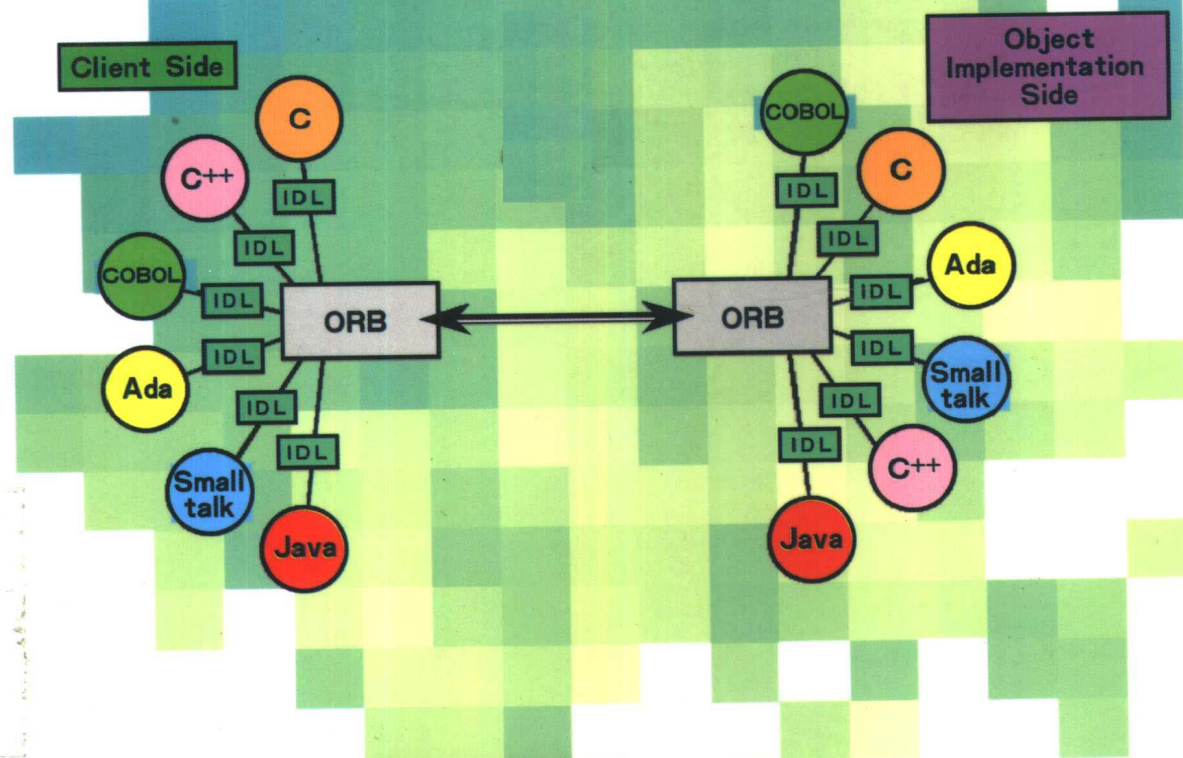


CORBA 语言映射

[美] OMG 编著

信息产业部电信研究院
朗讯科技贝尔实验室
北京邮电大学

韦乐平
薛君敖 主持编译
孟洛明



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

URL: <http://www.phei.com.cn>

CORBA 技术系列丛书

CORBA 语言映射

CORBA Language Mapping

[美] OMG 编著

信息产业部电信研究院	韦乐平	
朗讯科技贝尔实验室	薛君教	主持编译
北京邮电大学	孟洛明	

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书是《CORBA 技术系列丛书》的第二册。根据 OMG 1999 年 6 月版《CORBA Language Mapping》，由信息产业部电信研究院、北京邮电大学和朗讯科技贝尔实验室的多位专家、教授和工程师共同编译而成。内容包括 IDL-Java 语言映射、Java-IDL 语言映射、IDL-C++ 语言映射、IDL-C 语言映射、IDL-COBOL 语言映射、IDL-Ada 语言映射和 IDL-Smalltalk 语言映射。中国工程院院士顾冠群为本书作序。

本书可作为高等院校通信与计算机相关专业的教学参考书，以及相关专业软件工程师的实用参考书。



本书中文专有翻译出版版权由美国 OMG (Object Management Group) 授予信息产业部电信研究院。信息产业部电信研究院授予电子工业出版社本书的中文专有出版版权。该专有出版版权受法律保护。

图书在版编目 (CIP) 数据

CORBA 语言映射/美国对象管理组织编著; 韦乐平, 薛君敖, 孟洛明译.-北京: 电子工业出版社, 2001.6
(CORBA 技术系列丛书)

ISBN 7-5053-6732-3

I. C… II. ①美… ②韦… ③薛… ④孟… III. 分布式处理系统-应用软件, CORBA IV. TP316.4

中国版本图书馆 CIP 数据核字 (2001) 第 029788 号

丛 书 名: CORBA 技术系列丛书

书 名: CORBA 语言映射

原 书 名: CORBA Language Mapping

编 著 者: [美] OMG

主持编译: 韦乐平 薛君敖 孟洛明

责任编辑: 张 毅 zhangyi@phei.com.cn

特约编辑: 秋 阳

印 刷 者: 北京天宇星印刷厂

装 订 者: 河北省涿州桃园装订厂

出版发行: 电子工业出版社 URL: <http://www.phei.com.cn>

北京市海淀区万寿路 173 信箱 邮编: 100036

经 销: 各地新华书店

开 本: 787×1092 1/16 印张: 37.5 字数: 960 千字

版 次: 2001 年 6 月第 1 版 2001 年 6 月第 1 次印刷

书 号: ISBN 7-5053-6732-3
TP · 3764

印 数: 3000 册 定价: 88.00 元

版权贸易合同登记号 图字: 01-2000-2759 号

凡购买电子工业出版社的图书, 如有缺页、倒页、脱页问题者, 请向购买书店调换。

若书店售缺, 请与本社发行部联系调换, 电话: (010) 68279077。

序

分布式计算系统，在基于网络的信息系统中处于中间层，从功能上看，向下它可以传递和处理上层用户的各种请求，向上可以屏蔽下层的实现细节，提供各种增值服务。因此，分布式计算起到承上启下的作用，为用户构造基于网络的分布应用提供强有力的支持工具。

网络技术和应用的发展，与分布式计算技术密切相关。对象技术的发展和成熟，给分布式计算技术带来了新的发展方向，为异构系统的集成创立了新的支撑。

美国的对象管理组织（OMG, Object Management Group）于 1991 年制定了 CORBA（Common Object Request Broker Architecture, 公共对象请求代理结构）规范。CORBA 技术的优点，是既能够解决异构系统带来的“孤岛”问题，又能扩展客户/服务器模式，使得系统具有良好的缩放性，便于系统的开发和升级。目前，国际范围内已经有大量的企业、电信部门、政府机构和银行等领域运用了 CORBA 技术，取得了显著的效益。在国内，CORBA 研究和应用也在深入，也已经有了国产化的实现软件。随着在 CORBA 技术中引入组件技术和实时技术，CORBA 技术的优势将得到进一步增强。

本书将 OMG 制定的 CORBA IDL 语言映射部分的标准译成中文，使我国读者能系统、全面地了解和研究 CORBA 技术，这对于我国网络软件和分布式计算系统的开发将起到很好的促进作用。

本书对于高等学校计算机及通信专业的本科生、研究生以及相关领域的科技人员具有很好的参考价值。

祝贺本书的成功出版。

中国工程院院士

东南大学校长

顾冠群

2001 年 5 月

译者序

随着当今计算机技术的不断发展, CORBA 技术也在不断完善之中。

在 2001 年 1 月 15 日 SG 日内瓦会议上, ITU-T 的 CORBA 框架结构和 IDL 信息模型得到了通过。这意味着 CORBA 技术作为一种用于网管的标准中间件技术已被 ITU-SG4 采纳。

为了促进 CORBA 技术在国内的推广与应用, 我们根据与 OMG 达成的协议, 将 CORBA 技术中的重要规范译成中文。

本书是根据 OMG 1999 年 6 月版的《CORBA Language Mapping》编译而成, 内容包括 IDL-Java 映射、Java-IDL 映射、IDL-C++映射、IDL-C 映射、IDL-COBOL 映射、IDL-ADA 映射和 IDL-Smalltalk 映射。

本书的编译工作是在信息产业部电信研究院、朗讯科技(中国)贝尔实验室和北京邮电大学组成的编委会领导下进行的。参加本书编译工作的人员有: 信息产业部电信研究院葛彦、刘芮; 北京邮电大学刘封、邱雪松; 朗讯科技(中国)贝尔实验室叶岚、曾萍、刘燕、宋显军、汪智垚、邢江峰、朱于军、马新宇。

朗讯科技(美国)贝尔实验室的薛君敖博士、余爽及朗讯科技(中国)贝尔实验室宋显军对全书进行了最终的审校。在本书的编译过程中, 得到了朗讯科技贝尔实验室王泽霖博士、高培椿博士、叶成博士和朗讯科技深圳研究开发中心技术总监 Sekar Kalambur 和朗讯科技中国亚太区业务发展总监 Roger Hinckley 的具体指导和及时有力的帮助。在此, 对以上各位的支持以及对众多译校者的努力和卓有成效的工作表示衷心的感谢。

本书是《CORBA 技术系列丛书》的第二册, 其他两册为:

- 《CORBA 系统结构、原理与规范》(已于 2000 年 6 月出版)
- 《CORBA 服务》

希望本书的出版对于我国通信及计算机领域推广和提高 CORBA 技术的应用能起到积极的促进作用。

本书可作为大学通信和计算机等专业的教学参考书, 以及相关专业软件工程师的实用参考书。

CORBA 技术还在不断发展, 本书在编译过程中难免会有不足之处, 诚恳期待读者的批评和指正。

薛君敖 余爽 宋显军

朗讯科技贝尔实验室

2001 年 5 月

原 书 序

关于 CORBA 语言映射规范

CORBA 语言映射规范包含了对如下 7 个方面的映射规范：

- IDL-Java
- Java-IDL
- C++
- C
- COBOL
- Ada
- Smalltalk

每一种语言映射专门用一章来阐述。

下表列出了每一种语言和与其相对应的 CORBA 版本：

语 言 映 射	相对应的 CORBA 版本
IDL-Java	CORBA 2.3
Java-IDL	CORBA 2.3
C++	CORBA 2.3
C	CORBA 2.1
COBOL	CORBA 2.2
Ada	CORBA 2.2
Smalltalk	CORBA 2.0

CORBA 一致性的定义

符合 CORBA 一致性的最低要求是遵循 CORBA 核心规范以及一种编程语言映射方法规范。除此之外，每一种附加的编程语言映射方法都可作为一致性声明中的一个可选条目。其中，可选意味着如果在其应用领域中认为没有必要，用户可以不实现该条目，但是，如果用户实现了某个可选条目，该实现必须符合 CORBA 规范中对该条目所定义的规范，才能称之为符合 CORBA 规范。例如，如果一个软件提供商的 CORBA 产品中实现了 IDL 到 C++ 语言的映射，它必须符合本书中定义的 IDL 到 C++ 的联编规范。

互操作 (interoperability) 和互协作 (interworking) 是另外两个独立的一致性声明条目。关于互协作的一致性的详细内容参见《CORBA 系统结构、原理与规范》17.10 节“COM/CORBA 互协作的一致性”。

正如在《对象管理体系结构指南》中介绍的那样, OMG 的核心对象模型 (Core Object Model) 是由 ORB 核心和基于这一核心的应用单元组成的。与此相对应, CORBA 规范也可以分为两部分: ORB 核心规范和应用单元规范。本系列丛书的章节结构反映了这种分法。

CORBA 规范被译成如下三册分别出版。

1. 《CORBA 系统结构、原理与规范》(已于 2000 年 6 月出版)

包括以下内容:

- CORBA 核心 (第 1 章~第 11 章)
- CORBA 互操作性 (第 12 章~第 16 章)
- CORBA 互协作性 (第 17 章~第 21 章)

2. 《CORBA 语言映射》

包括以下内容:

- OMG IDL 到 Java 的映射
- Java 到 OMG IDL 的映射
- OMG IDL 到 C++ 的映射
- OMG IDL 到 C 的映射
- OMG IDL 到 COBOL 的映射
- OMG IDL 到 Ada 的映射
- OMG IDL 到 Smalltalk 的映射

3. 《CORBA 服务》

致谢

以下合作者提交了本规范的部分内容, 并得到 OMG 的同意而成为 CORBA 规范, 在此谨表感谢。

- BEA Systems, Inc.
- BNR Europe Ltd.
- Defense Information Systems Agency
- Expersoft Corporation
- FUJITSU LIMITED
- Genesis Development Corporation
- Gensym Corporation
- IBM Corporation
- ICL plc
- Inprise Corporation
- IONA Technologies Ltd.

- Digital Equipment Corporation
- Hewlett-Packard Company
- HyperDesk Corporation
- Micro Focus Limited
- MITRE Corporation
- NCR Corporation
- Novell USG
- Object Design, Inc.
- Objective Interface Systems, Inc.
- OC Systems, Inc.
- Open Group - Open Software Foundation
- Siemens Nixdorf Informationssysteme AG
- Sun Microsystems Inc.
- SunSoft, Inc.
- Sybase, Inc.
- Telefónica Investigación y Desarrollo S.A. Unipersonal
- Visual Edge Software, Ltd.

同时, OMG特别感谢SGI公司的Mark Linton和纽约州立大学的Doug Lea 在IDL到C++语言映射方面所做的贡献。

参考文献

IDL Type Extensions RFP, March 1995. OMG TC Document 95-1-35

The Common Object Request Broker: Architecture and Specification, Revision 2.2, February 1998

CORBA services: Common Object Services Specification, Revised Edition, OMG TC Document 95-3-31

COBOL Language Mapping RFP, December 1995. OMG TC document 95-12-10

COBOL 85 ANSI X3.23-1985 / ISO 1989-1985

IEEE Standard for Binary Floating-Point Arithmetic, ANIS/IEEE Std 754-1985

XDR: External Data Representation Standard, RFC1832, R. Srinivasan, Sun Microsystems, August 1995

OSF Character and Code Set Registry, OSF DCE SIG RFC 40.1 (Public Version), S. (Martin) O'Donnell, June 1994

RPC Runtime Support For I18N Characters — Functional Specification, OSF DCE SIG RFC 41.2, M. Romagna, R. Mackey, November 1994

X/Open System Interface Definitions, Issue 4 Version 2, 1995

目 录

第 1 章 IDL 到 Java 语言映射	(1)
1.1 概述	(1)
1.1.1 org.omg.*包	(2)
1.1.1.1 允许的修改	(2)
1.2 名字	(3)
1.2.1 保留名字	(3)
1.3 模块的映射	(4)
1.3.1 例子	(4)
1.4 基本类型的映射	(5)
1.4.1 概述	(5)
1.4.1.1 将来的支持	(5)
1.4.1.2 IDLEntity	(6)
1.4.1.3 Java 序列化	(6)
1.4.1.4 Holder 类	(6)
1.4.1.5 使用 Java null	(11)
1.4.2 布尔类型	(11)
1.4.3 字符类型	(12)
1.4.4 Octet 类型	(12)
1.4.5 字符串类型	(12)
1.4.6 整数类型	(12)
1.4.7 浮点数类型	(12)
1.4.8 定点类型	(12)
1.4.9 未来版本的长双精度类型	(13)
1.5 Helper 类	(13)
1.5.1 用于盒值的 Helpers	(13)
1.5.2 除盒值类型外的 Helper 类	(13)
1.5.2.1 值类型生成库的 convenience 方法	(14)
1.5.3 举例	(14)
1.6 常量映射	(15)
1.6.1 接口中的常量	(16)
1.6.1.1 举例	(16)
1.6.2 不在接口中的常量	(16)
1.6.2.1 举例	(16)
1.7 枚举类型的映射	(17)
1.7.1 举例	(18)
1.8 结构类型的映射	(20)
1.8.1 举例	(20)

1.9 联合类型的映射	(21)
1.9.1 举例	(22)
1.10 序列类型的映射	(24)
1.10.1 举例	(24)
1.11 数组类型的映射	(26)
1.11.1 举例	(27)
1.12 接口的映射	(28)
1.12.1 基本概念	(28)
1.12.1.1 举例	(29)
1.12.2 参数传递模式	(32)
1.12.2.1 举例	(33)
1.12.3 操作中的上下文参数	(34)
1.13 值类型的映射	(35)
1.13.1 用于值类型的 Java 接口	(35)
1.13.1.1 ValueBase 的接口	(35)
1.13.1.2 StreamableValue 接口	(36)
1.13.1.3 CustomMarshal 接口	(36)
1.13.1.4 CustomValue 接口	(37)
1.13.1.5 ValueFactory 接口	(37)
1.13.2 Stateful 值类型的基本概念	(37)
1.13.2.1 值类型的继承	(38)
1.13.3 抽象值类型	(39)
1.13.4 CORBA::ValueBase	(39)
1.13.5 例 A	(39)
1.13.6 例 B	(41)
1.13.7 参数传递模式	(44)
1.13.7.1 举例	(45)
1.13.8 值生成库和封装	(46)
1.14 盒值类型	(47)
1.14.1 一般的盒值 Helper 接口	(47)
1.14.2 映射为基本类型的盒值	(47)
1.14.2.1 基本类型的例子	(49)
1.14.3 复杂类型	(51)
1.14.3.1 复杂类型的例子	(51)
1.15 异常的映射	(53)
1.15.1 用户定义的异常	(54)
1.15.1.1 举例	(54)
1.15.1.2 未知的用户异常	(55)
1.15.2 系统异常	(56)
1.15.3 间接异常	(58)
1.16 Any 类型的映射	(59)
1.17 特定嵌套类型的映射	(64)

1.17.1 举例	(64)
1.18 typedef 类型的映射	(64)
1.18.1 简单的 IDL 类型	(64)
1.18.2 复杂的 IDL 类型	(65)
1.18.2.1 举例	(65)
1.19 伪对象到 Java 的映射	(65)
1.19.1 简介	(65)
1.19.1.1 伪接口	(66)
1.19.2 特定异常	(66)
1.19.3 Environment	(67)
1.19.4 NamedValue	(67)
1.19.5 NVList	(68)
1.19.6 ExceptionList	(69)
1.19.7 上下文	(70)
1.19.8 Request	(71)
1.19.9 TypeCode	(72)
1.19.10 ORB	(78)
1.19.10.1 set_delegate	(88)
1.19.10.2 get_value_def	(88)
1.19.11 CORBA::Object	(88)
1.19.12 Principal	(90)
1.20 服务器方的映射	(91)
1.20.1 概述	(91)
1.20.2 实现接口	(91)
1.20.2.1 PortableServer::Servant 的映射	(91)
1.20.2.2 动态程序框架接口的映射	(95)
1.20.2.3 程序框架的可移植性	(97)
1.20.2.4 程序框架的操作	(97)
1.20.2.5 基于继承的接口实现	(97)
1.20.2.6 基于授权的接口的实现	(98)
1.20.3 PortableServer::ServantManager 的映射	(101)
1.20.3.1 Cookie 的映射	(101)
1.20.3.2 ServantManagers 和 AdapterActivators	(101)
1.21 Java ORB 可移植性接口	(102)
1.21.1 概述	(102)
1.21.1.1 设计目标	(102)
1.21.2 总体结构	(102)
1.21.2.1 可移植性包	(103)
1.21.3 Streamable API	(103)
1.21.4 Streaming API	(103)
1.21.4.1 输入流方法的语义	(110)
1.21.4.2 输出流方法的语义	(110)

1.21.5	可移植根程序和程序框架接口	(111)
1.21.5.1	根程序/程序框架的结构	(111)
1.21.5.2	根程序和程序框架类的层次结构	(118)
1.21.5.3	可移植的 ObjectImpl	(119)
1.21.5.4	InvokeHandler	(124)
1.21.5.5	Response Handler	(125)
1.21.5.6	ApplicationException	(126)
1.21.5.7	RemarshalException	(126)
1.21.5.8	UnknownException	(126)
1.21.6	授权根程序	(127)
1.21.7	Servant	(130)
1.21.8	仆从的授权	(130)
1.21.9	ORB 初始化	(131)
1.21.9.1	标准特性	(131)
1.21.9.2	orb.properties 文件	(131)
1.21.9.3	ORB 初始化方法	(131)
第 2 章	Java 语言到 IDL 映射	(135)
2.1	概述	(135)
2.2	Java 的 RMI/IDL 子集	(135)
2.2.1	符合 RMI/IDL 类型一致性的概述	(135)
2.2.2	基本类型	(136)
2.2.3	RMI/IDL 远程接口	(136)
2.2.3.1	根程序和远程实现类	(137)
2.2.4	RMI/IDL 值类型	(137)
2.2.4.1	Java String 类型	(138)
2.2.5	RMI/IDL 数组	(138)
2.2.6	RMI/IDL 异常类型	(138)
2.2.7	CORBA 对象引用类型	(139)
2.2.8	IDL 实体类型	(139)
2.3	IDL 映射	(139)
2.3.1	概述	(139)
2.3.1.1	特殊大小写映射总结	(140)
2.3.2	Java 名字到 IDL 名字的映射	(140)
2.3.2.1	将包映射为模块	(140)
2.3.2.2	与 IDL 关键字冲突的 Java 名	(140)
2.3.2.3	有下划线的 Java 名	(141)
2.3.2.4	具有非法 IDL 标识字符的 Java 名	(141)
2.3.2.5	内部类的命名	(141)
2.3.2.6	重载方法名	(141)
2.3.2.7	仅大小写不同的名字	(142)
2.3.2.8	与其他名字冲突的方法名	(142)

2.3.2.9	引起 OMG IDL 名字冲突的命名	(142)
2.3.3	基本类型的映射	(142)
2.3.4	RMI/IDL 远程接口的映射	(143)
2.3.4.1	java.rmi.Remote 的特例	(143)
2.3.4.2	被继承接口	(144)
2.3.4.3	属性存取器方法	(144)
2.3.4.4	方法	(145)
2.3.4.5	常量	(145)
2.3.4.6	库 ID	(145)
2.3.4.7	举例	(145)
2.3.5	RMI/IDL 值类型的映射	(146)
2.3.5.1	继承基类	(147)
2.3.5.2	继承接口	(147)
2.3.5.3	方法	(147)
2.3.5.4	构造方法	(147)
2.3.5.5	常量	(148)
2.3.5.6	数据	(148)
2.3.5.7	库 ID	(149)
2.3.5.8	无 writeObject 的例子	(149)
2.3.5.9	有 writeObject 的例子	(150)
2.3.5.10	映射到 java.lang.String	(151)
2.3.5.11	映射到 java.lang.Class	(151)
2.3.6	RMI/IDL 数组的映射	(151)
2.3.6.1	防止盒化序列类型的重新定义	(152)
2.3.6.2	数组实例	(152)
2.3.7	RMI/IDL 异常的映射	(153)
2.3.7.1	IDL 值类型	(154)
2.3.7.2	IDL 异常	(154)
2.3.7.3	引用到 RMI/IDL 异常的映射	(154)
2.3.7.4	举例	(154)
2.3.8	CORBA 对象引用类型的映射	(155)
2.3.9	IDL 实体类型的映射	(156)
2.3.10	不符合一致性的类和接口的映射	(157)
2.3.10.1	java.io.Serializable 和 java.io. Externalizable	(157)
2.3.10.2	java.lang.Object 映射	(158)
2.3.10.3	继承接口	(158)
2.3.10.4	方法和常量	(158)
2.3.10.5	举例	(158)
2.3.11	抽象接口映射	(159)
2.3.11.1	继承接口	(159)
2.3.11.2	方法和常量	(159)
2.3.11.3	举例	(159)

2.3.12	映射实现类	(160)
2.3.12.1	举例	(160)
2.4	实时运行的问题	(161)
2.4.1	值对象的子类	(161)
2.4.2	远程引用根程序的定位	(161)
2.4.3	限定	(161)
2.4.4	为远程值分配 Tie	(162)
2.4.5	超大字符支持	(162)
2.4.6	根程序和 Tie 的定位	(162)
2.4.7	RMI 异常到 CORBA 系统异常的映射	(163)
2.4.8	CORBA 系统异常到 RMI 异常的映射	(163)
2.4.8.1	UnknownExceptionInfo 服务上下文的映射	(164)
2.4.9	代码下载	(164)
2.4.9.1	定义	(164)
2.4.9.2	Codebase 的选择	(165)
2.4.9.3	Codebase 的传输	(165)
2.4.9.4	codebase 的存取	(165)
2.4.9.5	codebase 的用法	(166)
2.4.10	定制封装格式	(167)
2.4.11	封装 RMI/IDL 数值	(168)
2.4.12	运行时间限制	(168)
2.4.12.1	共享的引用对象	(168)
2.4.12.2	分布式垃圾收集	(168)
2.4.12.3	限定	(168)
2.5	可移植性接口	(169)
2.5.1	可移植性 API	(169)
2.5.1.1	Tie	(169)
2.5.1.2	Stub	(170)
2.5.1.3	ValueHandler	(171)
2.5.1.4	Util	(174)
2.5.1.5	附加的可移植性 API	(176)
2.5.2	生成类	(176)
2.5.2.1	根程序类	(176)
2.5.2.2	本地根程序	(178)
2.5.2.3	Tie 类	(179)
2.5.3	API 实现的可置换性	(181)
2.5.3.1	StubDelegate	(181)
2.5.3.2	UtilDelegate	(182)
2.5.3.3	PortableRemoteObjectDelegate	(183)
2.5.3.4	指派机制	(184)
2.6	应用程序接口	(184)
2.6.1	PortableRemoteObject	(185)

2.7 生成的 IDL 文件结构	(186)
2.7.1 Java 定义	(189)
2.7.2 生成的 OMG IDL 定义	(189)
第 3 章 IDL 到 C++ 语言映射	(193)
3.1 预备信息	(194)
3.1.1 概述	(194)
3.1.1.1 关键的设计决定	(194)
3.1.1.2 一致性	(194)
3.1.1.3 C++ 实现要求	(194)
3.1.1.4 C 数据格式安排的兼容性	(194)
3.1.1.5 无实现描述	(195)
3.1.2 作用域的名称	(195)
3.1.3 C++ 类型大小的要求	(196)
3.1.4 CORBA 模块	(196)
3.2 模块的映射	(196)
3.3 接口的映射	(197)
3.3.1 对象引用类型	(198)
3.3.2 扩展对象引用	(198)
3.3.3 对象引用操作	(199)
3.3.4 限定对象引用	(200)
3.3.5 Nil 对象引用	(201)
3.3.6 对象引用 Out 参数	(201)
3.3.7 接口映射举例	(202)
3.4 常量的映射	(204)
3.4.1 超大字符和超大字符串常量	(205)
3.4.2 定点常量	(205)
3.5 基本数据类型的映射	(206)
3.6 枚举型的映射	(207)
3.7 字符串类型的映射	(208)
3.8 超大字符串类型的映射	(211)
3.9 结构化类型的映射	(211)
3.9.1 T_var 类型	(212)
3.9.2 T_out 类型	(216)
3.10 结构类型的映射	(217)
3.11 固定类型的映射	(220)
3.11.1 固定的 T_var 和 T_out 类型	(223)
3.12 联合类型的映射	(223)
3.13 序列类型的映射	(228)
3.13.1 序列举例	(232)
3.13.2 使用“release”构造函数参数	(233)
3.13.3 额外的内存管理功能	(234)

3.13.4	T_var 序列和 T_out 类型	(234)
3.14	数组类型的映射	(235)
3.15	typedef 的映射	(237)
3.16	Any 类型的映射	(239)
3.16.1	处理类型的值	(239)
3.16.2	到 Any 的插入值	(239)
3.16.3	从 Any 提取值	(243)
3.16.4	区别 boolean、octet、char、wchar、限定字符串和限定超大字符串	(245)
3.16.5	对象的扩展	(249)
3.16.6	抽象接口的扩展	(250)
3.16.7	处理 Untyped 值	(250)
3.16.8	TypeCode 替换	(251)
3.16.9	Any 构造函数、析构函数和指定操作符	(252)
3.16.10	Any 类	(253)
3.16.11	Any_var 类	(253)
3.17	valuetype 的映射	(253)
3.17.1	valuetype 的数据成员	(254)
3.17.2	构造函数、赋值操作符和析构函数	(256)
3.17.3	valuetype 操作	(256)
3.17.4	valuetype 举例	(257)
3.17.5	ValueBase 和引用计数	(258)
3.17.5.1	CORBA 模块的补充	(260)
3.17.6	引用计数混合类	(260)
3.17.7	值盒	(261)
3.17.7.1	底层盒类型的参数传递	(261)
3.17.7.2	基类、枚举和对象引用	(262)
3.17.7.3	结构类型	(263)
3.17.7.4	字符串和超大字符串类型	(265)
3.17.7.5	联合、序列、固定和 Any 类型	(267)
3.17.7.6	数组类型	(268)
3.17.8	抽象的 valuetype	(270)
3.17.9	valuetype 的继承	(270)
3.17.10	valuetype 生成库	(271)
3.17.10.1	ValueFactoryBase 类	(272)
3.17.10.2	ValueFactoryBase_var 类	(273)
3.17.10.3	类型特定的值生成库	(275)
3.17.10.4	拆包特性	(276)
3.17.11	定制打包	(277)
3.17.12	另一个 valuetype 范例	(277)
3.17.13	valuetype 的结构成员	(278)
3.18	抽象接口的映射	(279)
3.18.1	抽象接口库	(279)

3.18.2 客户端的映射	(280)
3.19 异常类型的映射	(281)
3.19.1 UnknownUserException	(284)
3.19.2 异常的 Any 插入和抽取	(285)
3.20 操作和属性的映射	(285)
3.21 操作的隐式变元	(286)
3.22 变元传递的考虑	(286)
3.22.1 操作参数和参数说明	(290)
3.23 伪对象到 C++ 的映射	(293)
3.24 使用	(294)
3.25 映射规则	(294)
3.26 与 C-PIDL 映射的关系	(295)
3.27 Environment	(296)
3.27.1 Environment 接口	(296)
3.27.2 Environment C++ 类	(296)
3.27.3 与 C-PIDL 的区别	(297)
3.27.4 内存管理	(297)
3.28 NamedValue	(297)
3.28.1 NamedValue 接口	(297)
3.28.2 NamedValue C++ 类	(297)
3.28.3 与 C-PIDL 的区别	(298)
3.28.4 内存管理	(298)
3.29 NVList	(298)
3.29.1 NVList 接口	(299)
3.29.2 NVList C++ 类	(299)
3.29.3 与 C-PIDL 的区别	(300)
3.29.4 内存管理	(300)
3.30 请求	(300)
3.30.1 Request 接口	(302)
3.30.2 Request C++ 类	(303)
3.30.3 与 C-PIDL 的区别	(304)
3.30.4 内存管理	(305)
3.31 上下文	(305)
3.31.1 Context 接口	(305)
3.31.2 Context C++ 类	(306)
3.31.3 与 C-PIDL 的区别	(306)
3.31.4 内存管理	(306)
3.32 TypeCode	(307)
3.32.1 TypeCode 接口	(307)
3.32.2 TypeCode C++ 类	(307)
3.32.3 与 C-PIDL 的区别	(308)
3.32.4 内存管理	(308)