

<<高级软件工程>>

图书基本信息

书名：<<高级软件工程>>

13位ISBN编号：9787302257202

10位ISBN编号：7302257205

出版时间：2011-6

出版时间：清华大学出版社

作者：方木云，刘辉 编著

页数：391

字数：637000

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<高级软件工程>>

内容概要

本书介绍20世纪90年代中期以后软件工程领域新的理论和技术，围绕如何提高软件的生产效率和质量来选取和组织材料。

前11章为理论部分，内容包括软件体系结构、中间件技术、在线演化技术、网构软件、仿生学、面向Agent的软件工程、面向服务架构技术、柔性 workflow、统一建模语言、云计算；后6章为实验部分，内容包括用VB实现的UDP通信、Windows API函数、端口的禁用、用VB写DLL文件、用VB写OCX控件以及COM、COM+和DCOM。

本书可作为高校研究生的教材，也可以作为教师和广大高级编程人员的参考书。

<<高级软件工程>>

书籍目录

理论篇

- 第1章 软件工程新技术31.1 新的构建技术
 - 1.2 新的软件形态
 - 1.3 新的开发工具
 - 1.4 大规模的应用
- 第2章 软件体系结构
 - 2.1 软件体系结构的形成
 - 2.1.1 无体系结构阶段
 - 2.1.2 萌芽阶段
 - 2.1.3 初级阶段
 - 2.1.4 高级阶段
 - 2.2 软件体系结构的概念
 - 2.3 软件体系结构的要素
 - 2.3.1 构件
 - 2.3.2 连接件
 - 2.3.3 约束
 - 2.4 软件体系结构的作用
 - 2.5 软件体系结构描述语言
 - 2.5.1 ACME
 - 2.5.2 Wright
 - 2.5.3 C
 - 2.5.4 UniCon
 - 2.5.5 其他比较有影响的ADL
 - 2.5.6 ADL分类与比较框架
 - 2.5.7 ADL之间的比较
 - 2.6 软件体系结构描述方法40目录高级软件工程 2.7 软件体系结构风格
 - 2.7.1 软件体系结构风格的概念
 - 2.7.2 软件体系结构风格的研究内容
 - 2.7.3 几种常见的软件体系结构风格
 - 2.7.4 新的软件体系结构风格的提出
 - 2.7.5 讨论软件体系结构风格时常见的问题
 - 2.8 软件体系结构评估
 - 2.8.1 软件体系结构评估的目的
 - 2.8.2 软件体系结构评估的方法与工具
 - 2.8.3 软件体系结构评估存在的问题
 - 2.9 基于体系结构的软件开发过程
 - 2.10 Windows体系结构介绍
 - 2.10.1 Windows系统的发展历史
 - 2.10.2 Windows系统的描述语言
 - 2.10.3 Windows NT体系结构简介
- 思考题
- 第3章 中间件技术
 - 3.1 中间件技术的形成
 - 3.2 中间件的基本概念
 - 3.2.1 中间件的定义

<<高级软件工程>>

- 3.2.2 中间件的分类
 - 3.2.3 中间件的新类型
 - 3.2.4 中间件的作用
 - 3.2.5 中间件面临的问题
 - 3.3 典型中间件
 - 3.3.1 消息中间件
 - 3.3.2 交易中间件
 - 3.3.3 对象中间件
 - 3.3.4 应用服务器
 - 3.3.5 安全中间件
 - 3.4 基于中间件的应用开发
 - 3.4.1 中间件基础技术
 - 3.4.2 基于中间件的软件开发过程
 - 3.5 中间件的现状
 - 3.6 中间件的发展趋势
- 思考题
- 第4章 在线演化技术
- 4.1 软件演化的产生
 - 4.2 软件演化理论
 - 4.2.1 演化与维护的关系
 - 4.2.2 软件演化的分类
 - 4.3 构件运算的概念及其性质
 - 4.3.1 构件与软件复用
 - 4.3.2 构件软件系统的抽象描述
 - 4.3.3 构件运算的性质
 - 4.3.4 构件运算的完整性证明
 - 4.4 软件演化方法
 - 4.4.1 基于构件运算的软件演化
 - 4.4.2 基于体系结构的软件演化
 - 4.4.3 软件演化的核心算法
 - 4.4.4 构件运算表达式的体系结构化
 - 4.5 基于构件的软件在线演化实现
 - 4.5.1 基于构件的演化信息的捕获方法
 - 4.5.2 基于构件的软件在线演化框架实现
 - 4.6 软件演化研究现状
- 思考题
- 第5章 网构软件
- 5.1 网构软件的产生
 - 5.2 网构软件的特征
 - 5.3 基于开放软件模型的网构软件模型
 - 5.3.1 开放软件模型
 - 5.3.2 网构软件模型
 - 5.3.3 网构软件中间件
 - 5.3.4 网构软件按需协同的实现模型
 - 5.4 基于Agent的网构软件模型
 - 5.4.1 开发协同模型框架的设计途径
 - 5.4.2 基于移动Agent的协同程序设计技术

<<高级软件工程>>

- 5.4.3 多模式交互机制及基于Agent中间件模型
- 5.4.4 面向体系结构的协同程序设计方法
- 5.5 以体系结构为中心的网构软件开发方法
 - 5.5.1 ABC方法概览
 - 5.5.2 基于特征的网构软件需求建模
 - 5.5.3 基于体系结构的网构软件自适应性建模
- 5.6 网构软件的实现和测试方法的研究
 - 5.6.1 网构软件
 - 5.6.2 软件测试
- 5.7 网构软件带来的挑战
- 思考题
- 第6章 仿生学
 - 6.1 仿生学的定义及意义
 - 6.1.1 仿生学的定义
 - 6.1.2 仿生学的意义
 - 6.2 仿生学的现状及前景
 - 6.2.1 仿生学的现状
 - 6.2.2 仿生学的前景
 - 6.3 仿生学的学科特征及研究方法
 - 6.3.1 仿生学的学科特征
 - 6.3.2 仿生学的研究方法
 - 6.4 仿生学在计算机领域的应用
 - 6.4.1 计算机结构的仿生特性
 - 6.4.2 软件的仿生学生产方法
 - 思考题
- 第7章 面向Agent的软件工程
 - 7.1 面向Agent的技术
 - 7.1.1 Agent的基本概念
 - 7.1.2 Agent的体系架构
 - 7.1.3 Agent的组织类型
 - 7.1.4 Agent与组件的对比
 - 7.1.5 语义复用的Agent和组件
 - 7.2 面向Agent的软件工程
 - 7.2.1 面向Agent的研究内容
 - 7.2.2 面向Agent的开发过程
 - 7.3 面向Agent的经典开发方法
 - 7.3.1 Gaia方法
 - 7.3.2 Tropos方法
 - 7.3.3 MASE方法
 - 7.4 按需定制的开发框架HDA
 - 7.4.1 HDA的定义
 - 7.4.2 HDA的使用规则
 - 7.4.3 元模型
 - 7.4.4 潜在的问题
 - 7.4.5 Agent建模阶段
 - 7.5 多Agent系统的基础设施
 - 7.5.1 MAS的基础设施

<<高级软件工程>>

- 7.5.2 基础设施的授权与协调
- 7.5.3 授权型基础设施JADE
- 7.5.4 MAS的协调基础设施
- 7.5.5 协调对MAS工程化的影响
- 7.5.6 MAS协调的行为理论框架
- 7.5.7 制品与协调基础设施
- 7.5.8 MAS工程中的协调平衡

思考题

第8章 面向服务架构技术

- 8.1 面向服务体系架构的产生
 - 8.1.1 传统的软件体系结构
 - 8.1.2 基于组件的软件体系结构
 - 8.1.3 面向服务体系阶段
- 8.2 SOA的基本概念
 - 8.2.1 SOA的定义
 - 8.2.2 SOA的架构
 - 8.2.3 SOA的优点
- 8.3 支持SOA的Web Service技术
 - 8.3.1 SOA的基本原则
 - 8.3.2 基于SOA的解决方案
 - 8.3.3 Web Service技术
 - 8.3.4 SOA架构的不足
- 8.4 适于SOA的建模方法
 - 8.4.1 MDA
 - 8.4.2 敏捷方法
 - 8.4.3 SOA的五视图法
- 8.5 基于SOA架构的软件开发方法
 - 8.5.1 面向服务的架构
 - 8.5.2 基于SOA架构技术的优势
 - 8.5.3 SOA应用的构建步骤
 - 8.5.4 SOAD与传统软件开发的区别
- 8.6 基于SOA架构的ERP系统
 - 8.6.1 SOA在ERP中的应用
 - 8.6.2 基于SOA的ERP系统总体设计
 - 8.6.3 基于SOA的ERP系统架构

思考题

第9章 柔性工作流

- 9.1 柔性工作流的概念
 - 9.1.1 工作流的定义
 - 9.1.2 工作流的要素
 - 9.1.3 柔性工作流的概念
- 9.2 工作流管理系统的参考模型
 - 9.2.1 工作流管理系统概述
 - 9.2.2 工作流管理系统的参考模型
 - 9.2.3 工作流管理系统的体系结构
- 9.3 柔性工作流的描述
 - 9.3.1 柔性工作流模型的组成

<<高级软件工程>>

- 9.3.2 柔性工作流模型的形式化描述
- 9.3.3 基于XML的组件式柔性工作流模型描述

- 9.4 柔性工作流的建模
 - 9.4.1 基本模型单元
 - 9.4.2 模型扩展属性
 - 9.4.3 运行时的动态支持
 - 9.4.4 工作流角色
 - 9.4.5 柔性工作流引擎框架

- 9.5 柔性工作流的应用
 - 9.5.1 柔性工作流应用的必要性
 - 9.5.2 柔性工作流实现的技术
 - 9.5.3 柔性化应用系统的特征

思考题

第10章 统一建模语言

- 10.1 UML概述
 - 10.1.1 UML产生的背景
 - 10.1.2 UML定义
- 10.2 UML的研究内容
 - 10.2.1 UML语言的定义
 - 10.2.2 UML的图形表示法
- 10.3 UML建模
 - 10.3.1 UML建模过程
 - 10.3.2 UML在RUP中的应用
- 10.4 基于UML的集成化CASE平台的模仿设计
 - 10.4.1 CASE的起源和发展
 - 10.4.2 集成CASE环境
 - 10.4.3 参考模型——Rational Rose
 - 10.4.4 模型检查
 - 10.4.5 平台的设计原则
 - 10.4.6 平台设计的关键技术
 - 10.4.7 平台的功能设计
 - 10.4.8 平台的具体实现

思考题

第11章 云计算

- 11.1 云计算的基本概念和主要特征
 - 11.1.1 云计算的基本概念
 - 11.1.2 云计算的主要特征
 - 11.1.3 云计算的分类
 - 11.1.4 云计算与网格计算
 - 11.1.5 云计算的现状和发展趋势
- 11.2 云计算的原理与关键技术
 - 11.2.1 云计算的原理
 - 11.2.2 云计算的实现机制
 - 11.2.3 Google云计算技术
- 11.3 云计算与移动Internet
 - 11.3.1 移动Internet的发展概况
 - 11.3.2 云计算助力移动Internet发展

<<高级软件工程>>

11.3.3 移动Internet的“端”-“管”-“云”

11.4 云计算与物联网

11.4.1 物联网的定义

11.4.2 物联网的三大特征

11.4.3 物联网的前景

11.4.4 物联网的关键技术

11.4.5 物联网云计算

11.4.6 物联网云计算运营平台的构建

思考题

实验篇

第12章 用VB实现的UDP通信

12.1 网络开发基本概念

12.1.1 TCP/IP 协议体系

12.1.2 网络编程术语

12.1.3 Socket介绍

12.2 Socket网络编程原理

12.2.1 面向连接的套接字的编程模式

12.2.2 无连接协议的套接字的编程模式

12.3 用VB实现的UDP编程方式

12.3.1 WinSock控件的使用

12.3.2 WinSock API的使用

12.3.3 总结比较

12.4 在VB下使用WinSock控件开发UDP聊天室

第13章 Windows API函数

13.1 Windows API简介

13.2 API函数分组

13.3 传统类库、控件在软件开发中的不足

13.4 在VB中调用API函数

第14章 端口的禁用

14.1 端口的定义

14.2 端口的分类

14.3 常用的端口

14.4 常被黑客利用的端口

14.5 端口的禁用

第15章 用VB写DLL文件

15.1 DLL的简单介绍

15.2 用VB做一个DLL文件

15.3 对ActiveX DLL的测试

15.3.1 在VB上的测试

15.3.2 在VC上的测试

15.3.3 注意事项

第16章 用VB写OCX控件

16.1 OCX控件的介绍

16.2 用VB做一个OCX控件

16.3 对OCX控件的测试

16.3.1 在VB上的测试

16.3.2 在VC上的测试

<<高级软件工程>>

16.3.3 注意事项

第17章 COM、COM+、DCOM

17.1 COM

17.1.1 COM组件的特点

17.1.2 COM接口和组件类型

17.1.3 COM组件的设计和应用

17.2 COM+

17.2.1 COM+简介

17.2.2 COM+的特点

17.3 DCOM

17.3.1 DCOM的使用

17.3.2 DCOM的特点

17.3.3 DCOM的灵活配置和扩展机制

17.3.4 在应用间共享连接管理

17.3.5 DCOM的安全性设置

参考文献

章节摘录

版权页：插图：连接件（connector）是用来建立构件间的交互以及支配这些交互规则的体系结构构造模块。

构件之间的交互包括消息或信号量的传递，功能或方法调用，数据的传送和转换，构件之间的同步关系、依赖关系等。

在最简单的情况下，构件之间可以直接完成交互，这时体系结构中的连接件就退化为直接连接。

在更为复杂的情况下，构件间交互的处理和维持都需要连接件来实现。

常见的连接件有管道（pipe）（管理过滤器体系结构中）、通信协议或通信机制（客户服务器体系结构中）等。

连接件的接口由它与所连接构件之间的一组交互点构成，这些交互点被称作角色。

角色代表了参与连接的构件的作用和地位，并体现了连接所具有的方向性。

因此，角色有主动和被动、请求和响应之分。

体系结构级的通信需要用复杂协议来表达，为了抽象这些协议并使之能够复用，可以将连接件构造为类型。

构造连接件类型可以将那些用通信协议定义的类型系统化，并对立于实现，或者作为内嵌的、基于它们的实现机制的枚举类型。

为了保证体系结构中的构件连接件连接以及它们之间的通信正确，连接件应该导出所期待的服务作为它的接口。

为完成接口的有用分析、保证跨体系结构抽象层的细化一致性、强制互联与通信约束等，体系结构描述提供了连接件协议以及变换语法。

为了确保执行计划的交互协议，建立起内部连接件依赖关系，强制用途边界，就必须说明连接件约束。

连接件的主要特性有可扩展性、互操作性、动态连接性和请求响应特性。

连接件的可扩展性是连接件允许动态改变被关联构件的集合和交互关系的性质。

互操作性指的是被连接的构件通过连接件对其他构件进行直接或间接操作的能力。

动态连接性即对连接的动态约束，指连接件对于不同的连接构件实施不同的动态处理方法的能力。

请求响应特性包括响应的并发性、时序性。

在并行或并发系统中，多个构件有可能并行或并发地提出交互请求，这就要求连接件能够正确协调这些交互请求之间的逻辑关系和时序关系。

对于构件而言，连接件是构件的粘合剂，是构件交互的实现。

连接件和构件的区别主要在于它们在体系结构中承担着不同的作用。

连接件也是一组对象。

它把不同的构件连接起来，形成体系结构的一部分，一般表现为框架式对象或转换式对象（调用远程构件资源）。

<<高级软件工程>>

编辑推荐

《高级软件工程》是“十一五”国家重点图书。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>