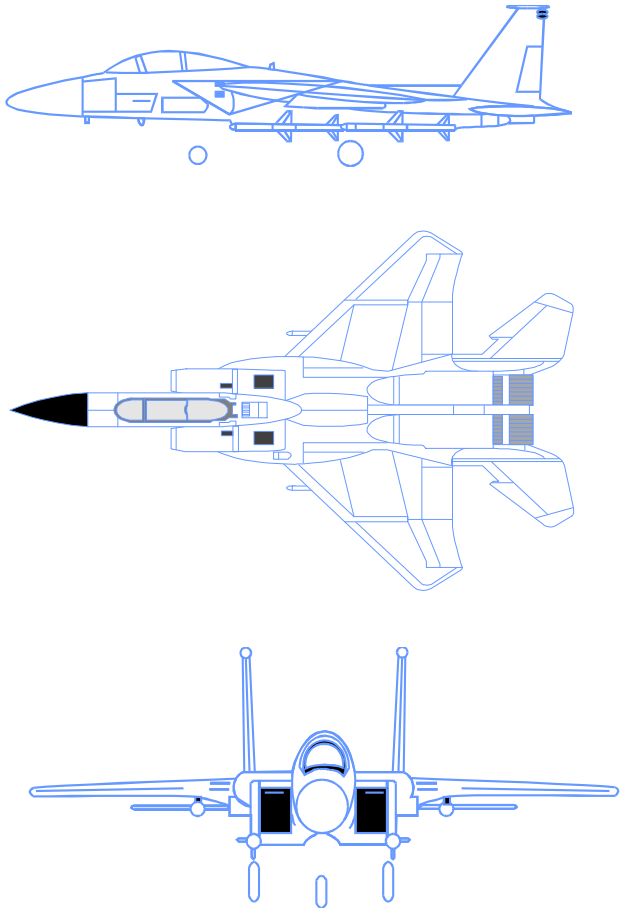
系统安全

可靠性，可扩展性

成千上万行程序

模型?

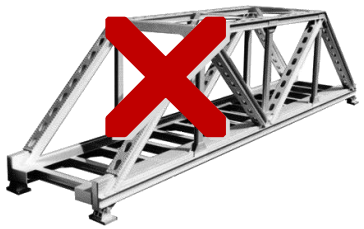
- A model is a simplification of reality. 模型是对现实的简化



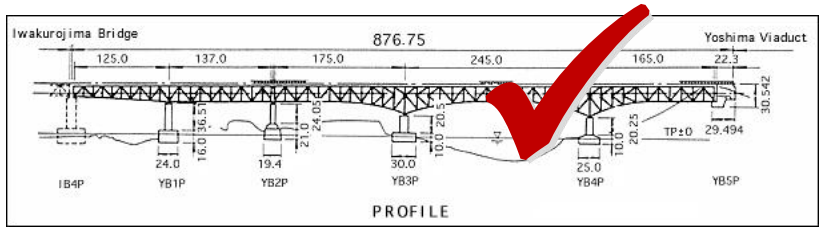
工程师们这样做...

- 他们在建造实际的物体之前...

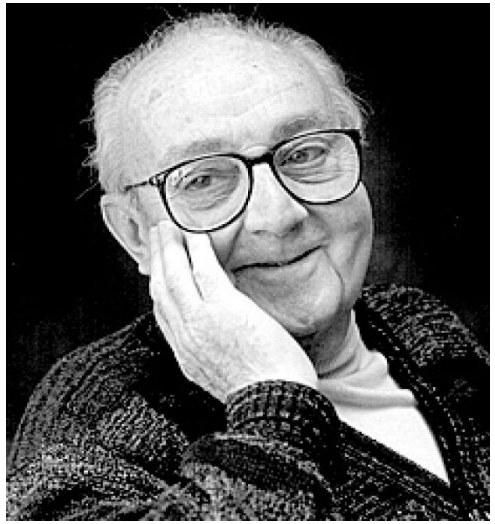
...首先建立模型



...然后在模型的基础上进行研究和改进



模型特性



George Box

所有模型都是错误的；
一些模型又是有用的。

※ 为什么错误？

- 模型忽略了实体的某些信息

※ 为什么有用？

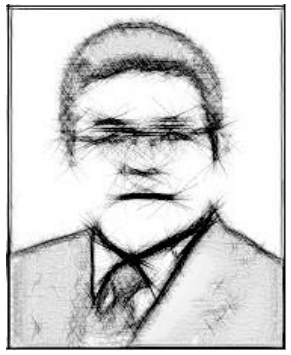
※ 模型突出了实体的某些信息

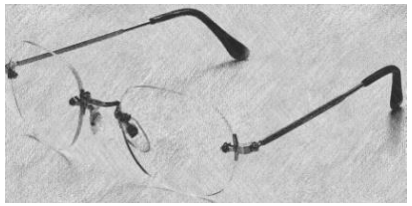
George Box（1919--2013）是一位著名的统计学家，对质量控制、时序分析、实验设计等的研究尤为突出 。
个人信息网页：http://en.wikipedia.org/wiki/George_E._P._Box

模型特性

※ 模型 (Model)

- 客观信息体在某个特定视角上的抽象视图
- 忽略次要信息、突出主要信息





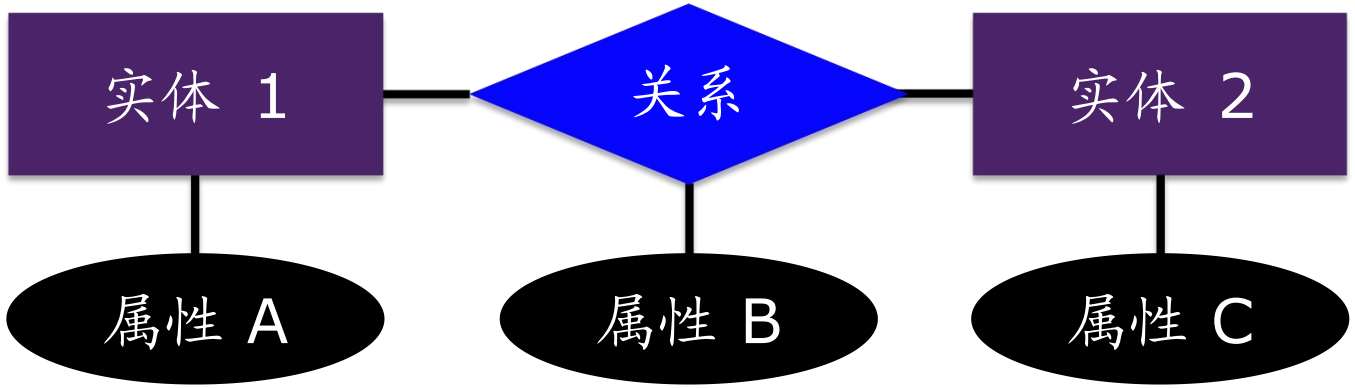
模型三元素

※ 模型的三大组成元素

- 实体 (Entity)
- 关系 (Relationship)
- 属性 (Attribute)

※ ER建模(ERA建模)

- 由陈品山于1976年提出，是通用建模的一般方法



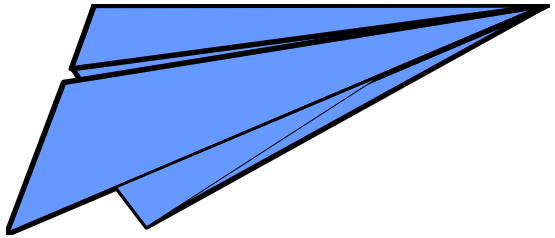
模型功能

- 在正式启动工程项目之前发现设计中的错误和遗漏之处
  - 通过(形式化的)分析和实验，降低工程的风险
- 研究和比较不同的解决方案
- 用来和项目的所有者进行交流
  - 客户、用户、实现者、测试者、开发文档管理员，等等.
- 促进工程的实现

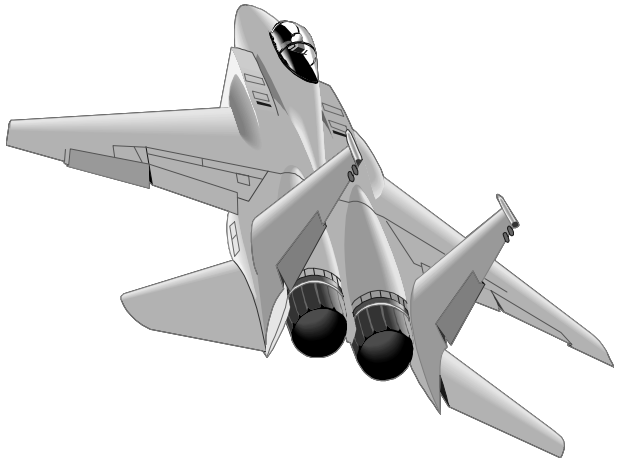
建模的重要性

Less Important

More Important



Paper Airplane



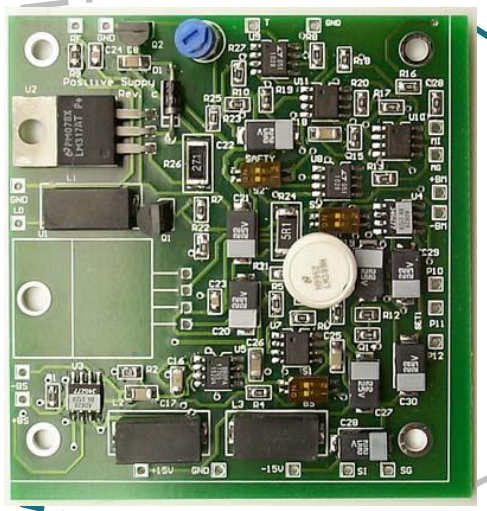
Fighter Jet

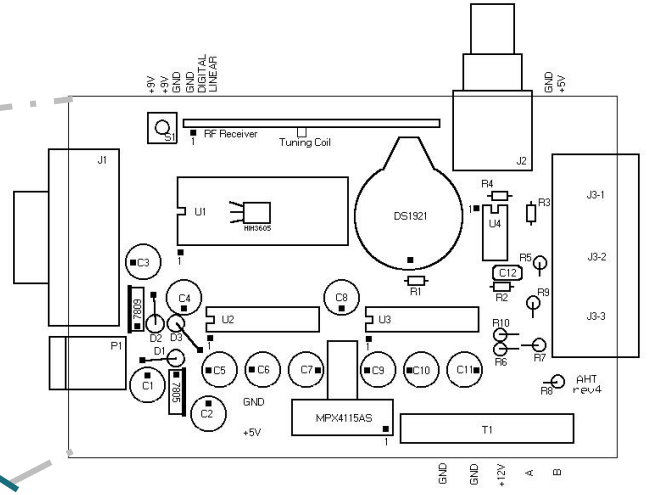
有用模型的特征

- 抽象性
  - 突出重点方面，去除无关紧要的细节
- 可理解性
  - 模型的表达方式能被模型的观察者很容易地理解
- 精确性
  - 忠实地表达被建模的系统
- 说明性
  - 能够被用来对被建模系统进行直观地分析，并得出正确的结论
- 经济性
  - 模型的建立和分析比被建模系统更廉价，更经济

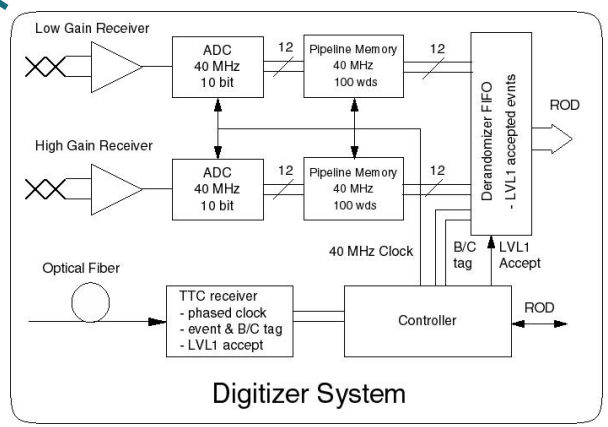
作为有用的工程模型，必需具备以上所有特征!

模型的多个视图





物理布局视图



功能视图

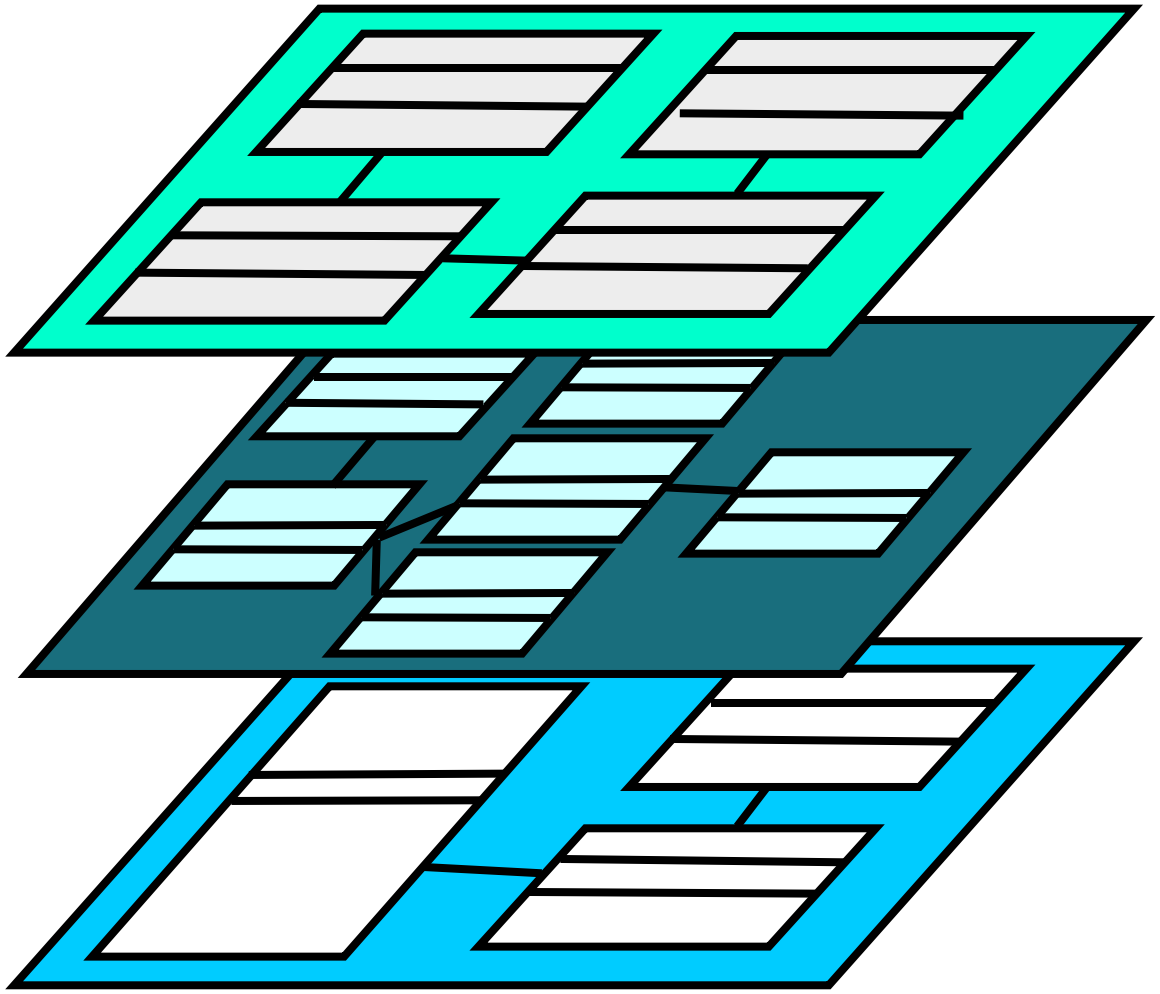
举例：UML 的软件模型视图

- 需求:
  - 用例图 Use Case diagram
- 结构:
  - 本体论：类图 Class diagram
  - 实例: 对象图 Object diagram
- 行为
  - 状态图 Statechart diagram
  - 活动图 Activity diagram
  - 交互：顺序图和协作图　Sequence diagram & Collaboration diagram
- 实现:
  - 构件图 Component diagram
  - 部署图 Deployment diagram

这些视图相互集成

构成一个完整的模型

软件的三种模型



Computation-Independent Model (CIM)

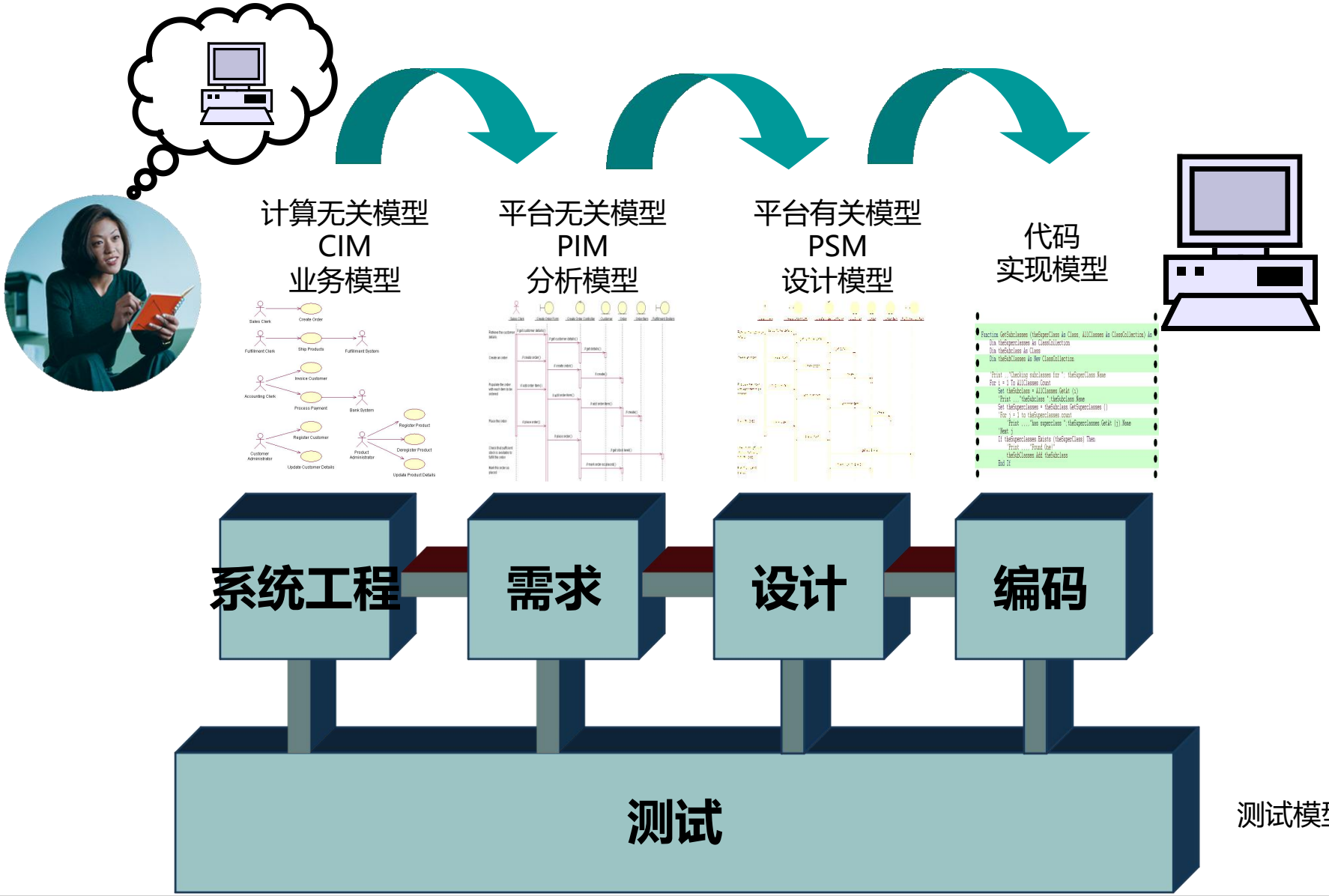
Platform-Independent Model (PIM)

Platform-Specific Model (PSM)

More Abstract

More Specific

模型间的转换


计算无关模型
CIM
业务模型
平台无关模型
PIM
分析模型
平台有关模型
PSM
设计模型
代码
实现模型
系统工程
需求
设计
编码
测试
测试模型

第三章 软件工程模型和方法

大 纲

01-什么是模型

02-软件建模方法

03-面向对象方法概述

@第3章.教材

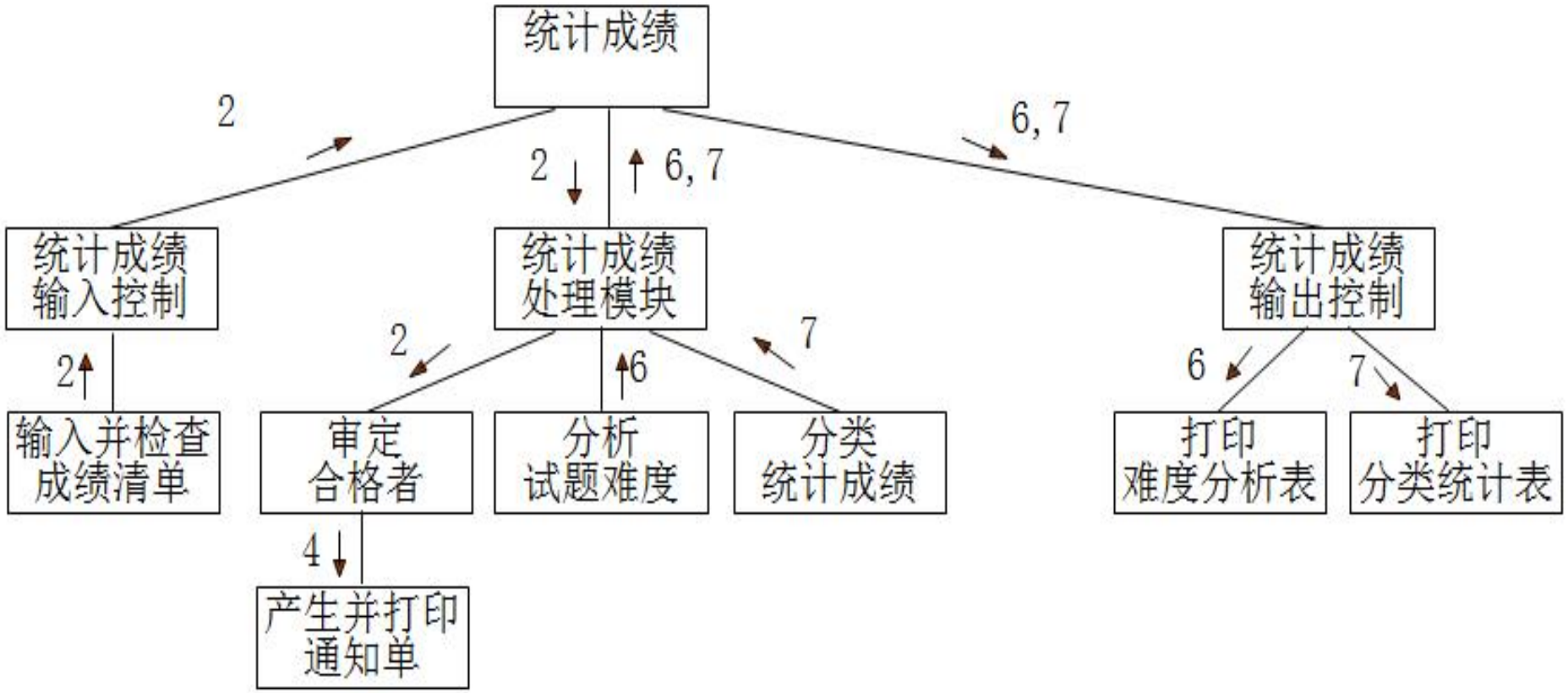
软件建模的方法

- 结构化方法 (Structured Method)
- 面向对象方法 (Object Oriented Method)
- 基于构件的软件开发方法 (Component Based Software Development)
- 面向服务方法 (Service Oriented Method)
- 模型驱动的开发方法 (Model-Driven Development)
- 形式化方法 (Formal Method)
- ……

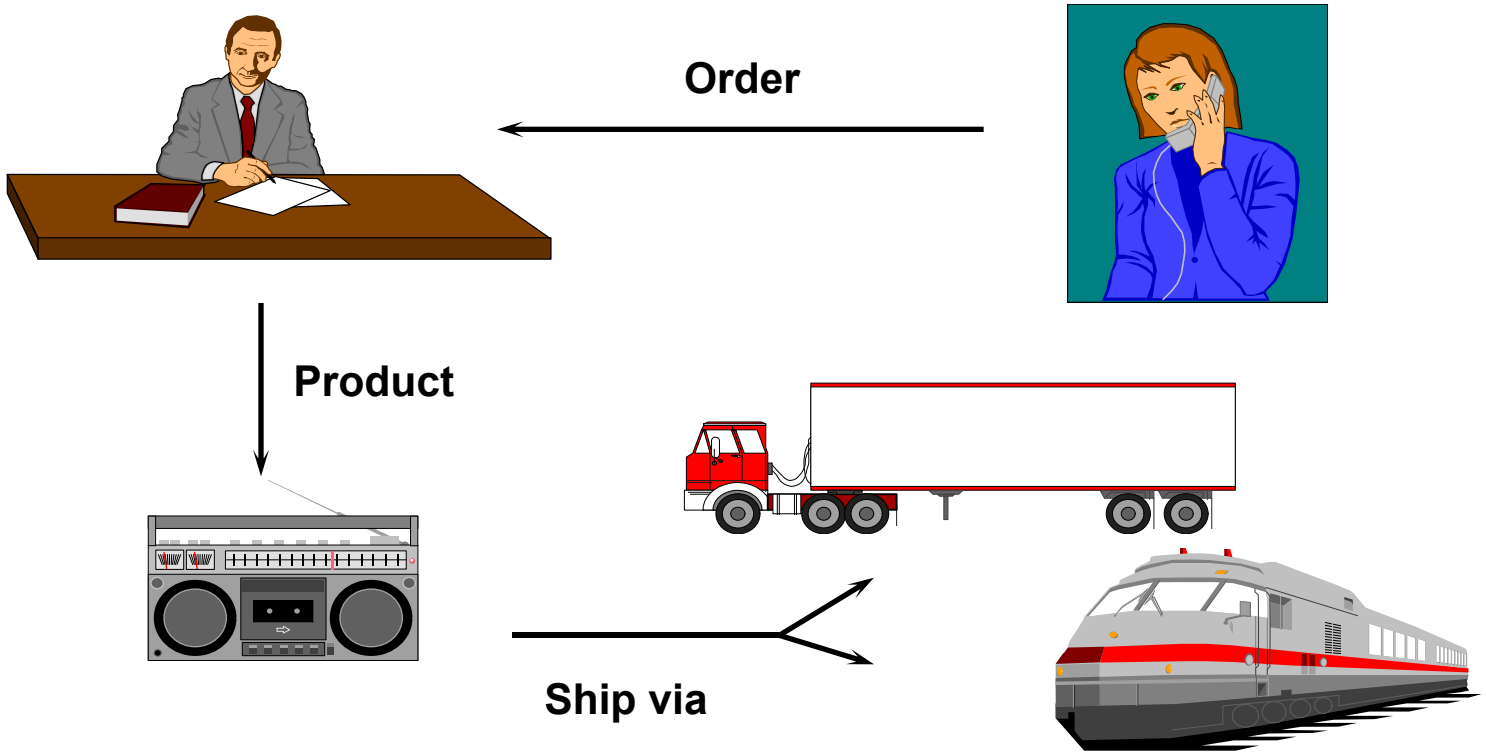
结构化方法

- 核心: 数据和处理
- 手段: 自顶向下，逐步求精、模块化
- 常用建模工具:
  - 需求建模:
    - DFD(数据流图) 、 DD(数据字典)、ERD(实体关系图)、STD(状态图)
  - 设计建模:
    - 结构图（SC）
    - 流程图、 N-S图、PAD图、伪代码

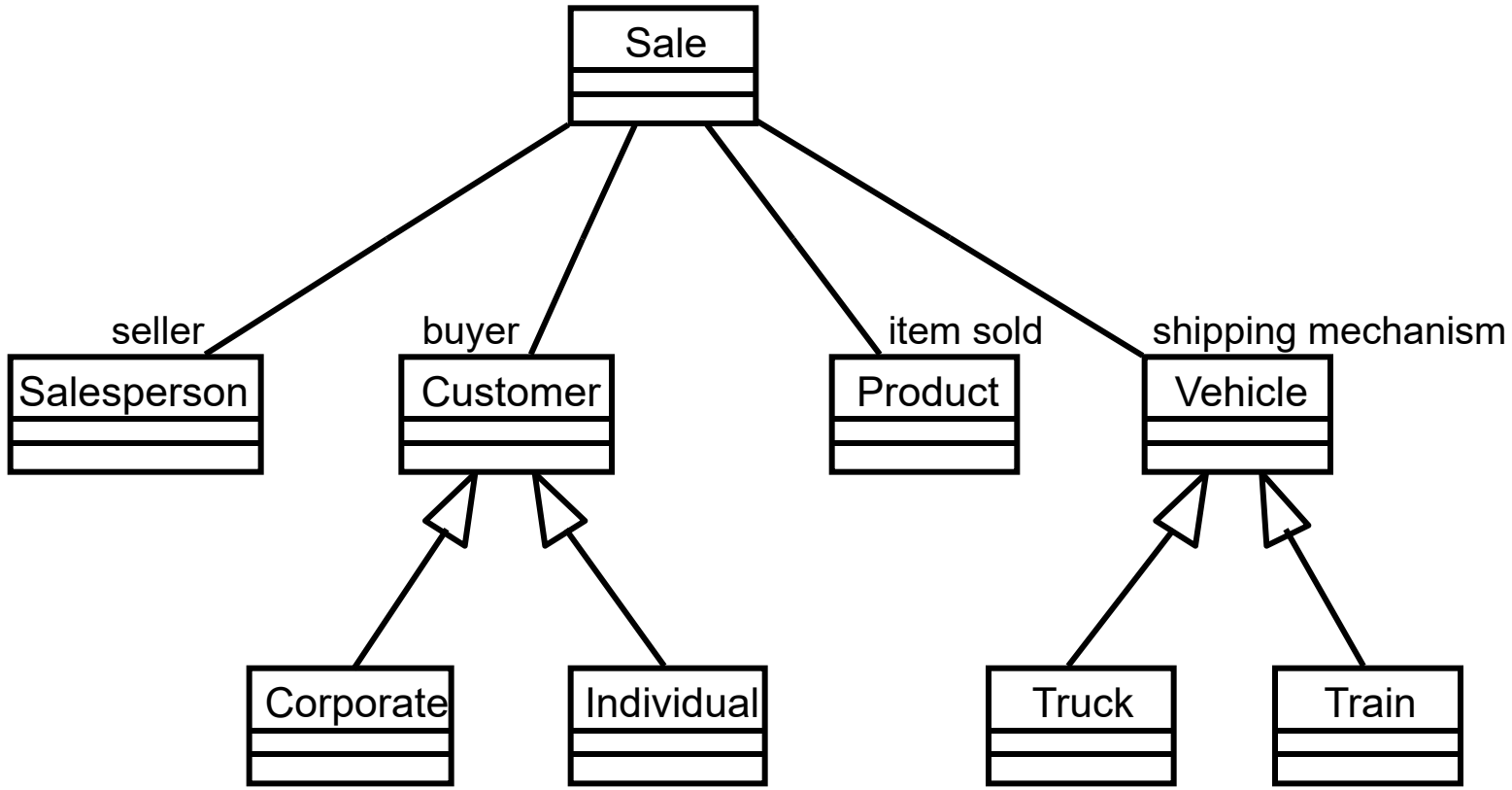
结构化设计的系统示例


统计成绩
2
2
6, 7
6, 7
统计成绩
输入控制
统计成绩
处理模块
统计成绩
输出控制
2
2
6
7
6
7
输入并检查
成绩清单
审定
合格者
分析
试题难度
分类
统计成绩
打印
难度分析表
打印
分类统计表
4
产生并打印
通知单

面向对象方法示例：销售订单

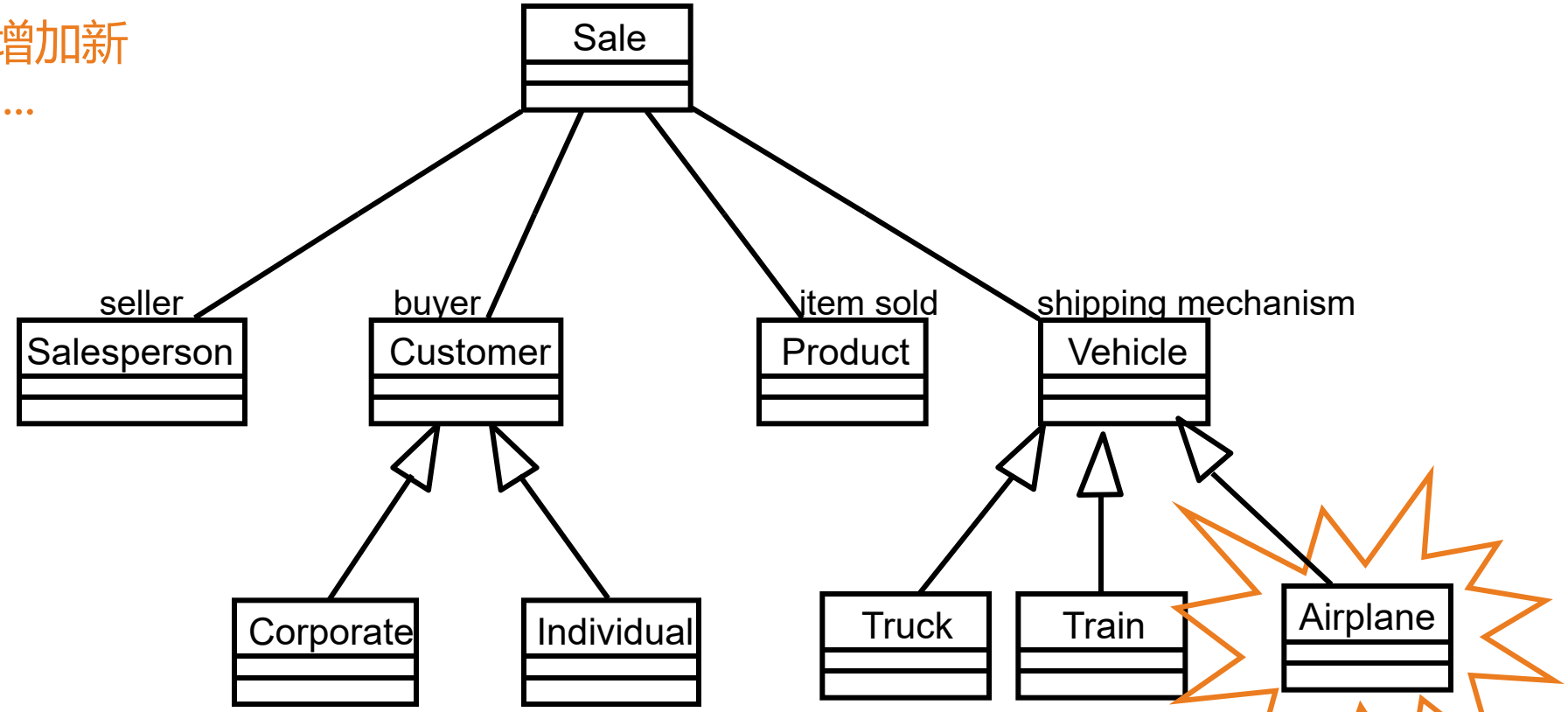


销售订单的类图


Sale
seller
buyer
item sold
shipping mechanism
Salesperson
Customer
Product
Vehicle
Corporate
Individual
Truck
Train

需求变更的影响

假定你需要增加新的运输工具 ...

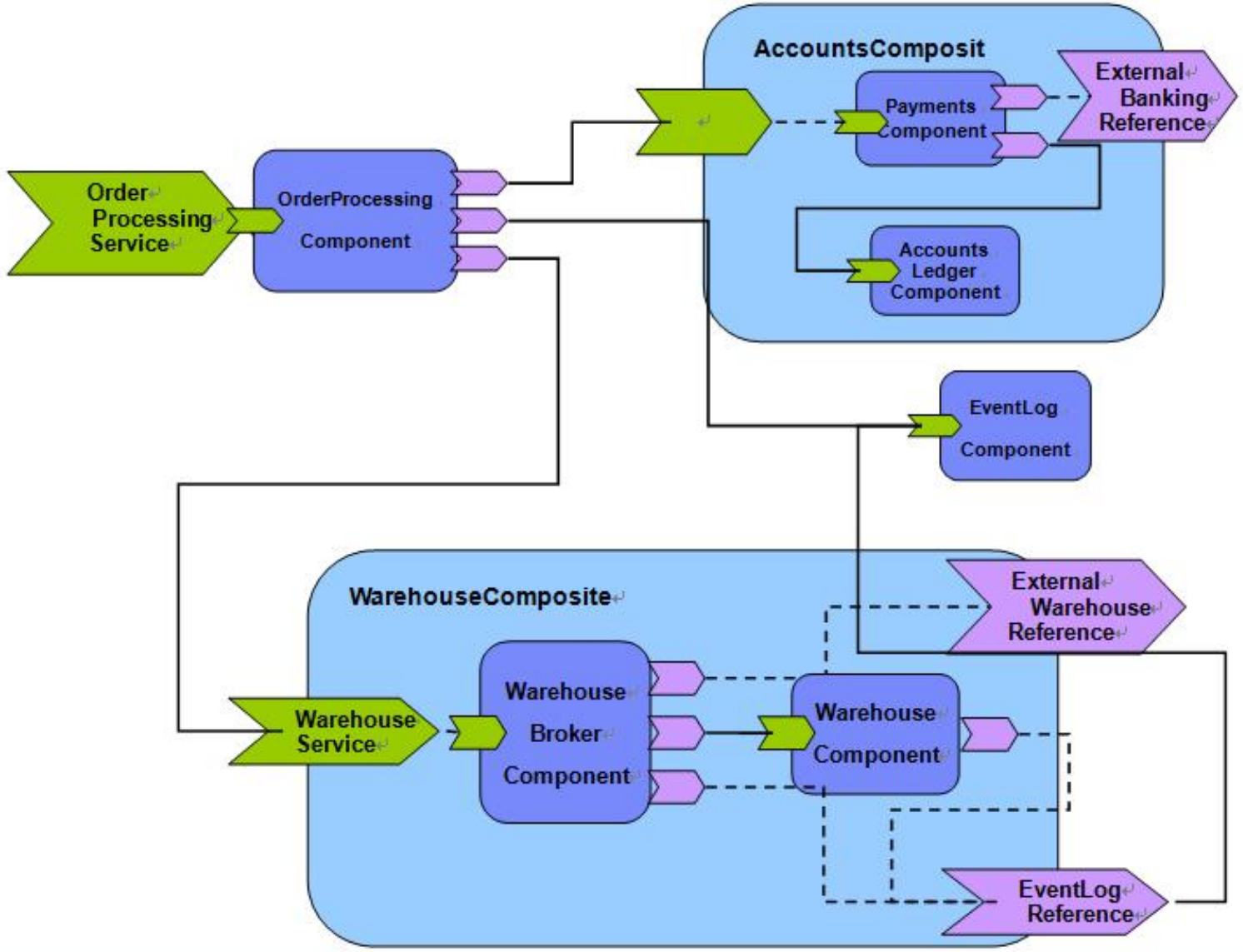


增加一个新的子类

面向对象方法

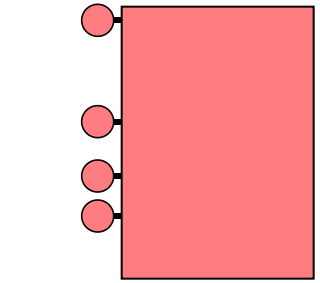
- 九十年代以来的主流开发方法
  - 符合人们对客观世界的认识规律
  - 开发的系统结构易于理解、易于维护
  - 继承机制有力支持软件复用
- 常见的面向对象方法
  - Booch method 1994
  - Coad and Yourdon method 1991
  - Rambaugh method -- OMT 1991
  - Jacobson method – OOSE 1992
  - Wirfs-Brock method 1990
  - ……
  - 国际标准统一建模语言 UML 1997

基于构件的软件系统示例

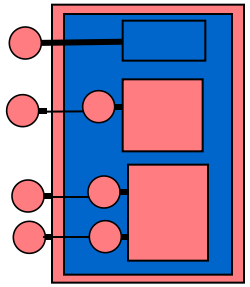


构件

- 构件是软件复用的重要手段，是核心和基础
- 构件由构件规约与构件实现两部分组成

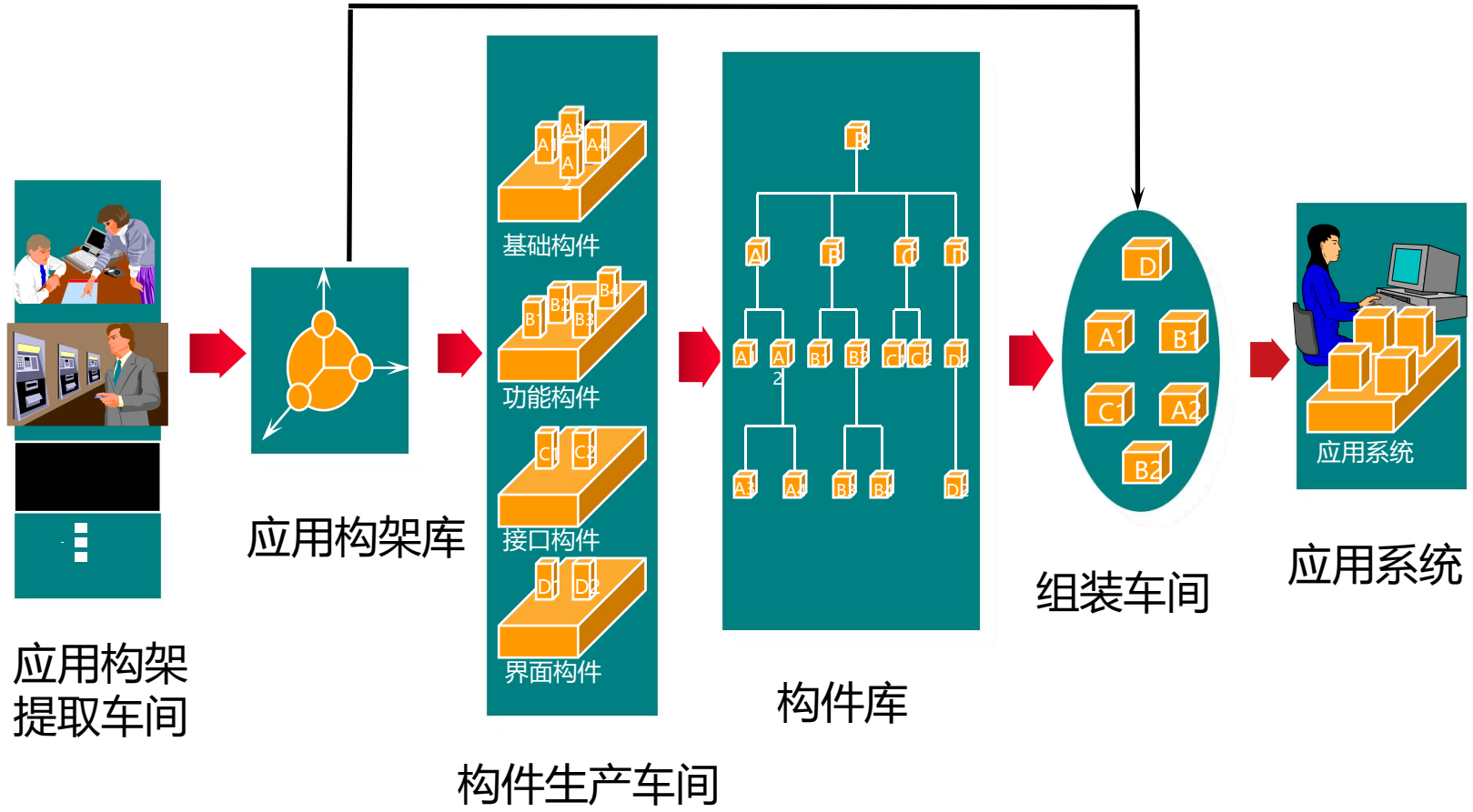


Component interface



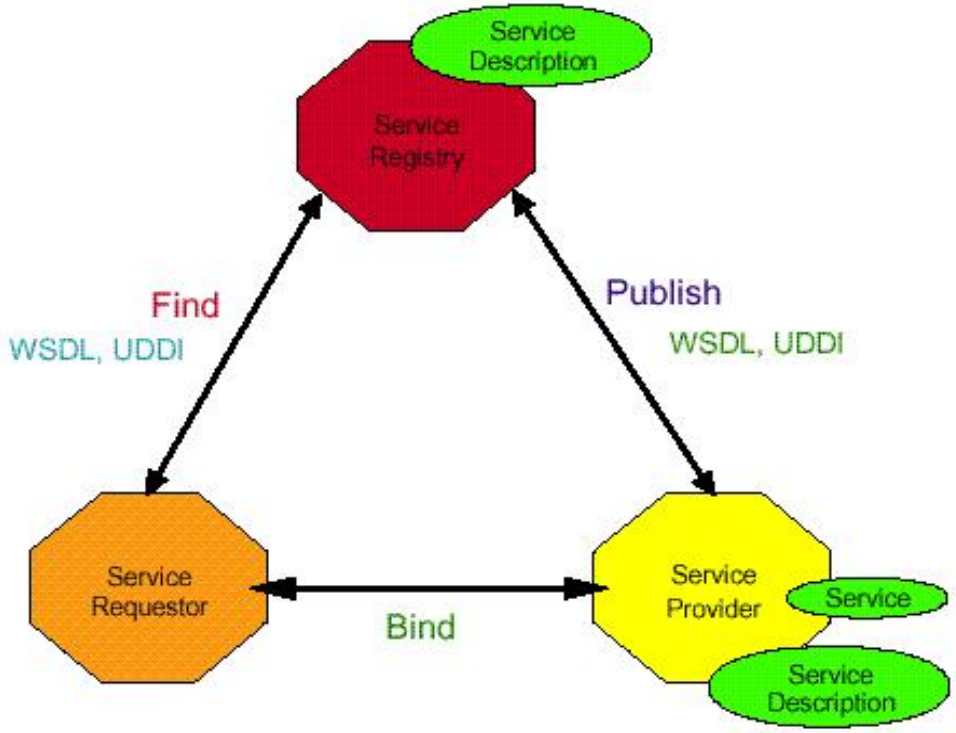
Component implementation

基于构件的开发



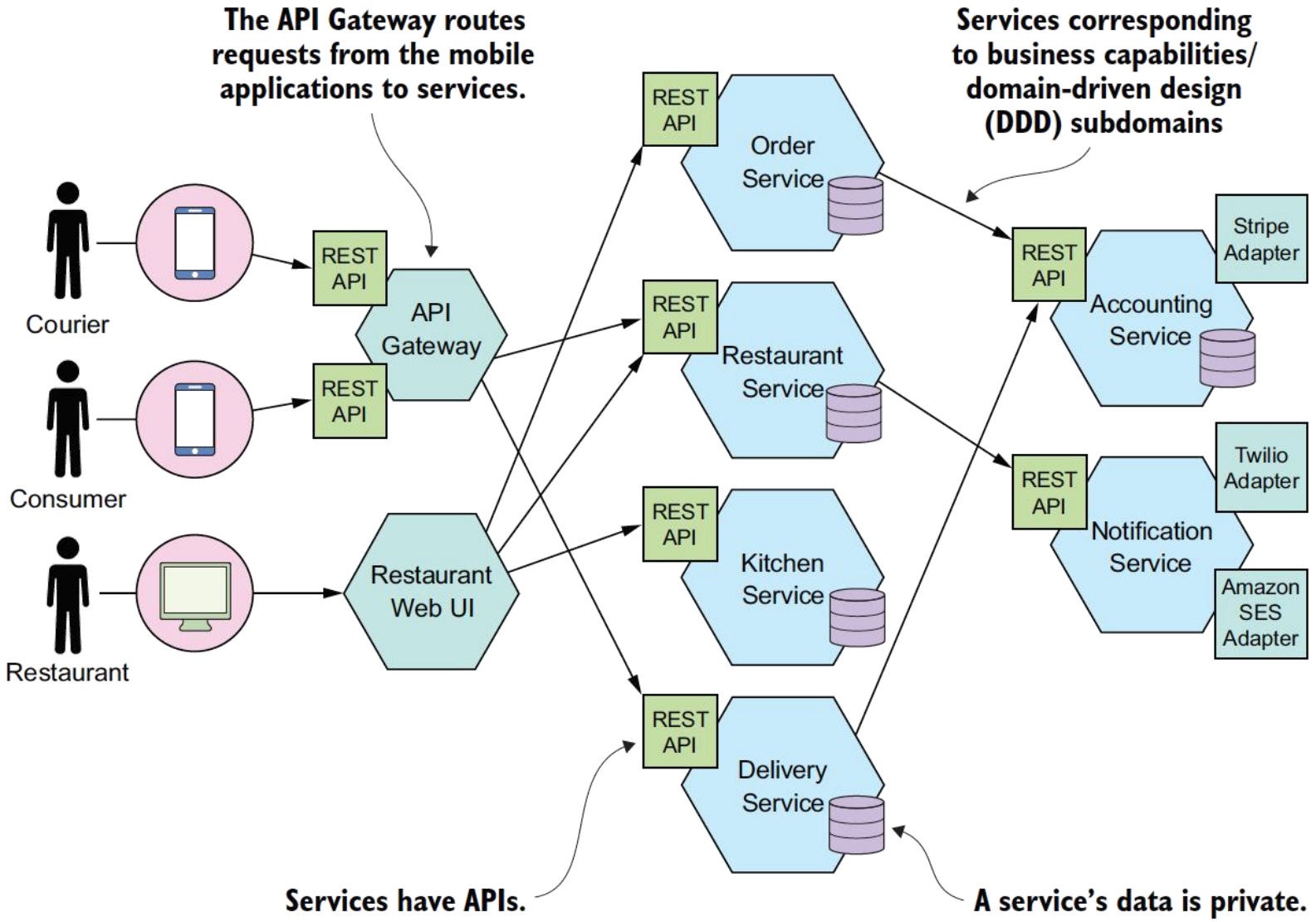
标准规范　　与　　质量保证

面向服务的方法



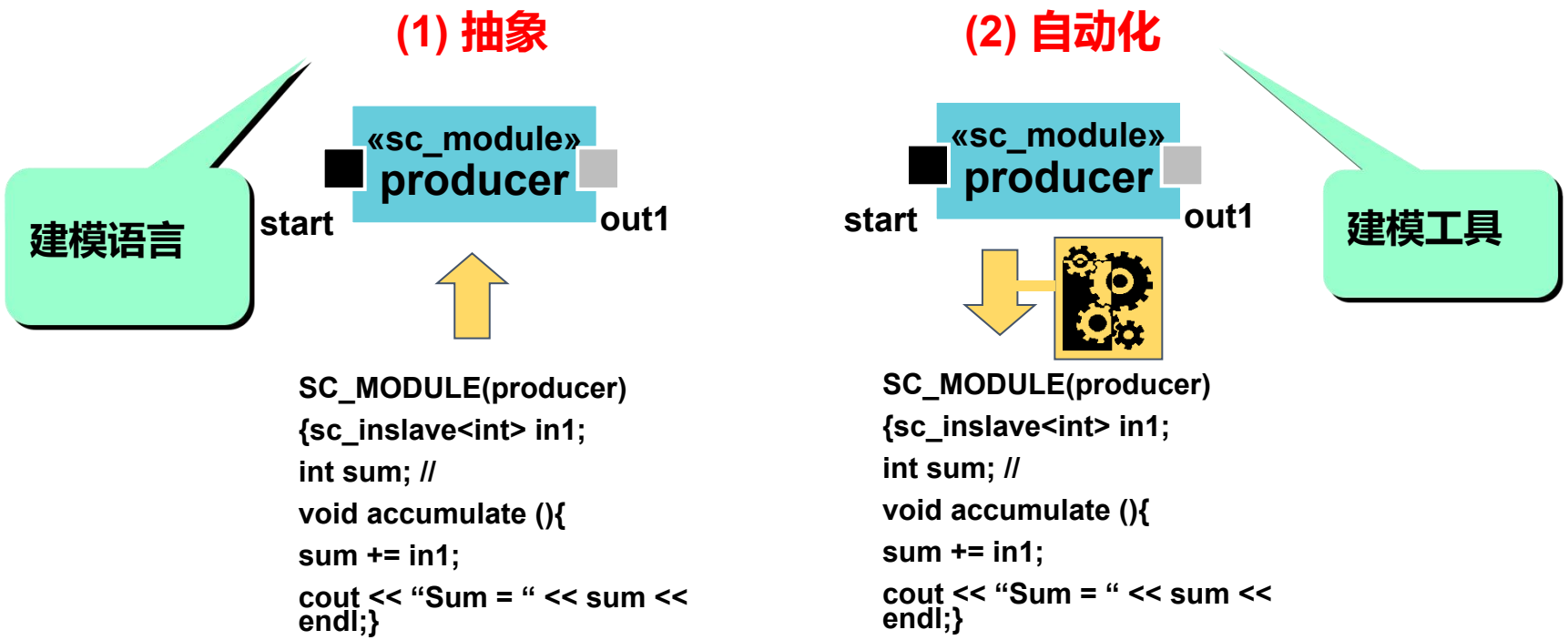
- 服务的抽象性（基于接口的编程）
- 服务的自治性（实现分布式应用）
- 服务间的松耦合式绑定，基于标准化消息进行通信
- 服务的自描述性（支持动态发现与延迟绑定）
- 服务的粗粒度（支持基于业务逻辑的积木式装配）

面向服务的系统示例（基于微服务架构的网络订餐系统）



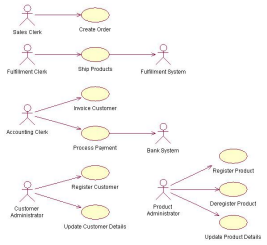
模型驱动的软件开发 (MDD)

- 以模型为中心（相对于以代码为中心）的软件开发方法
- 基于以下两种久经考验的技术

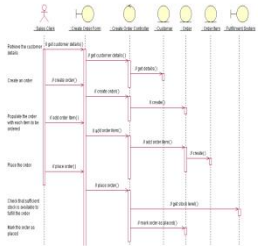


模型间的(半)自动转换

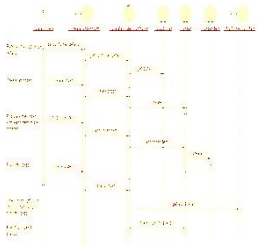
计算无关模型
CIM
业务模型



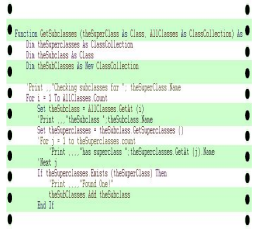
平台无关模型
PIM
分析模型

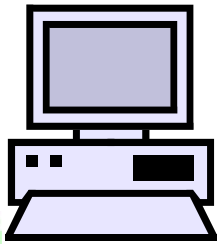


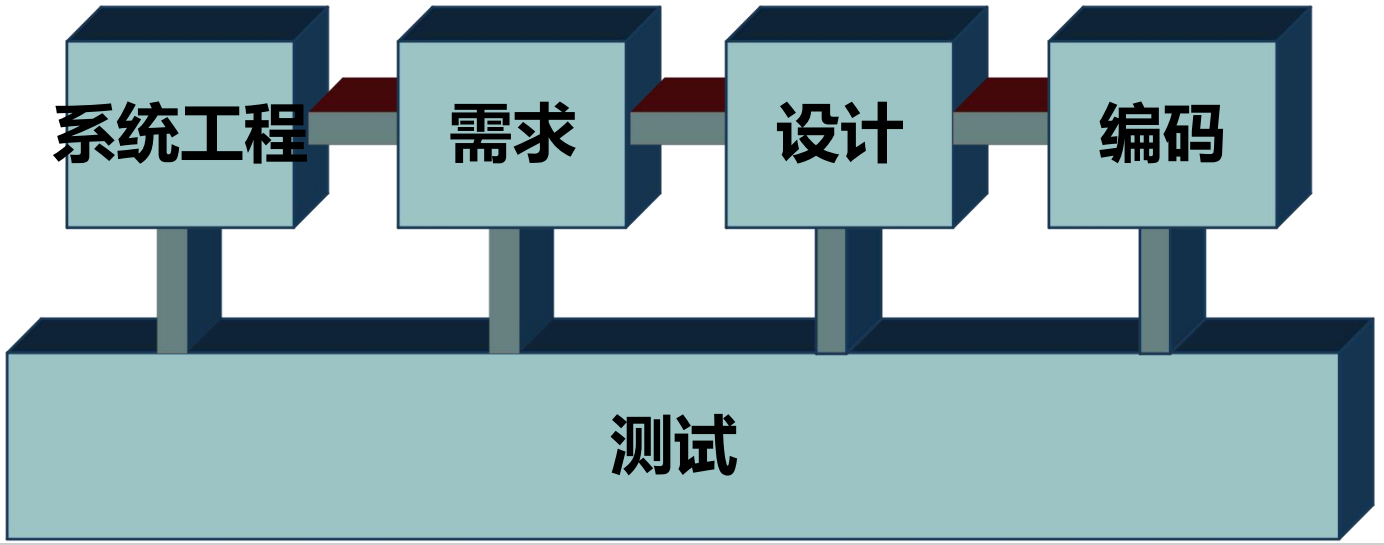
平台有关模型
PSM
设计模型



代码
实现模型







测试模型

形式化方法

- 形式化方法是基于数学的技术开发软件，如集合论、模糊逻辑、函数。
- 形式化方法的好处：
  - 无二义性
  - 一致性
  - 正确性
  - 完整性

形式化验证
形式化开发

国产替代
安全可信

举例

--------AddBlock----------------

△BlockHandler

Ablocks？： P BLOCKS

Ablocks？⊆ used

BlockQueue'=BlockQueue⌢ 〈Ablocks？〉

used'=used∧

free'=free

加一个块集合到队列的尾部, 采用Z语言

形式化方法的不足

- 形式化规约主要关注于功能和数据，而问题的时序、控制和行为等方面却更难于表示。此外，有些问题元素(如，人机界面)最好用图形技术来刻划。
- 使用形式化方法来建立规约比其他方法更难于学习，并且对某些软件实践者来说它代表了一种重要的“文化冲击”。
- 难以支持大的复杂系统。

尚未成为主流的开发方法，实践和应用较少

Review：软件建模的方法

- 结构化方法 (Structured Method)
- 面向对象方法 (Object Oriented Method)
- 基于构件的软件开发方法 (Component Based Software Development)
- 面向服务方法 (Service Oriented Method)
- 模型驱动的开发方法 (Model-Driven Development)
- 形式化方法 (Formal Method)
- ……

第三章 软件工程模型和方法

大 纲

01-什么是模型

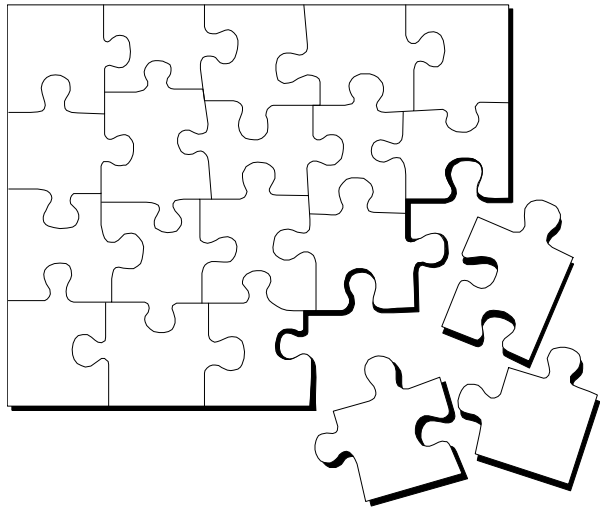
02-软件建模方法

03-面向对象方法概述

@第3章.教材

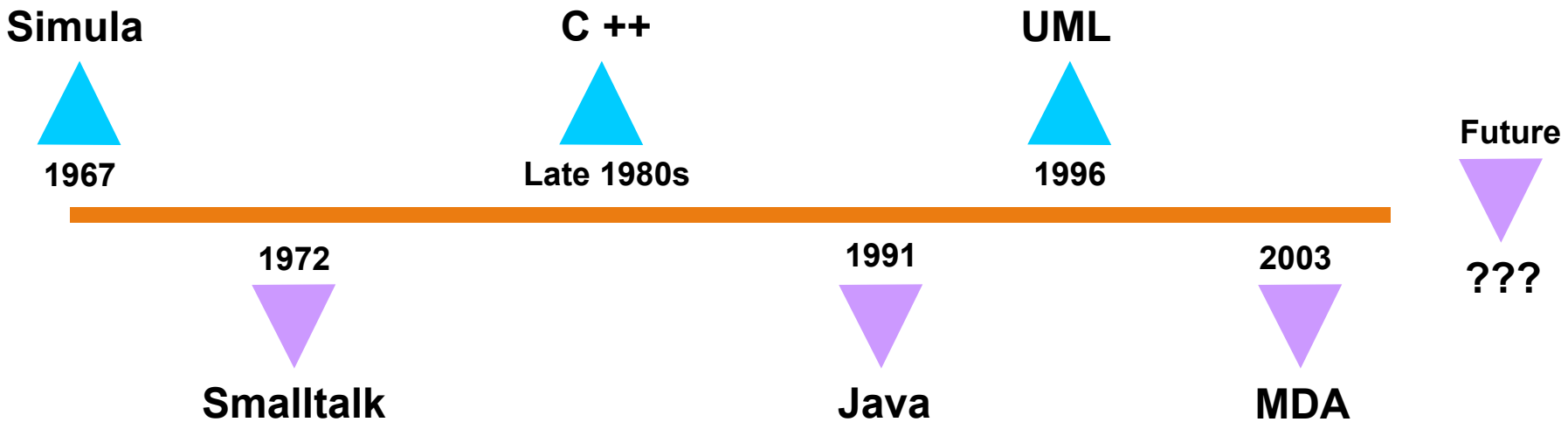
Object Technology

- What Is Object Technology?
  - A set of principles (abstraction, encapsulation, polymorphism) guiding software construction, together with languages, databases, and other tools that support those principles.
- The Strengths of Object Technology
  - Reflects a single paradigm
  - Facilitates architectural and code reuse
  - Reflects real world models more closely
  - Encourages stability
  - Is adaptive to change

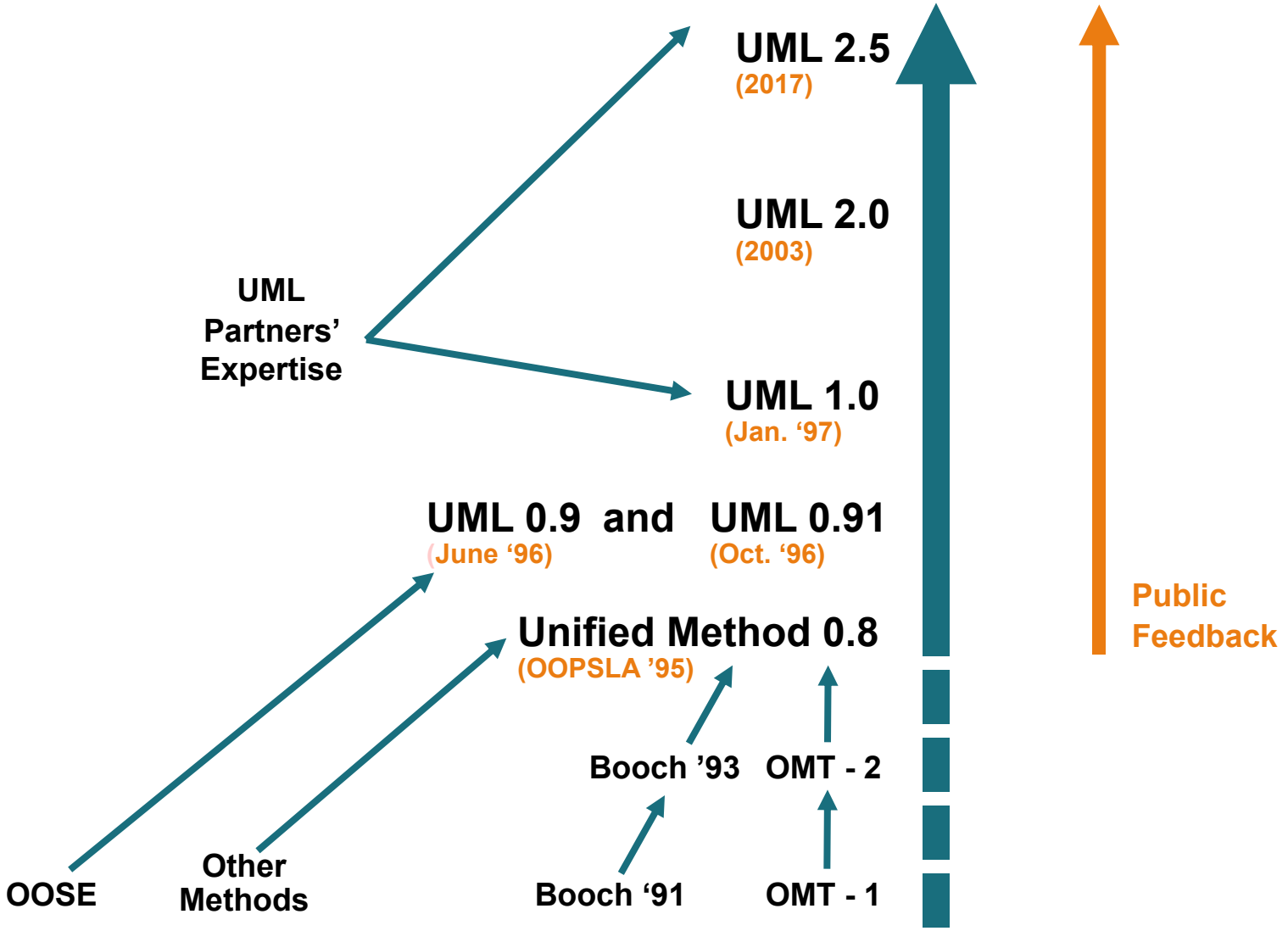


The History of Object Technology

- Major object technology milestones

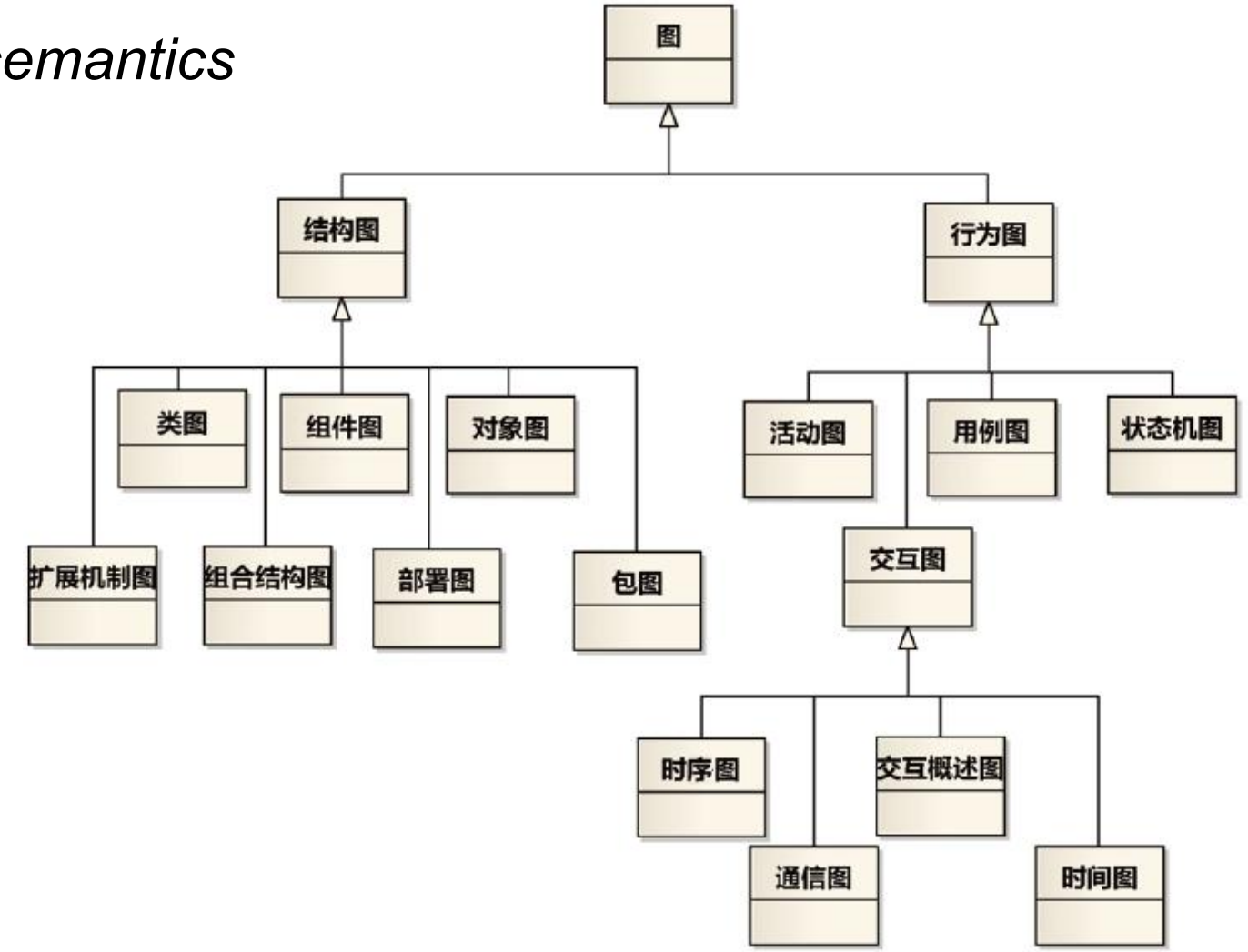


UML (Unified Modeling Language)



UML模型

- *Multiple views*
- *Precise syntax and semantics*

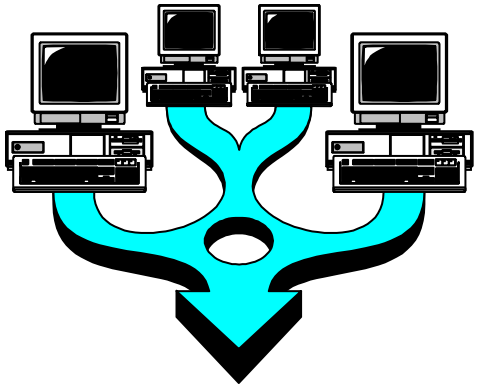


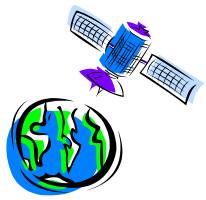
UML建模工具

- IBM Rhapsody 和RSA
- Sparx Systems EA
- Sybase PowerDesigner
- Borland Together
- Microsoft Visio
- StarUML 开源
- ArgoUML 开源
- …

Where Is Object Technology Used?

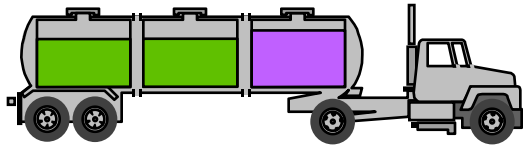
- Client/Server Systems and Web Development
- Real-time Systems
- Embedded System
- Multimedia System
- Middleware
- ……





What Is an Object?

- Informally, an object represents an entity, either physical, conceptual, or software.
  - Physical entity

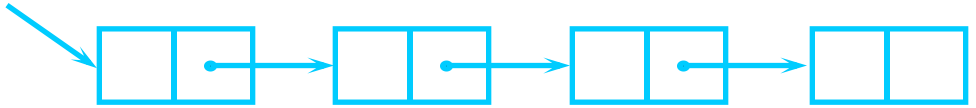


Truck

  - Conceptual entity

Chemical Process

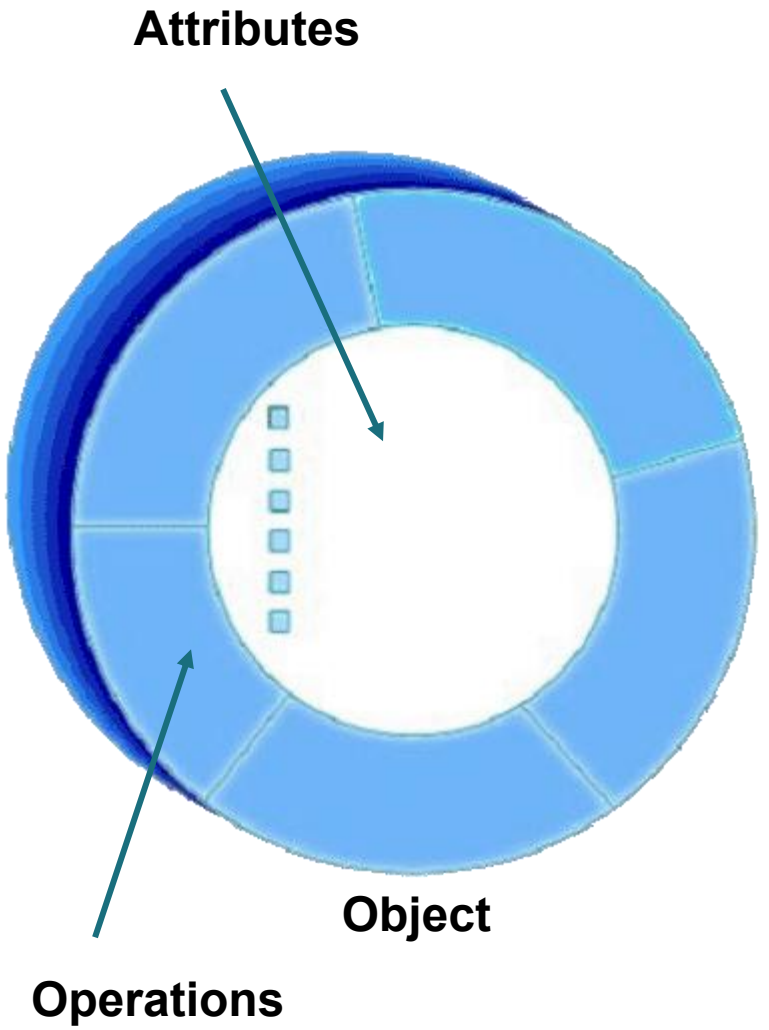
  - Software entity



Linked List

A More Formal Definition

- An object is an entity with a well-defined boundary and identity that encapsulates state and behavior.
  - State is represented by attributes and relationships.
  - Behavior is represented by operations, methods, and state machines.



Representing Objects in the UML

- An object is represented as a rectangle with an underlined name.

Professor J Clark

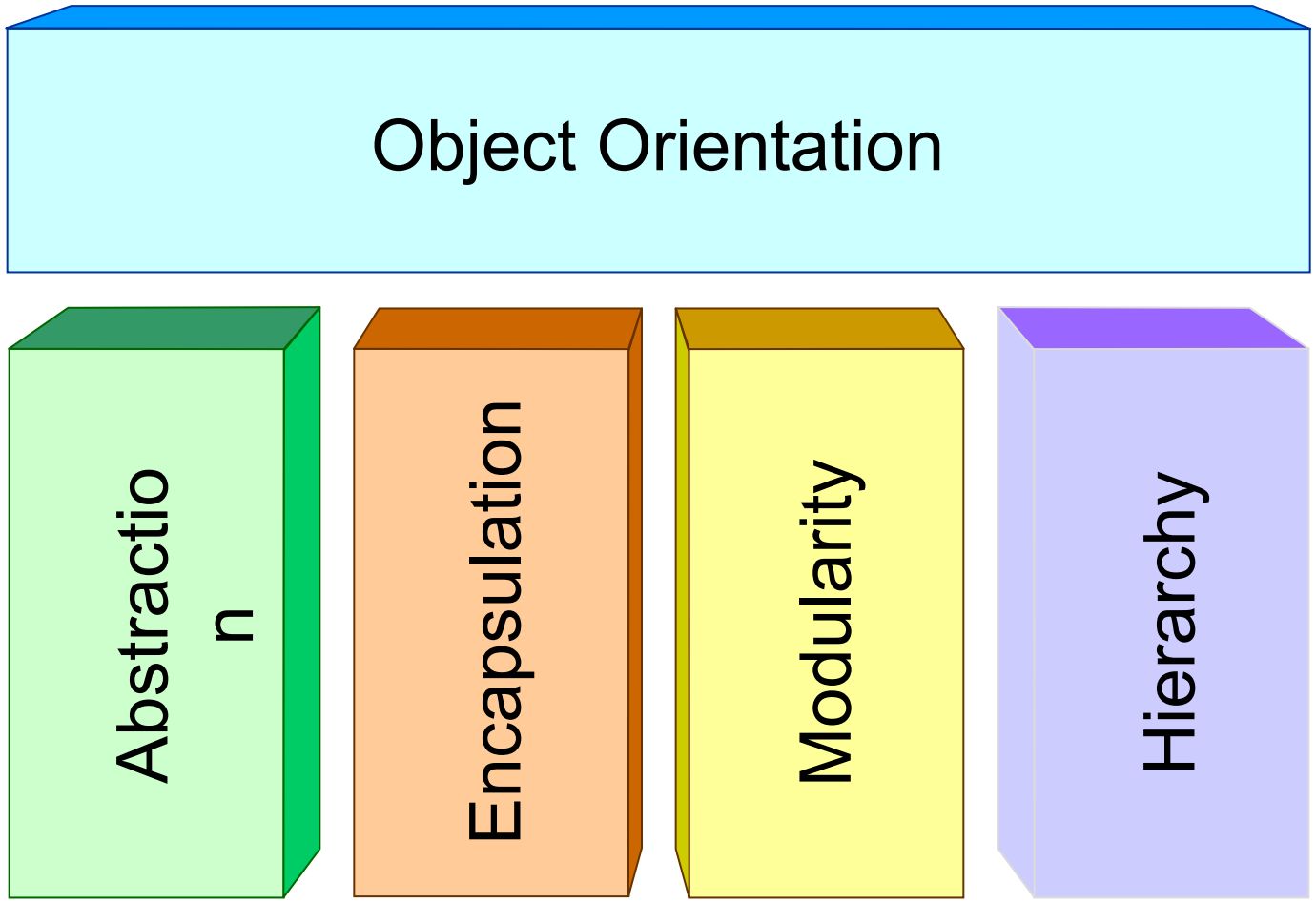
J Clark :
Professor

Named Object

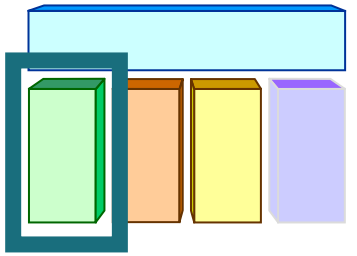
: Professor

Anonymous Object

Basic Principles of Object Orientation

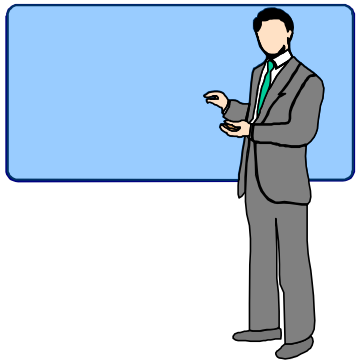


What Is Abstraction?

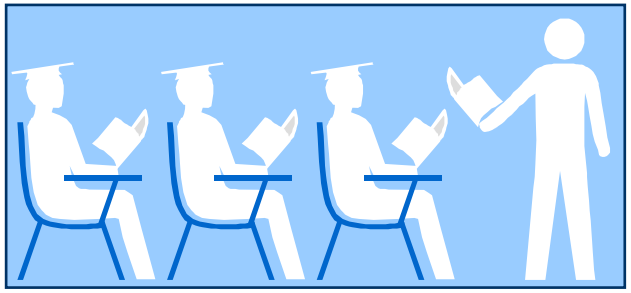


- The essential characteristics of an entity that distinguishes it from all other kinds of entities.
- Defines a boundary relative to the perspective of the viewer.
- Is not a concrete manifestation, denotes the ideal essence of something.

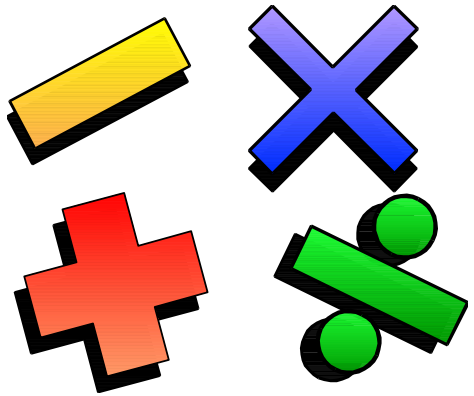
Student



Professor

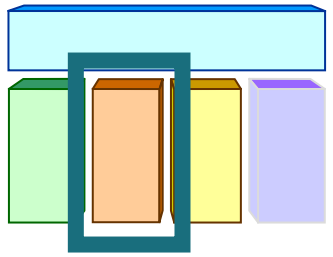


Course Offering (9:00 a.m., Monday-Wednesday-Friday)

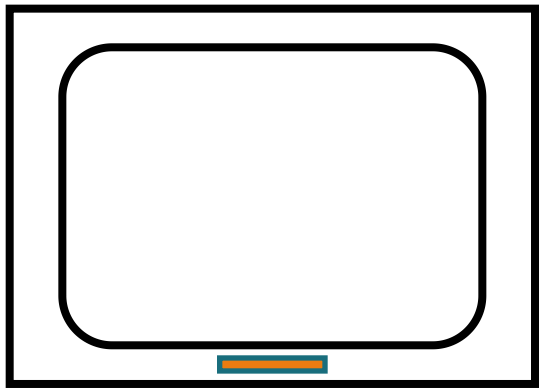


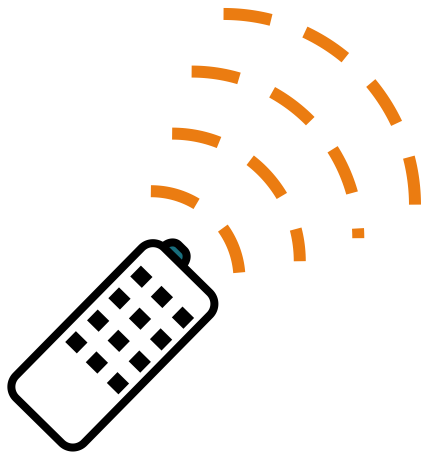
Course (e.g. Algebra)

What Is Encapsulation?



- Hides implementation from clients.
  - Clients depend on interface.

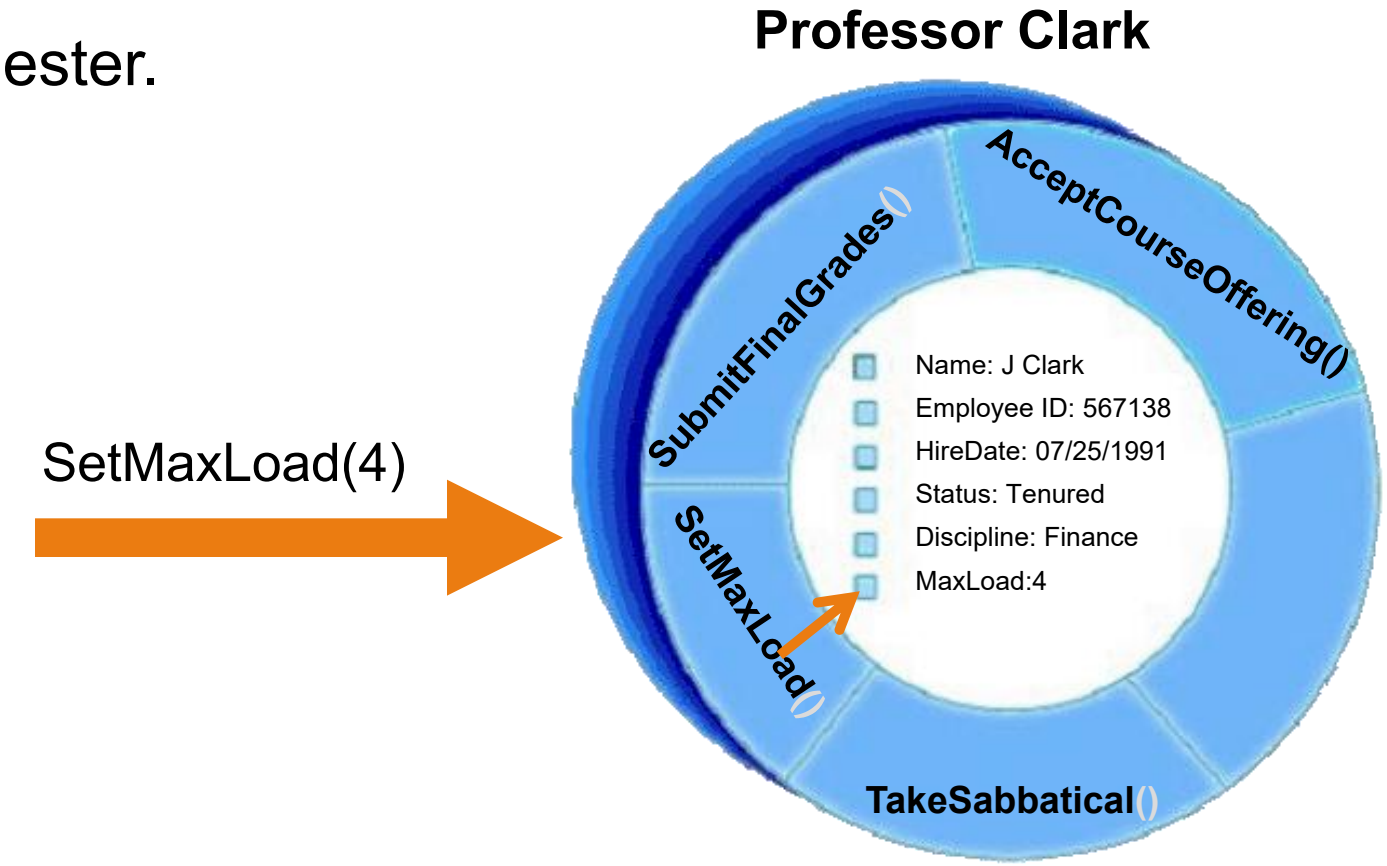




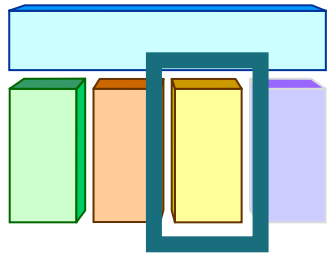
Improves Resiliency

Encapsulation Illustrated

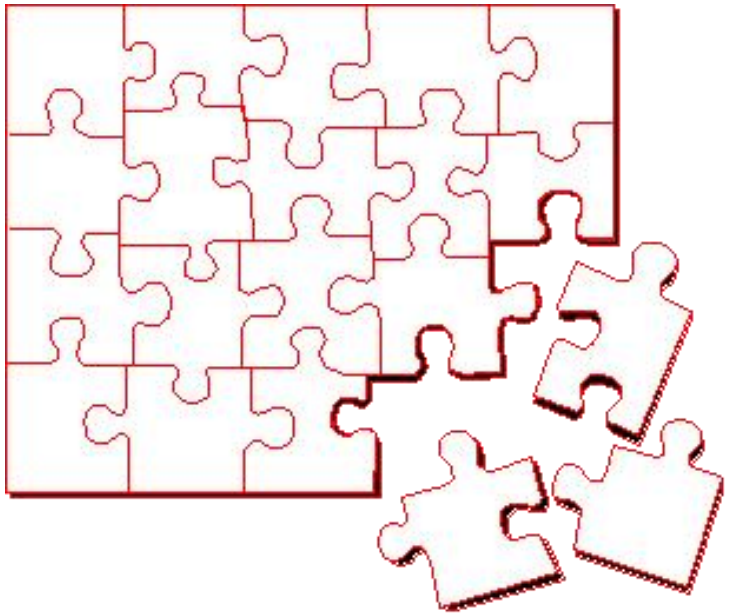
- Professor Clark needs to be able to teach four classes in the next semester.



What Is Modularity?

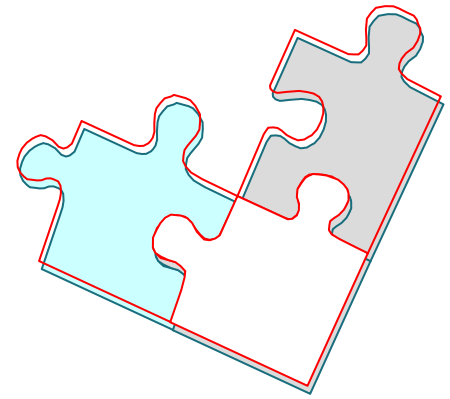


- Breaks up something complex into manageable pieces.
- Helps people understand complex systems.

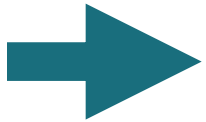


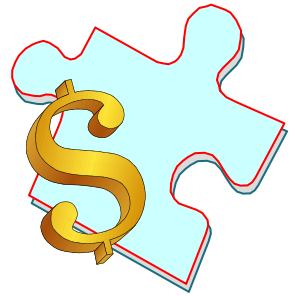
Example: Modularity

- For example, break complex systems into smaller modules.

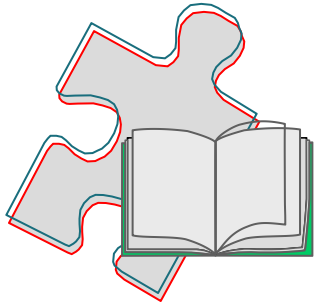


Course Registration System

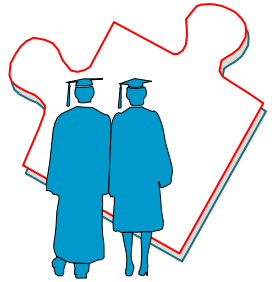




Billing System

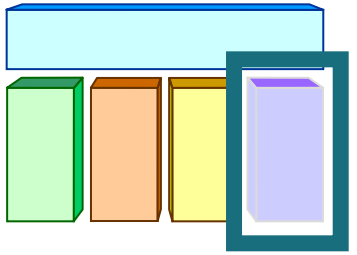


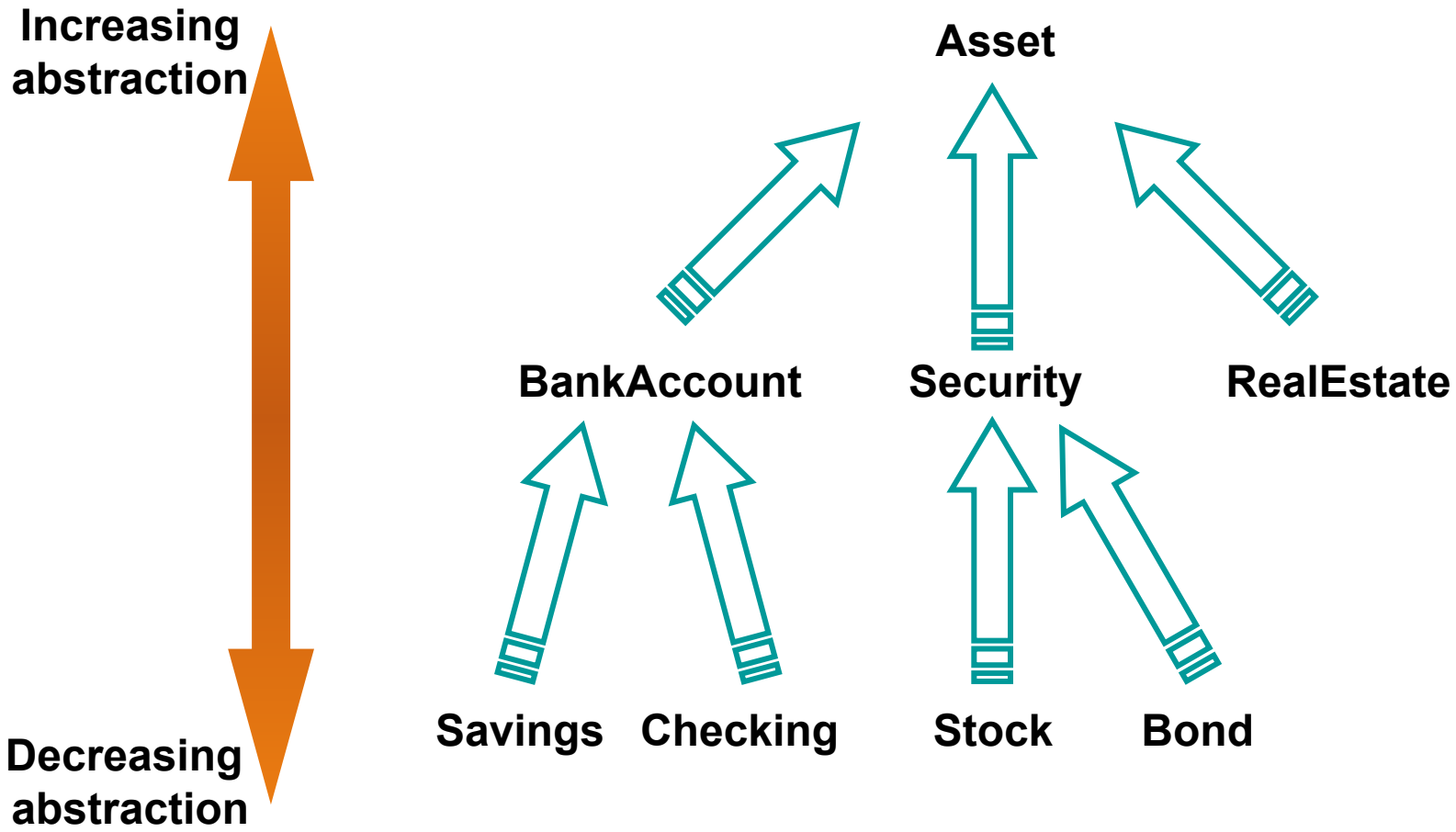
Course Catalog System



Student Management System

What Is Hierarchy?





Elements at the same level of the hierarchy should be at the same level of abstraction.

What Is a Class?

- A class is a description of a set of objects that share the same *attributes*, *operations*, *relationships*, and *semantics*.
  - An object is an instance of a class.
- A class is an abstraction in that it
  - Emphasizes relevant characteristics.
  - Suppresses other characteristics.

Representing Classes in the UML

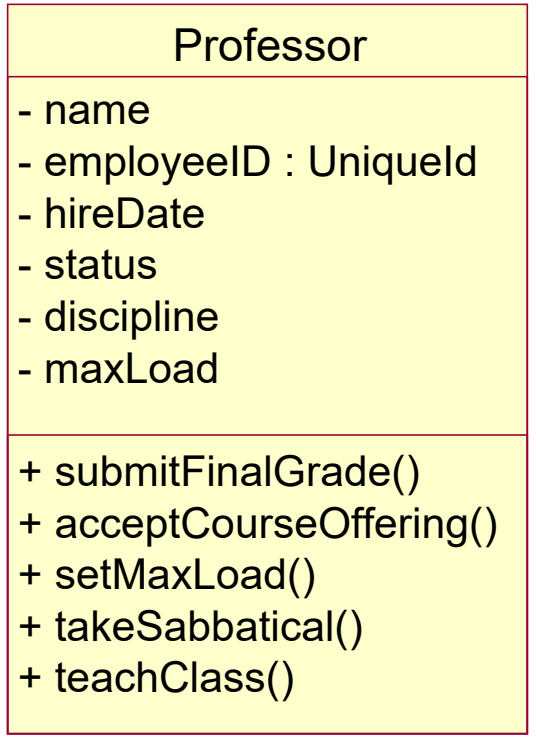
- A class is represented using a rectangle with three compartments:
  - The class name
  - The structure (attributes)
  - The behavior (operations)

Visibility:

Public: +

Private: -

Protected: #



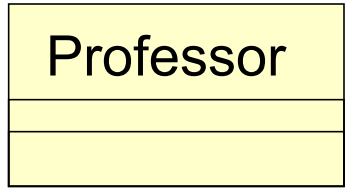
The Relationship between Classes and Objects

- A class is an abstract definition of an object.
  - It defines the structure and behavior of each object in the class.
  - It serves as a template for creating objects.
- Classes are not collections of objects.

Professor Torpie

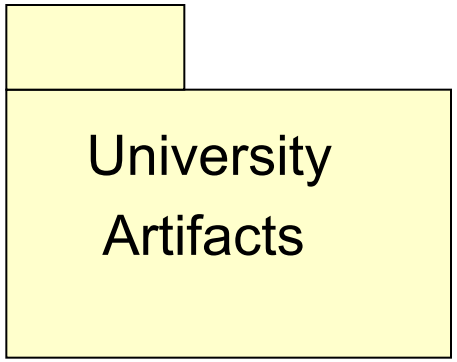
Professor Meijer

Professor Allen



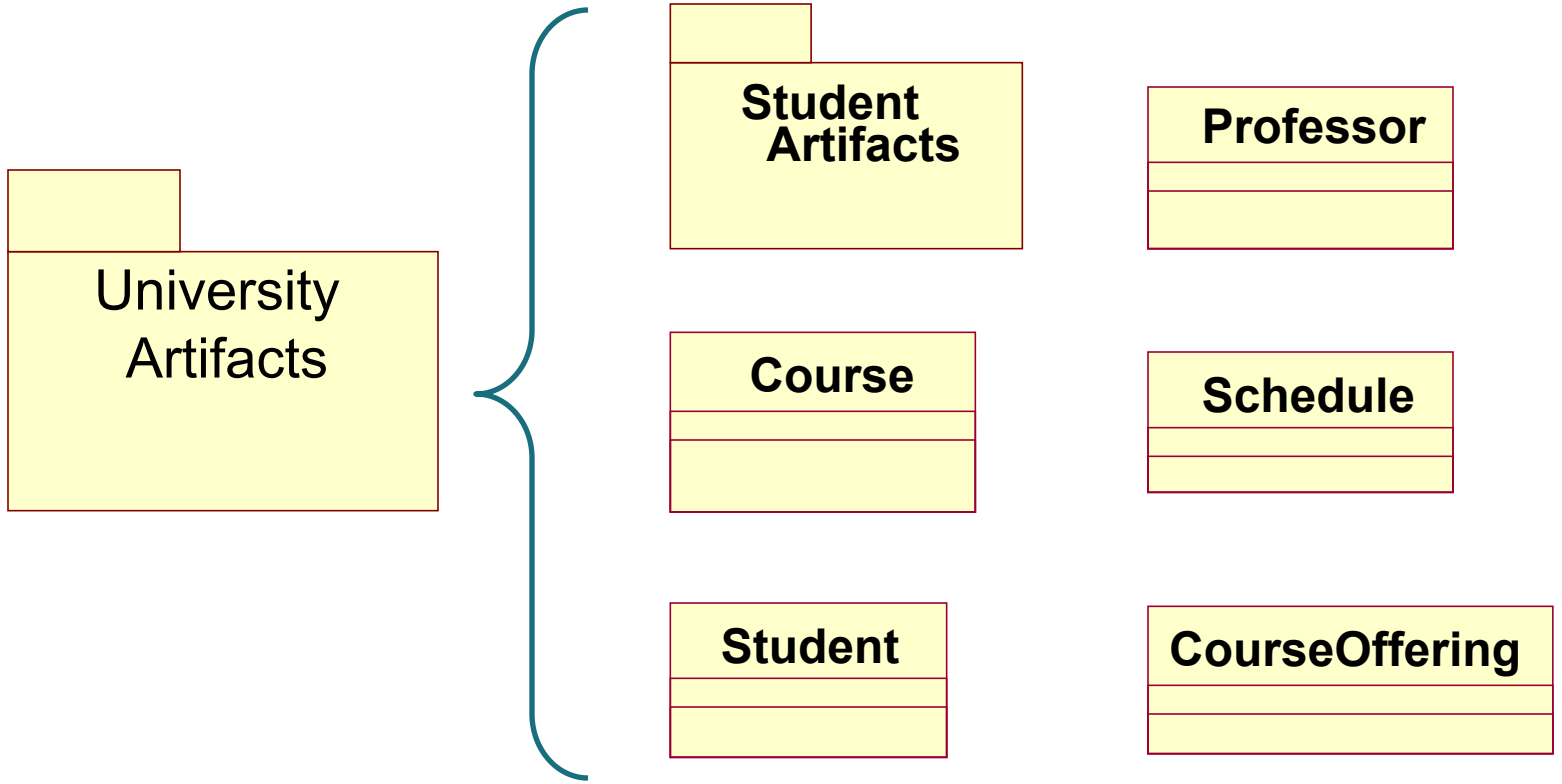
What Is a Package?

- A general purpose mechanism for organizing elements into groups.
- A model element that can contain other model elements.
- A package can be used:
  - To organize the model under development.
  - As a unit of configuration management.



Package 示例

- The package, University Artifacts, contains one package and five classes.



面向对象方法的步骤

- 面向对象分析
  - Object Oriented Analysis, OOA
- 面向对象设计
  - Object Oriented Design, OOD
- 面向对象编程
  - Object Oriented Programming, OOP
- 面向对象测试
  - Object Oriented Testing, OOT