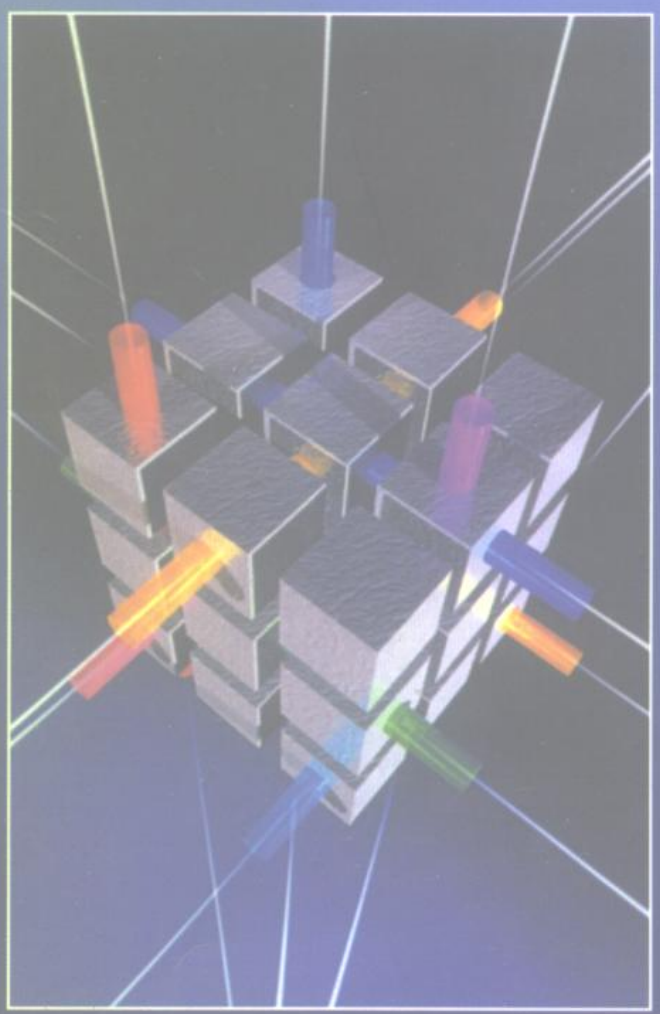
Prentice Hall PTR

北京科海培训中心

CORBA 教程

公共对象请求代理体系结构



[美] R.Otte,P.Patrick,M.Roy 著

李师贤 等译校

清华大学出版社

Prentice Hall PTR　　北京科海培训中心

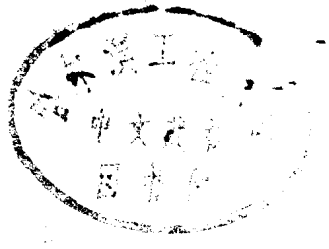
CORBA 教程

公共对象请求代理体系结构

[美] R. Otte, P. Patrick

M. Roy　　著

李师贤　　等译校



清华大学出版社

(京)新登字 158 号

著作权合同登记号:01-1999-2094

内 容 提 要

CORBA 规范是由 OMG(对象管理组)提出的应用软件体系结构和对象技术规范,其核心是一套标准的语言、接口和协议,以支持异构分布应用程序间的互操作性及独立于平台和编程语言的对象重用。

全书共分 4 部分,分别介绍了 CORBA 的基本思想、体系结构以及 CORBA 应用程序的设计、开发与调配。

本书是面向对 CORBA 感兴趣或是想用 CORBA 设计、开发、调配分布式应用程序的读者,也可作为大专院校相关专业的教材或教学参考书。

Understanding CORBA : the common object request broker architecture/Randy Otte, Paul Patrick and Mark Roy

书　名:CORBA 教程:公共对象请求代理体系结构

作　者:R. Otte, P. Patrick, M. Roy

译　者:李师贤等

出版者:清华大学出版社(北京清华大学校内,邮编 100084)

印刷者:北京门头沟胶印厂

发　行:新华书店总店北京科技发行所

开　本:16　印张:10.75　字数:261 千字

版　次:1999 年 10 月第 1 版　　2000 年 1 月第 2 次印刷

印　数:6001～10000

书　号:ISBN 7-302-03745-0/TP・2095

定　价:18.00 元

前 言

本书是面向对 CORBA 一般概念感兴趣或是想用 CORBA 设计、开发、调配分布式应用程序的读者。

如果对 CORBA 的一般概念感兴趣，例如 CORBA 和其他标准的关系、CORBA 如何提出以及 CORBA 的构成，则可以阅读第 1 部分。

如果读者是应用程序的设计者或开发者，并对如何用 CORBA 创建分布式应用程序感兴趣，则可以通读全书。第 2、3 和 4 部分分别讨论了如何设计、构建和调配分布式 CORBA 应用程序。

在写作本书的过程中，我们假定读者没有面向对象编程的经验，不熟悉 CORBA、OMG（对象管理组）的工作或特定软件供应商的 CORBA 系统。

本书使用以下约定：

符号	含义
黑体字	指出将引入一个在术语表中定义的新词
等宽字	指代码片段、例程、操作、结构或属性
…	指如下条件之一： • 省略掉可选的参数 • 前面的一项或多项可以重复一次或多次 • 附加的参数、值或其他可以输入的信息
⋮	指从代码例子或命令行中省略掉的项与要讨论的主题无关
[]	在例子中，指出中括号里的内容是可选的，可以不选择、只选一个或全部都选

本书中的 OMG IDL 和 C 语言源代码例子使用以下约定：

编码约定	描述
/* OMG IDL */或 /* C */注释	代码片段前的注释指出所用的编程语言（OMG IDL 和 C 代码在形式上很相似）
{ }	大括号指出代码段的开始和结束，并用左边缩进表示
模块名 （MODULENAMES）	模块名用大写表示，并不含下划线。在模块名中不包含下划线可以把嵌套的模块名和非嵌套的模块名区分开来。例如，模块 B 包含在模块 A 中，可以命名为 A_B。如果在模块名中包括下划线，嵌套的模块名和非嵌套的模块名就会发生冲突。CORP 是一正确的模块名

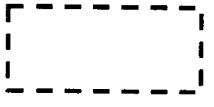
（续表）

编码约定	描述
操作名 (operation_names)	OMG IDL 语言中的操作名以小写表示，且可以在各个单词间含有下划线。例如，promote 是一正确的只含一个单词的操作名
参数名 (argument_names)	参数名以小写表示，且可以用下划线隔开单词。例如 new_job_class 是一正确的参数名
常量和枚举类型成员名	常量和枚举类型成员名以大写字母表示，名字中的各个单词用下划线隔开。例如，HIRED_HOURLY 是一正确的枚举类型成员名
自定义类型名 (UserDefinedTypeNames)	自定义类型名中可以混合大小写字母且不包含下划线。变量名中的各个单词以大写字母开头；单词中的其他字母以小写表示。自定义类型可以用 structures、typedefs、interfaces、attributes 等来创建。例如，DeptInfo 是一正确的自定义类型名
非自定义数据类型名	非自定义数据类型名以小写字母表示，且不用下划线来分隔单词，但可以用空格分隔。例如，unsigned long 是一正确的数据类型名
CORBA 的 C 语言联编 (CORBA_C_bindings)	CORBA 的 C 语言联编中，非自定义的 OMG IDL 数据类型名都有前缀 CORBA_。虽然 CORBA1.1 版本的规格说明中不使用这种约定，但这却是 CORBA1.2 版本的规格说明中的一部分，在本书中也采用这种约定。例如，对应于 C 语言中的 long 数据类型的 CORBA 数据类型是 CORBA_long

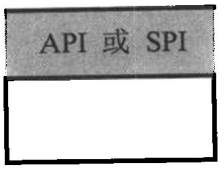
下面是在本书的插图中使用的约定：

指一个计算机系统或一组计算机系统。

指一 CORBA 构件。相互偏离又有重叠的多个方框指该构件可能会有多个拷贝，例如多个客户机应用程序。

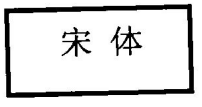


指概念上的构件和接口，这些构件和接口在某种场合下是最终用户提供的，例如 OMGIDL 代码。



指 CORBA 定义的应用程序编程接口(API)或系统编程接口(SPI)。这些接口概念上属于实现了它们或包含了它们的物理构件(例如,对象适配器实现了 BOA API)。

指 CORBA 体系结构定义的构件和接口。



指不是由 CORBA 体系结构定义的构件和接口,但它们和 CORBA 体系结构相关。

以下可能是对读者有用的有关书目,它们按所涉及的主题分组:

- 分布式计算
  - Distributed Computing Implementation and Management Strategies
    作者 Roman Khanna,Prentice-Hall 出版(ISBN:0-13-220138-0)
- CORBA 和对象管理组(OMG)
  - Object Management Architecture Guide by the OMG
  - The Common Object Request Broker: Architecture and Specification by the OMG
  - Common Object Services Specification , Volume I by the OMG
    (ISBN 0-471-07684-8)
    读者可以通过 FTP 在因特网上从 OMG 处获得规范的拷贝,网址是:
    ftp.omg.org 。
- 分布式计算环境(DCE)
  - Understanding DCE ,作者 Ward Rosenberry、David Kenney 和 Gerry Fisher,O'Reilly&Associates,Inc 出版。(ISBN:1-56592-005-8)
  - Guide to Writing DCE Applications ,作者 John Shirley,O'Reilly&Associates,Inc 出版。(ISBN: 1-56592-004-X)
- 面向对象分析与设计
  - Object-Oriented Analysis and Design with Applications ,作者 Grady Booch,Benjamin/Cummings Publishing Co. 出版。(ISBN:0805353402)
  - Object-Oriented Analysis and Design ,作者 James Martin 和 James J. Odell,Prentice-Hall 出版。(ISBN: 0-13-630245-9)
  - Object-Oriented Software Construction ,作者 bertrand Meyer,Prentice-Hall 出版。(ISBN: 0-13-629049-3)
  - Designing Object-Oriented Software,作者 Rebecca Wirfs-Brock、Brian Wilkerson 和 Lauren Weiner,Prentice-Hall 出版。(ISBN: 0-13-629825-7)

- 数字设备公司的对象代理器产品，它实现了 CORBA
  - — ObjectBroker Overview and Glossary
  - — ObjectBroker Installation and Configuration Guide
  - — ObjectBroker System Integrator′s Guide
  - — ObjectBroker Reference Manual

  这些文档可通过数字设备公司获取（电话：1-800-DIGITAL）。

致　谢

离开了众人的工作和支持，这本书不可能完成的，这些人当中包括我们的亲友，尽管他们对于我们能否完成本书抱有疑问。

首先，我们最感谢的是来自数字设备公司的 Steve Baron 和 Pat Carney，他们给了我们充足的时间来完成这个项目，另外还有 Prentice Hall 的员工，他们十分乐于出版我们的著作。

特别感谢 Pat Srite，他完成了本书最后阶段的工作，并把它改编成现在 Prentice Hall 出版的形式。

另外，我们还要感谢以下人员，他们或是抽出时间阅读了本书，或是以其他形式给予了帮助：Neal Jacobson，Dr. Daniel Frantz，Paul Reilly，Mary Jane Grinham，Gray Schmitt，Evelyn Mckay，Susan Hunziker 和 Michelle Chambers。

译校者的话

本书的第 1 章至第 5 章由王迪和马壮展翻译，第 6 至第 8 章由马壮展和叶文超翻译，第 9 章至第 11 章及所有附录和术语表由吴涛和王迪翻译。李师贤对全书进行了校译。由于译校者水平所限，译文中难免有不妥之处，恳请读者批评指正。

目 录

前言 …… (1)

第 1 部分 CORBA 简介

第 1 章 为什么使用 CORBA …… **(1)**

1.1 什么是 CORBA …… (2)
1.2 CORBA 的用途 …… (2)
1.3 基于分布式对象计算的 CORBA …… (3)
1.4 什么是分布式计算 …… (3)
1.4.1 分布式计算的优点 …… (4)
1.4.2 分布式计算的现有机制 …… (4)
1.4.3 CORBA 如何增强分布式计算 …… (4)
1.5 什么是对象模型 …… (7)
1.5.1 使用对象模型的优点 …… (8)
1.5.2 CORBA 结合对象模型的优点 …… (8)
1.6 分布式计算与对象模型如何互相补充 …… (9)
1.7 CORBA——通信中间件 …… (10)
1.8 CORBA 与分布式计算环境(DCE) …… (10)
1.8.1 CORBA 与 DCE 分布式模型的比较 …… (11)
1.8.2 所支持的通信形式的比较 …… (11)
1.8.3 差异小结 …… (11)
1.9 OMG 对象管理体系 …… (12)
1.10 CORBA 的现状 …… (13)
1.11 CORBA 的前景 …… (13)

第 2 章 CORBA 概念总览 …… **(15)**

2.1 CORBA 正式分离客户机和服务器 …… (15)
2.2 请求 …… (15)
2.3 OMG 接口定义语言 …… (16)
2.4 OMG IDL 文件 …… (16)
2.5 对象实例和引用 …… (17)
2.6 对象的实现 …… (17)
2.7 异常 …… (18)
2.8 CORBA 示例 …… (18)
2.9 小结 …… (19)
2.10 下一步内容 …… (20)

第 3 章 CORBA 体系结构综述 …… (21)

3.1 CORBA 体系结构概貌 …… (21)
3.2 客户机应用程序 …… (21)
3.3 桩类型激发 …… (25)
3.4 动态激发 …… (25)
3.5 ORB …… (26)
3.6 上下文对象 …… (27)
3.7 接口仓库 …… (28)
3.8 OMG IDL …… (28)
3.9 OMG IDL 文件和编译器 …… (29)
3.10 服务器应用程序 …… (30)
3.11 对象适配器 …… (31)
3.11.1 基本对象适配器 …… (31)
3.11.2 BOA 操作 …… (32)
3.12 服务器框架 …… (33)
3.13 实现仓库 …… (33)
3.14 下一步内容 …… (33)

第 2 部分 CORBA 应用程序框架设计

第 4 章 设计 CORBA 应用程序 …… (34)

4.1 应用程序框架 …… (34)
4.1.1 使用框架 …… (34)
4.1.2 设计框架 …… (35)
4.2 把分布式应用程序设计成框架 …… (35)
4.3 模型设计 …… (36)
4.3.1 框架示例 …… (37)
4.3.2 第一步:标识要解决的问题 …… (37)
4.3.3 第二步:建立抽象模型 …… (38)
4.3.4 第三步:建立对象模型 …… (39)
4.3.5 第四步:用 OMG IDL 代码表示对象模型 …… (40)
4.4 面向对象模型与传统设计模型的比较 …… (40)

第 5 章 细化对象模型 …… (42)

5.1 使用现实世界模型 …… (42)
5.2 使操作一般化 …… (42)
5.3 决定如何创建对象 …… (43)
5.4 使用接口继承 …… (44)
5.4.1 使用接口继承的一个例子 …… (44)
5.4.2 使用多继承 …… (44)
5.5 有效地使用操作和属性 …… (45)

5.5.1 什么是对象属性 …… (45)
5.5.2 何时使用对象属性或操作 …… (46)
5.5.3 何时使用只读对象属性 …… (47)

第 6 章 分布式设计 …… (48)

6.1 细化分布式环境模型 …… (48)
6.1.1 远程操作 …… (48)
6.1.2 加入接口抽象层 …… (48)
6.1.3 把用户交互从数据交互中分离出来 …… (49)
6.1.4 增加有用的函数 …… (50)
6.2 选择一个交互模型 …… (50)
6.2.1 经典交互模型 …… (51)
6.2.2 对等交互模型 …… (51)
6.2.3 代理器交互模型 …… (52)
6.3 交互模型的通用应用程序 …… (55)
6.4 在框架中使用交互模型 …… (56)
6.5 决定如何包装应用程序 …… (56)

第 7 章 用 OMG IDL 编码 …… (58)

7.1 编写 OMG IDL 代码的基本原则 …… (58)
7.2 用 OMG IDL 定义对象属性 …… (60)
7.2.1 声明对象属性 …… (60)
7.2.2 声明只读对象属性 …… (61)
7.3 用 OMG IDL 定义继承 …… (62)
7.3.1 声明单继承 …… (62)
7.3.2 声明多继承 …… (64)
7.3.3 声明跨模块继承 …… (65)
7.4 用 OMG IDL 定义异常 …… (67)
7.4.1 使用标准异常 …… (68)
7.4.2 定义自定义异常 …… (68)
7.5 用 OMG IDL 定义传播的上下文 …… (69)
7.6 其他 …… (71)

第 3 部分 开发 CORBA 应用程序框架

第 8 章 开发应用程序的客户机部分 …… (72)

8.1 请求的结构 …… (73)
8.1.1 请求的 C 语言映射 …… (73)
8.2 选择激发的通信样式 …… (74)
8.2.1 同步通信 …… (74)
8.2.2 延迟同步通信 …… (75)
8.2.3 单向通信 …… (75)

8.3 选择激发类型 …… (76)
8.3.1 桩类型激发 …… (76)
8.3.2 动态激发 …… (80)
8.4 建立 CORBA 客户机应用程序的一般过程 …… (86)
8.5 开发一个简单的客户机应用程序 …… (86)
8.5.1 支持桩类型激发 …… (87)
8.5.2 对客户桩的进一步研究 …… (88)
8.5.3 请求的格式 …… (90)
8.5.4 请求的结果 …… (90)
8.5.5 请求的对象引用 …… (91)
8.5.6 处理错误和异常 …… (92)
8.6 开发复杂的客户机应用程序 …… (99)
8.6.1 使用上下文对象 …… (100)
8.6.2 使用动态激发 …… (102)
8.7 完整的客户机 C 代码示例 …… (105)

第 9 章 开发应用程序的服务器端部分 …… (110)

9.1 使用 BOA …… (110)
9.2 CORBA 对象的生命周期 …… (111)
9.3 构造 CORBA 服务器的一般过程 …… (112)
9.4 服务器编程 …… (114)
9.4.1 决定实现的激活策略 …… (114)
9.4.2 生成与链接服务器框架 …… (117)
9.4.3 编写代码以启动和终止服务器 …… (118)
9.4.4 编写代码以激活和冻结实现 …… (120)
9.4.5 将客户机请求分发到实现方法 …… (120)
9.4.6 方法编程 …… (121)
9.4.7 创建初始对象引用 …… (126)

第 10 章 客户机操作与服务器方法的关联 …… (127)

10.1 CORBA_BOA_create 操作 …… (127)
10.2 供应商如何把接口映射到实现 …… (128)
10.2.1 简单情形:一个接口对应一个实现 …… (128)
10.2.2 更灵活的情形:一个接口对应多个实现中的一个 …… (129)
10.2.3 最灵活的情形:一个接口对应多个实现 …… (129)
10.3 关联何时发生 …… (130)

第 4 部分 CORBA 应用程序调配到 CORBA 系统

第 11 章 调配应用程序 …… (132)

11.1 总体的调配考虑 …… (132)
11.1.1 决定支持哪种客户机和服务器平台 …… (132)

11.1.2 决定如何包装套件 …… (132)
11.1.3 安装过程提示 …… (133)
11.1.4 调配框架时的激发问题 …… (133)
11.2 调配基于CORBA的客户机 …… (134)
11.2.1 为客户机获取对象引用 …… (134)
11.2.2 调配接口定义 …… (134)
11.2.3 选择用以调配客户桩的格式 …… (134)
11.3 调配基于CORBA的服务器 …… (134)
11.3.1 用服务器创建和分布对象 …… (135)
11.3.2 调配实现定义 …… (135)
11.3.3 调配持久服务器 …… (135)

附录A 扩展代码的例子 …… (136)

A.1 模块CORP.IDL中的人事对象和操作的OMG IDL例子 …… (136)
A.2 客户机应用程序的C语言代码例子 …… (138)
A.3 客户机应用程序的C++语言代码例子 …… (141)

附录B CORBA版本1.1和1.2中操作和对象的命名 …… (145)

附录C CORBA的标准异常 …… (147)

附录D 动态激发操作的总结 …… (148)

术语表 …… (151)

第 1 部分　CORBA 简介

第 1 部分引入了 CORBA，介绍了它从哪儿来的，对一个软件设计者或开发人员来说它有什么优点。这一部分包括以下章节：

- 第 1 章介绍了 CORBA，描述它解决什么问题及它如何把分布式计算和对象模型结合起来。
- 第 2 章阐述了 CORBA 的主要概念，包括术语的定义和相关的面向对象概念。
- 第 3 章提供了 CORBA 系统体系结构的概况，包括体系结构图和对 CORBA 构件(component)与接口(interface)的解释。

第 1 章　为什么使用 CORBA

在今天的企业界，用户比以往任何时候都需要在整个企业内共享信息。共享信息对不同的企业有不同的原因，就如同他们为什么使用计算机一样。共享的信息来自不同的地方。目前，有数以千计的独立的应用程序位于数十种计算机软硬件配置上，而这些软硬件很少是为共享信息或与同一平台上的其他应用程序通信而设计的，更不用说与那些不同平台上的应用程序通信了。即便某个应用程序的确可与其他应用程序共享信息，它一般也是与指定的少量应用程序通信。

把这些应用程序和系统集成起来不是件容易的事情，不同制造商的计算机使用的数据格式通常是不一样的(16 位、32 位或 64 位)。另外，字节的顺序在不同的系统间也是不一样的，要在不同的计算机系统之间共享数据则需要特殊的转换器。

把运行在各种软硬件配置上的应用程序结合起来通常需要一个定制的解决方案。这个解决方案可能是既花时间又费金钱，因为有许多不同的应用程序要连接。而且，如果半年后又有另一个应用程序或平台要加入这个定制的环境，那么集成这个应用程序或平台可能需要花费更多的金钱和时间。

这些系统的用户每天都要同他们计算环境的限制问题打交道。下面列出几个典型的示例说明由于应用程序和系统缺乏集成度而引起的问题：

- 某个工程公司的信息来源不同。该工程公司的广告部使用图形应用程序，财务部用 PC 电子数据表，工程部用基于工作站的 CAD/CAM 应用程序，而该公司必须共享这些部门的不同信息。经理们当然想在他们喜欢的 PC 电子数据表上工作，并且能简易地通过网络获得他们想要的数据，包括历史数据和当前数据。
- 某个大型制造公司有不同年代生产的系统。该公司拥有通过局域网(LANs)和广域网(WANs)连起来的大型机、小型机和 PC 机，以及几种数据库管理系统。同时还有

在不同平台上运行的各种应用程序，如在Uinx系统上的CAD/CAM程序，在OpenVMS系统上的工作流程控制应用程序和在Microsoft Windows系统上的产品信息应用程序。该公司想要把这些不同的工作单元集成起来，以便他们能把现有的软硬件作为下一个业务问题解决方案的一部分。由于新的业务需求、新技术的引进、组织变动等原因，软件环境是会经常变动的。

- 一个小的城市银行把它的系统和一个大的银行系统合并起来，两个银行的合并给大银行留下许多信息存取的问题。例如，出纳员做一些简单的工作，诸如要得到一个顾客的多个帐户的平衡预算，都要与多个应用程序和数据库打交道，这样顾客为了得到信息就要等很长的时间。该银行想把现有的系统和应用程序集成起来，以便员工能快速有效地访问多个来源的信息。该银行还希望建立的新应用程序越简单越好，并且具有可扩展性，以使将来无须重写这些应用程序。
- 一个投资公司必须以多个来源的不断变化的信息进行工作。该投资公司每天得处理证券价格、世界汇率、历史价格信息、市场和业界动态等实时信息，以及公司现在正处理的各种业务信息。所有这些信息都存储在大型机上的关系数据库中。该公司所用的软件来自十多个供应商，并运行在各种硬件平台上。财务计划者要花许多时间来为报表和顾客合同存取和合并数据。

CORBA(the Common Object Request Broker Architecture，公共对象请求代理体系结构)可以帮助用户解决这些问题。CORBA既能解决集成传统应用程序和系统时所遇到的问题，也能为动态变化的企业环境提供适应性。

1.1 什么是CORBA

CORBA是一种标准的面向对象应用程序体系规范。CORBA最初是对象管理组(OMG)在1990年11月出版的《对象管理体系指南》中定义的。OMG是一个非盈利性的组织，成立于1989年，现有500多个成员公司。OMG致力于推广在现有技术基础上集成应用程序的面向对象标准。

CORBA发布的定义包含在《公共对象请求代理：体系结构和规范》中。1991年12月出版了该规范的1.1版本，描述如何开发CORBA实现。1.1版本是DEC公司、Hewlett-Packard公司、HyperDesk公司、NCR公司、对象设计公司和SunSoft公司联合开发的，最终由OMG审议并获得通过。

接着推出了CORBA规范的两个升级版本：1.2版本和2.0版本。这两个版本还都在草案阶段，直到写这本书时(1996年，译者注)还未正式出版。OMG预期在1995年6月推出2.0版本。2.0版本将综合1.2版本和2.0版本的变化。这本书是以1.2版本为蓝本的。

可以用ftp获得这两个规范的草案(指导信息详见前言)。

1.2 CORBA的用途

现在企业所面临的主要问题是需要集成多种工作单元，以使企业能用现有软硬件解决当前的或将来遇到的业务问题。这个问题的一个重要部分是用较新的桌面环境集成大型机

上已有的应用程序。重新创建解决方案则太花时间和金钱。

可以用 CORBA 来解决这些问题。通过 CORBA，可以无须与现有硬件、网络和软件结构打交道，就可以把 PC 机及它的应用程序同企业其他部分连起来。CORBA 提供如下能力：

- 存取来自现行桌面应用程序的分布信息和资源。
- 使现有业务数据和系统成为可供利用的网络资源。
- 为某一特定业务用的定制的功能和能力来增强现行桌面工具和应用程序。
- 改变和发展基于网络的系统以反映新的拓扑结构或新资源。

1.3　基于分布式对象计算的 CORBA

在 CORBA 环境中，应用程序的集成是基于面向对象模型的。该模型提供分析、设计和实现技术来构造可扩展的、可重用的软件，而且其开发成本和维护成本比面向功能的软件要低。

CORBA 的实现提供了开发下一代软件的基础，即开发可重用软件，就如同我们插上或拔下硬盘设备和内存条一样，我们差不多也可以插上和拔下网络、类库和应用程序。通过应用类似于实现了 CORBA 的分布式面向对象系统和其他新的面向对象技术，如 C＋＋类库，现在已经有了商业化的可重用软件。

CORBA 通过分布式对象计算，即分布式计算和面向对象计算的结合，以实现软件重用。分布式对象计算有两个重要组成部分：

- 分布式计算和对象模型的结合

  CORBA 是这两者的完美结合，这两部分不仅带来了自身的优点，而且还完善了对方的优点。CORBA 使应用程序能共享和访问其他应用程序的对象，效果是使这些对象对于所有实现了 CORBA 的应用程序来说都是一样的。
- 代理器的使用

  CORBA 使用代理器，或称中介，来处理系统中客户机与服务器间的消息（称为请求）。代理器能选中一个最符合客户机请求的服务器，并把客户机所看到的接口从服务器的实现中分离出来（实现是指实际完成客户机对一个对象上某个操作的请求的软件）。接口与实现分离的好处是可以采用灵活的、积木式的开发方法，实现对客户机隐藏服务器的变化。只要接口及其行为没变，就可以构造一个新的服务器或修改已有的服务器而无需改变客户机。

以下几节将更详细地描述分布式计算、对象模型、代理器等概念，以及它们在 CORBA 中的应用。

1.4　什么是分布式计算

简单地说，分布式计算是两个或多个软件互相共享信息。这些软件既可以在同一台机器上运行，也可以在通过网络连起来的几台不同机器上运行。绝大多数的分布式计算是基于客户机/服务器模型的。在客户机/服务器模型内，有两类主要的软件：客户机软件，它提出信息