组件技术

潘爱民

panaimin@icst.pku.edu.cn

内容

- 关于本课程
- 软件技术的发展
- 当前软件状况
- COM和CORBA简介
- 本课程的内容
- 课程安排

关于本课程

- 加强基础
- 展示最新软件技术
- 提高软件设计能力

软件领域的发展

- 程序设计方法的发展
- 软件需求的变化
- 软件环境的变化
- 软件开发模式的变化

程序设计方法的发展

- 功能分解法 —— 计算任务
- 结构化程序设计 —— 以数据为中心
- 面向对象程序设计 —— 以对象为中心
- 组件程序设计 —— 以组件为中心

综合使用

软件需求的变化

- 计算
- 实用
- 管理(MIS)
- 分布式系统

软件环境的变化



软件开发模式

机器码



高级语言
和脚本代码

OS + Compiler + Editor



OS + Compilers + IDEs
+ Libraries + Wizards
+ Components
+ other inforstructures
+ 经验

发展变化的结果

- 软件学科的形成
- 不变因素：
  - 计算机的基本运作模式

回顾软件学科10多年来的变化

- 软件的产生过程：

  原始代码->编译->Link->终代码、脚本 — 混合

- 软件的分发过程：

  文档、说明书、自动安装工具、培训、升级

- 软件工程：

  需求-〉分析-〉设计-〉code->test->使用

- 对象技术和组件技术的不断普及，与大环境发展相结合

对象技术

- 用一种新的思路来看待问题
  - 10多年的发展证明了OO符合软件的规律
  - 对象技术的发展需要开发工具和开发语言的支持
- 三大特性：封装性、继承性、多态性
- Object-Based
- Object-Oriented

组件技术

- 组件技术适合于现代的软件环境
  - 软件更新快、规模大、强调协作等
  - 组件的概念、软件IC
- 组件可独立发布要求“接口与实现分离”
- 组件infrastructure，组件体系
- COM、CORBA（和Java）
- 中间件(middleware)市场

国内的状况

- 在对象技术和组件技术方面，国内落后
- 问题：
  - 好的技术没能及时引进来
  - 缺乏好的教材、技术杂志、技术书籍
  - 缺乏良好的教育环境和工业环境

基础

- 编译过程，程序产生的过程

- C++之必要性：
  - 最能反映对象技术、组件技术的核心概念
  - 存在的问题：
    - 缺乏好的书籍
    - 国内的应用领域不够广泛

概念：类、对象

- 类是定义，它封装了状态和操作
- 对象是实例，它反映了具体的事物

- 对象由类来定义
- 一个对象可以与多个具有继承关系的类相联系

概念：组件

- 定义：可独立发布的二进制单元
- 有的地方称作“package”

- 容易混淆：
  - 服务器(server)、对象
- 例如：
  - COM组件、Java Applet

概念：reuse（重用、复用）

- 软件设计的目标
- 不同形式的reuse
- 不同层次的reuse
- 例子：
  - 类库、中间件

概念：接口与实现分离

- Interface and implementation
- 对象技术和组件技术都强调

- 接口是对象或者组件的通信协议
- 实现是对象或者组件的内部细节

- 接口的发布
- 客户只关心接口

概念：decouple

- 去耦
- 尽可能地消除软件之间或者软件的不同部分之间的联系
- 是软件设计的重要手段，也是目标
- 不同层次上的decouple

概念：多态性(polymorphism)

- 对象技术中的重要概念
- 以一致的方式来看待不同的事物

- 在C++中，通过继承和虚成员函数来实现
- 运行时刻绑定：runtime binding

- 在组件技术中，通过接口来体现

概念：module

- 模块
- 不同层次上有不同的定义
- 在Windows和Linux中，指可独立装载的二进制代码单元
- 有些操作与模块相关

概念：模板(template)

- 函数模板和类模板
- 抽象层次更高
- 泛型技术(generic programming)

- 已纳入C++标准之中

- 例子：
  - STL、ATL

概念：C/S和多层结构

- 这是指软件的结构
- C/S是基础
- 多层结构具有较好的灵活性和可扩展性
- 多层结构与基于组件的系统相结合
- 变种：
  - B/S

概念：其他

- 现代软件的发展形成了很多重要的概念
- 掌握这些概念对于提高自己的软件理解水平非常重要
- 防止混淆和滥用概念

软件形式

- 单机软件
  - 整体发布
- Client/Server软件
  - 简单的网络应用
- 分布式软件

软件发展面临的问题

- 软件的复杂度增加
  - 代码开发管理
- 软件升级和更新
  - 牵一而发动全身？
- 分布式软件的发布和配置
  - 用户需要培训，程序员和设计人员更需要

软件发展面临的问题(续)

- Custom-made versus standard software
  - 如何平衡？
- 依赖于全才开发人员
  - 软件层次划分
- 软件的可扩展性
  - 用户可定制、脚本支持
- 与标准接轨
  - 如Web ——〉B/S

软件发展趋势

- 组件软件、软件组件
  - 中间件市场已经形成
- 软件基础设施的重要性
  - 经验的重用
- 软件的可重用性要求
  - 重用粒度不同
- API的多样化
- 软件方案(solution)，consulting市场

组件软件

- 部分升级
  - 补丁
  - 缩短软件的更新周期
- 增加了异构软件的可能
  - 标准接口、标准规范、标准协议
- 动态组合，提高软件生产力
- 符合现代软件的发展要求

软件组件

- 关键：接口设计
  - 接口不变性
  - 接口可扩展性
- 对基础设施的依赖
  - COM、CORBA、…
- 内部独立设计、独立开发
- 独立发布、独立测试

基础设施：组件体系

- COM(Component Object Model)
  - 93年发布，脱胎于MS的OLE
  - 规范+实现
- CORBA(Common Object Request Broker Architecture)
  - 91年第1版，OMG(Object Management Group)
  - 规范+实现

COM：组件对象模型

- 建立在二进制层次上的标准
  - 许多概念与C++的特征相似，如vtable
  - 程序与程序之间的通信建立在此基础上
- COM规范
  - 平台无关
  - 定义了大量的标准接口用于各种用途
- COM实现
  - 平台相关
  - Windows实现了规范要求以及许多辅助功能

COM特性

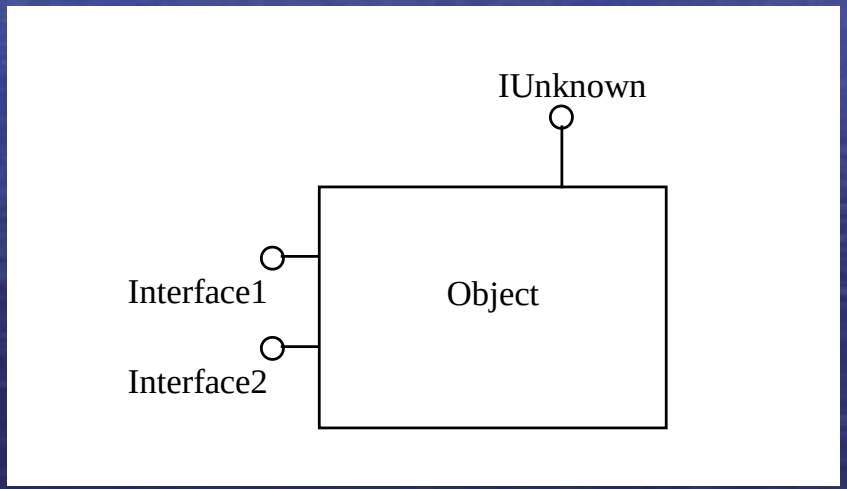
- 语言无关
  - 规范和实现都是语言无关的
- 进程透明 ——〉位置透明
  - inproc server、out-of-proc server
- 可重用性
- 多线程特性
- 可扩展性

COM：接口

- COM接口
  - IID：128位整数
  - 接口定义语言COM IDL
  - 对应于C++中由纯虚函数组成的类
  - 单继承，基接口：IUnknown
- COM接口特性：
  - 接口不变性
  - 继承性
  - 多态性——运行时刻的多态性

COM：对象

- COM对象
  - CLSID
  - 实现多个接口



- COM对象特性：
  - 身份确定
  - 封装
  - 可重用

COM：组件

- COM对象的载体
  - 包含多个COM对象
- 进程内组件
  - DLL
  - Export Entrys
- 进程外组件
  - IPC：RPC/LPC

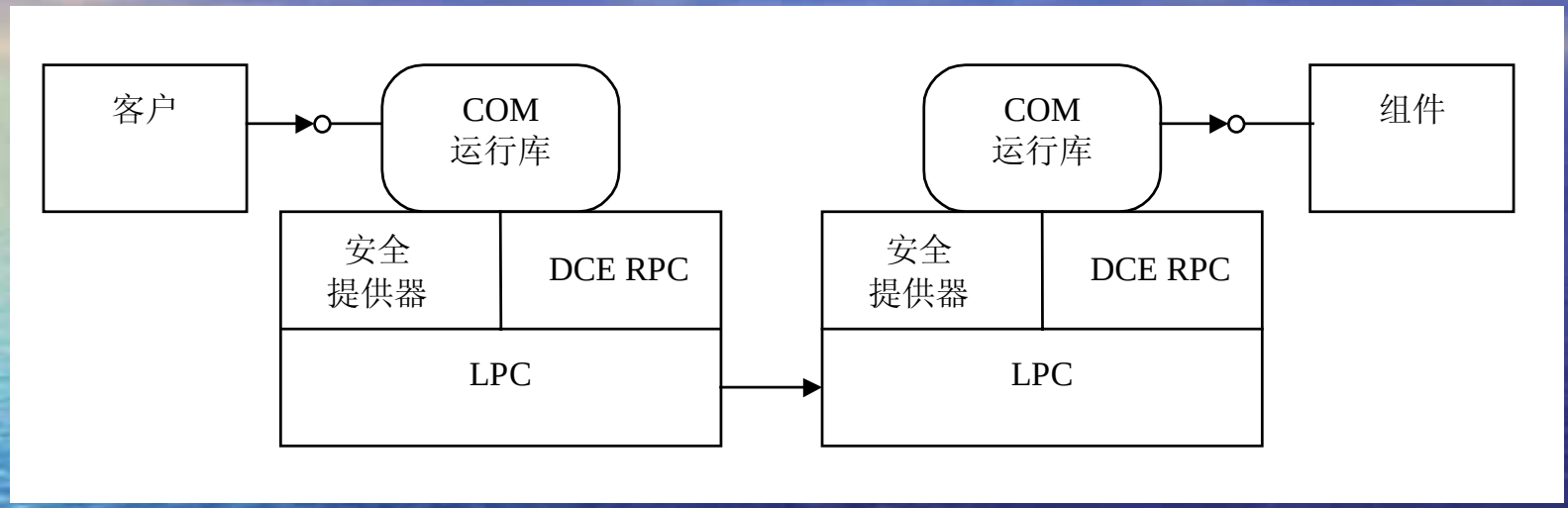
COM：进程内组件

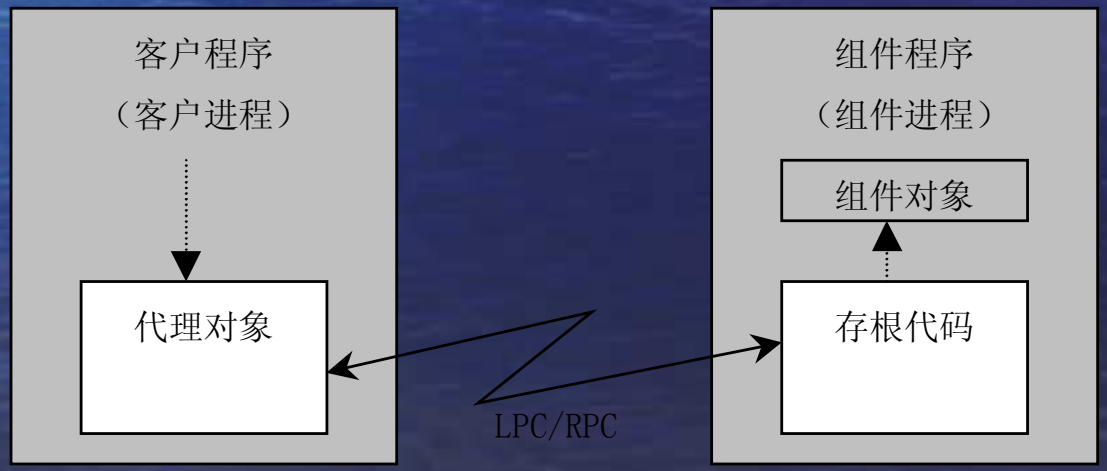
- Inproc server
  - Export Entrys



COM：进程外组件

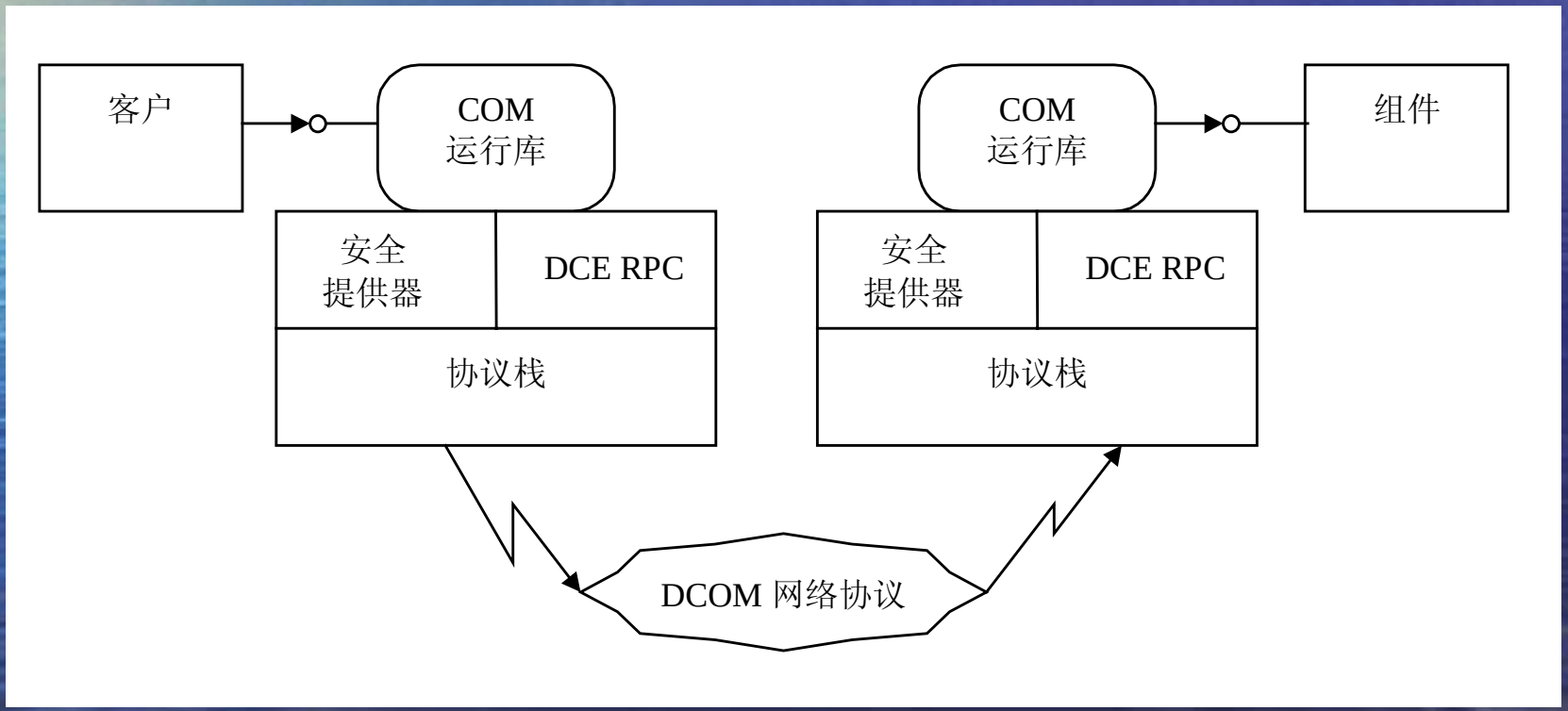
- Out-of-proc server





COM：远程组件

- remote server

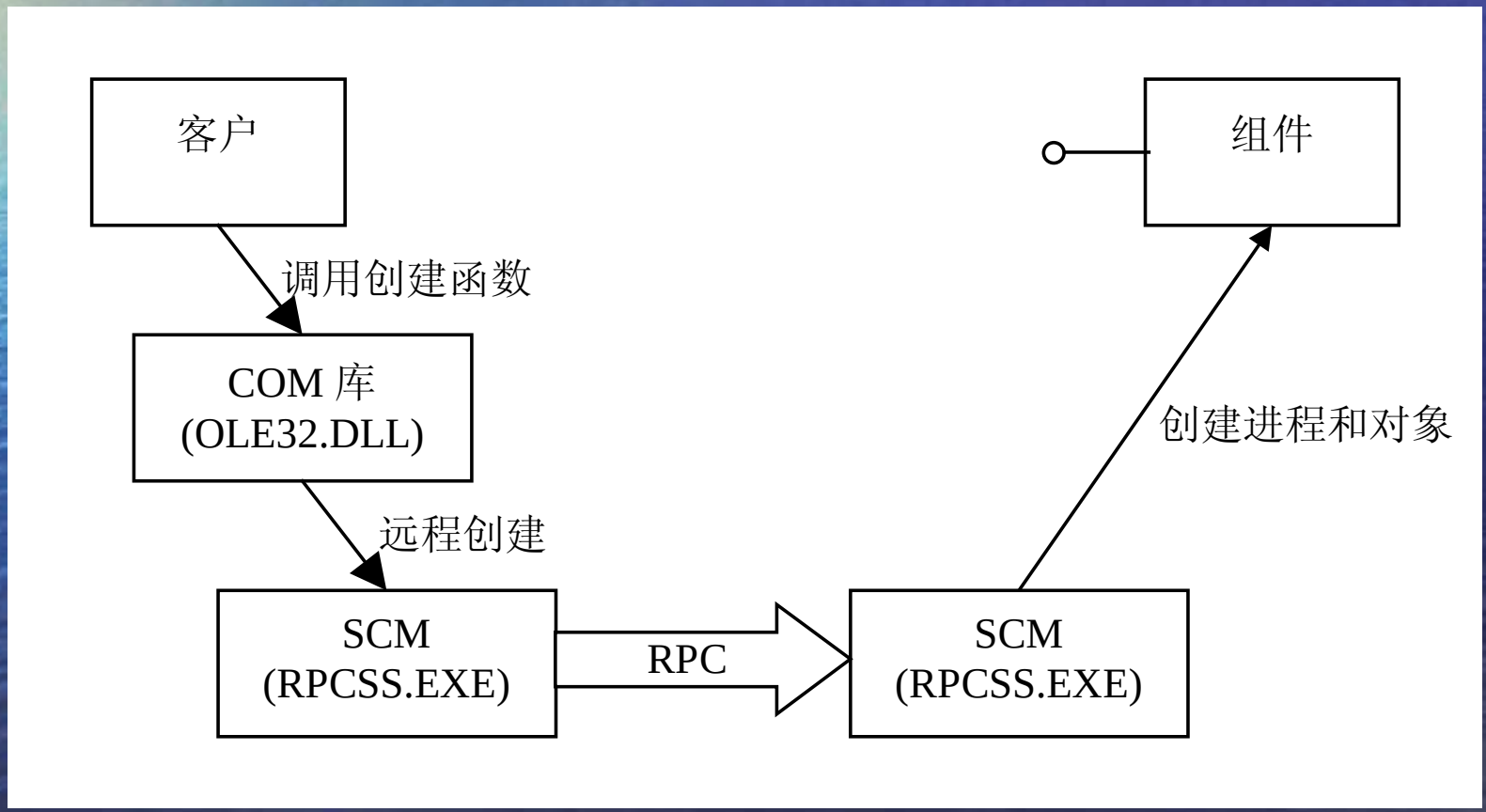


COM：组件信息发布

- 注册表
  - 发布组件的信息、对象的信息、接口的信息等
- 注册工具和配置工具
  - Regsvr32.exe
  - .reg
  - 自注册 -RegServer
  - DCOMCNFG.EXE
  - MTS Explorer、活动目录

COM：创建COM对象

- 对象的创建过程



COM：COM对象管理

- 客户使用规则，控制对象生命周期
- COM实现优化
  - 各种辅助手段
- 组件类别 component category
- MTS和COM+

COM：COM体系

- 基本的功能
- 扩展功能
  - 连接点
  - 结构化存储
  - 名字服务
  - 自动化
  - 其他
- COM+

COM：COM+

- 建立在COM的基础上，COM的自然延伸
- Services
  - 事件服务
  - 队列服务
  - 事务服务
  - 负载平衡
  - 其他

COM：应用

- OLE
- ActiveX control
- 脚本功能扩展
- OLE DB
- ADSI
- ASP
- 其他

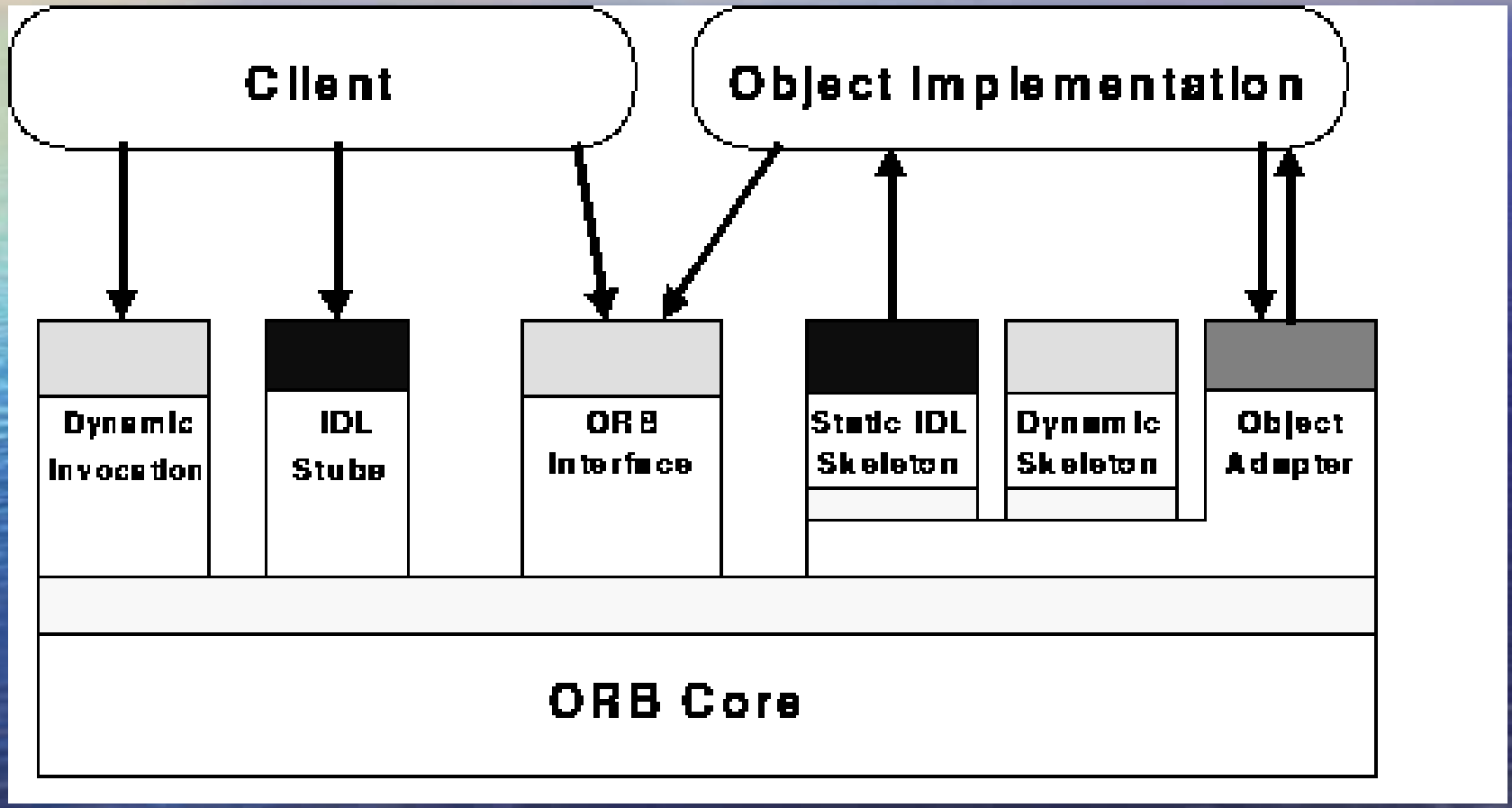
COM：分布式应用

- COM/COM+为基于Windows的分布式应用提供了基础设施
  - Transaction
  - Security
  - OLE DB
  - Web
  - 集成Active Directory
  - 管理功能

CORBA

- OMG，800多家厂商联合制定的标准
  - 91年 1.0，主要是C语言映射
  - 95年 2.0，加入IIOP
  - 目前最新版本2.3
- CORBA规范
  - 制订了互操作规范，上层标准协议
  - 映射到多种语言的规范
- CORBA实现：ORB

CORBA体系结构

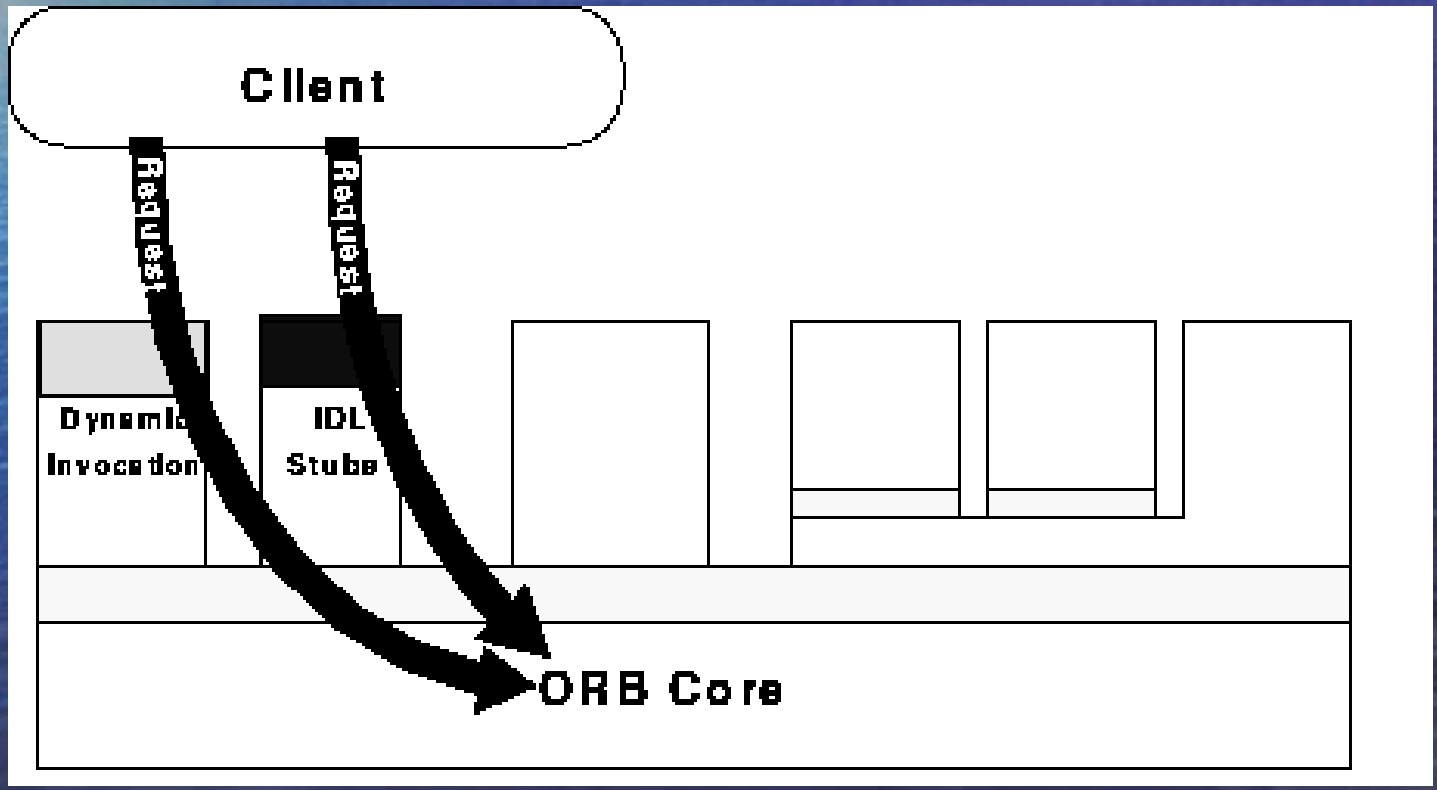

Client
Object Implementation
Dynamic Invocation
IDL Stubs
ORB Interface
Static IDL Skeleton
Dynamic Skeleton
Object Adapter
ORB Core

CORBA：跨平台、跨语言

- 典型的语言：C++语言、Java语言、Smalltalk
- 多种平台
  - 关键：IOR(Interoperable Object reference)
- ORB
  - IONA的Orbix
  - Visigenic Software的Visibroker
  - 东南大学的ORBUS

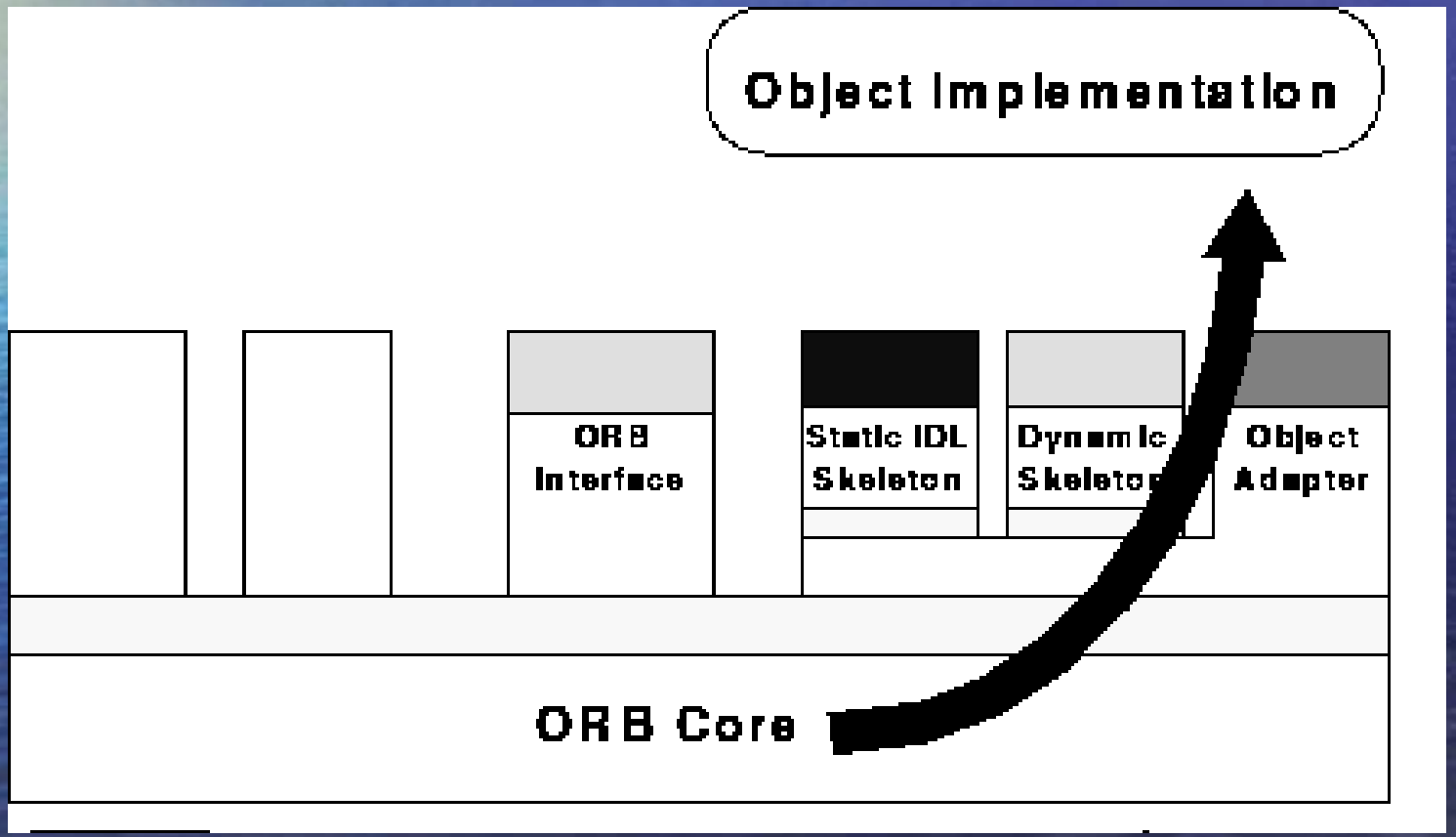
CORBA：客户

- 客户通过stub或者CORBA库与对象或者ORB进行通信



CORBA：对象

- 对象与OA进行通信



CORBA：C++映射开发过程

- 编写IDL
- IDL-〉C++
- 利用生成的stub类编写客户
- 利用生成的impl类编写服务器对象
- {以字符串形式传递OR}

CORBA：核心

- CORBA IDL
  - 支持多继承
  - 支持结构化数据类型
- 字符串形式的OR
- POA
  - CORBA对象的环境context
  - 策略设置

CORBA：核心

- GIOP(General Inter-ORB Protocol)
  - 互操作协议框架
  - IIOP(Internet Inter-ORB Protocol)
- IOR
  - CORBA对象的环境context
- 实现仓库、接口仓库
  - 未标准化

CORBA：服务

- 命名服务
  - 层次命名
  - 绑定
- Trading Service
  - 功能更强、使用更灵活的对象查询服务
- 事件服务
  - 事件发送方和接受方之间的decouple

课程内容(一)

- 准备知识
  - 组件技术介绍
  - C++基础

课程内容(二)

- COM/DCOM/COM+
  - COM接口和IDL
  - COM对象和COM实现
  - COM重用性和COM跨进程实现
  - 自动化和脚本技术
  - COM开发，安排作业
  - COM连接点、命名和绑定技术
  - 结构化存储和统一数据传输
  - DCOM和ActiveX控制
  - COM+

课程内容(三)

- 设计模式
- CORBA介绍
- 中间插1-2次新技术报告

考核办法

- 作业
  - 1-2次
- 考试

参考书目

- 《COM本质论》
- 《 COM原理与应用》
- 《 COM技术内幕》
- 《 Design Pattern 》或者《设计模式》
- 《 Advanced CORBA Programming with C++ 》