

0.1	About This Document	xxiii
0.1.1	Object Management Group.	xxiii
0.1.2	X/Open	xxiv
0.2	Intended Audience	xxiv
0.3	Context of CORBA	xxiv
0.4	Associated Documents	xxv
0.5	Definition of CORBA Compliance.	xxvi
0.6	Structure of This Manual	xxvii
0.7	Acknowledgements	xxix
0.8	References	xxx
1.	The Object Model	1-1
1.1	Overview	1-1
1.2	Object Semantics	1-2
1.2.1	Objects	1-3
1.2.2	Requests	1-3
1.2.3	Object Creation and Destruction	1-4
1.2.4	Types	1-4
1.2.5	Interfaces	1-5
1.2.6	Operations	1-6
1.2.7	Attributes	1-7
1.3	Object Implementation.	1-8
1.3.1	The Execution Model: Performing Services. . .	1-8
1.3.2	The Construction Model.	1-8
2.	CORBA Overview	2-1
2.1	Structure of an Object Request Broker.	2-2
2.1.1	Object Request Broker	2-6
2.1.2	Clients	2-7
2.1.3	Object Implementations	2-7
2.1.4	Object References.	2-8
2.1.5	OMG Interface Definition Language	2-8
2.1.6	Mapping of OMG IDL to Programming Languages	2-8
2.1.7	Client Stubs	2-9
2.1.8	Dynamic Invocation Interface.	2-9
2.1.9	Implementation Skeleton	2-9
2.1.10	Dynamic Skeleton Interface	2-10
2.1.11	Object Adapters	2-10
2.1.12	ORB Interface.	2-10

2.1.13	Interface Repository	2-11
2.1.14	Implementation Repository	2-11
2.2	Example ORBs.	2-11
2.2.1	Client- and Implementation-resident ORB	2-11
2.2.2	Server-based ORB	2-12
2.2.3	System-based ORB.	2-12
2.2.4	Library-based ORB	2-12
2.3	Structure of a Client	2-12
2.4	Structure of an Object Implementation	2-13
2.5	Structure of an Object Adapter.	2-15
2.6	CORBA Required Object Adapter.	2-17
2.6.1	Portable Object Adapter	2-17
2.7	The Integration of Foreign Object Systems	2-18
3.	OMG IDL Syntax and Semantics.	3-1
3.1	Overview	3-2
3.2	Lexical Conventions.	3-3
3.2.1	Tokens	3-6
3.2.2	Comments.	3-6
3.2.3	Identifiers	3-6
3.2.4	Keywords	3-7
3.2.5	Literals	3-7
3.3	Preprocessing	3-9
3.4	OMG IDL Grammar.	3-10
3.5	OMG IDL Specification.	3-14
3.5.1	Module Declaration	3-14
3.5.2	Interface Declaration	3-15
3.6	Inheritance	3-16
3.7	Constant Declaration	3-18
3.7.1	Syntax.	3-18
3.7.2	Semantics	3-20
3.8	Type Declaration	3-22
3.8.1	Basic Types.	3-23
3.8.2	Constructed Types	3-25
3.8.3	Template Types.	3-27
3.8.4	Complex Declarator	3-29
3.8.5	Native Types.	3-29
3.9	Exception Declaration	3-30
3.10	Operation Declaration	3-31

3.10.1	Operation Attribute.....	3-31
3.10.2	Parameter Declarations.....	3-32
3.10.3	Raises Expressions.....	3-32
3.10.4	Context Expressions.....	3-33
3.11	Attribute Declaration.....	3-33
3.12	CORBA Module.....	3-34
3.13	Names and Scoping.....	3-35
3.14	Differences from C++.....	3-37
3.15	Standard Exceptions.....	3-37
3.15.1	Standard Exceptions Definitions.....	3-38
3.15.2	Object Non-Existence.....	3-39
3.15.3	Transaction Exceptions.....	3-39
4.	ORB Interface.....	4-1
4.1	Overview.....	4-1
4.1.1	Converting Object References to Strings.....	4-3
4.1.2	Getting Service Information.....	4-3
4.2	Object Reference Operations.....	4-4
4.2.1	Determining the Object Interface.....	4-4
4.2.2	Duplicating and Releasing Copies of Object References	4-5
4.2.3	Nil Object References.....	4-5
4.2.4	Equivalence Checking Operation.....	4-5
4.2.5	Probing for Object Non-Existence.....	4-6
4.2.6	Object Reference Identity.....	4-6
4.2.7	Getting Policy Associated with the Object.....	4-7
4.2.8	Getting the Domain Managers Associated with the Object	4-8
4.3	ORB and OA Initialization and Initial References.....	4-8
4.4	ORB Initialization.....	4-8
4.5	Obtaining Initial Object References.....	4-10
4.6	Current Object.....	4-12
4.7	Policy Object.....	4-12
4.8	Management of Policy Domains.....	4-14
4.8.1	Basic Concepts.....	4-14
4.8.2	Domain Management Operations.....	4-16
4.9	Thread-related operations.....	4-19
4.9.1	work_pending.....	4-19
4.9.2	perform_work.....	4-19

4.9.3	run	4-20
4.9.4	shutdown	4-20
5.	Dynamic Invocation Interface	5-1
5.1	Overview	5-2
5.1.1	Common Data Structures	5-2
5.1.2	Memory Usage	5-4
5.1.3	Return Status and Exceptions	5-4
5.2	Request Operations	5-5
5.2.1	create_request	5-5
5.2.2	add_arg	5-7
5.2.3	invoke	5-8
5.2.4	delete	5-8
5.3	Deferred Synchronous Operations	5-8
5.3.1	send	5-8
5.3.2	send_multiple_requests	5-9
5.3.3	get_response	5-10
5.3.4	get_next_response	5-10
5.4	List Operations	5-11
5.4.1	create_list	5-12
5.4.2	add_item	5-12
5.4.3	free	5-12
5.4.4	free_memory	5-13
5.4.5	get_count	5-13
5.4.6	create_operation_list	5-13
5.5	Context Objects	5-13
5.6	Context Object Operations	5-14
5.6.1	get_default_context	5-15
5.6.2	set_one_value	5-16
5.6.3	set_values	5-16
5.6.4	get_values	5-16
5.6.5	delete_values	5-17
5.6.6	create_child	5-17
5.6.7	delete	5-17
5.7	Native Data Manipulation	5-17
6.	Dynamic Skeleton Interface	6-1
6.1	Introduction	6-1
6.2	Overview	6-2
6.3	ServerRequestPseudo-Object	6-3

6.3.1	ExplicitRequest State: ServerRequestPseudo-Object	6-3
6.4	DSI: Language Mapping	6-4
6.4.1	ServerRequest's Handling of Operation Parameters	6-4
6.4.2	Registering Dynamic Implementation Routines	6-5
7.	Dynamic management of Any values	7-1
7.1	Overview	7-2
7.2	DynAny API	7-3
7.2.1	Locality and usage constraints	7-5
7.2.2	Creating a DynAny object	7-5
7.2.3	The DynAny interface	7-7
7.2.4	The DynFixed interface	7-10
7.2.5	The DynEnum interface	7-10
7.2.6	The DynStruct interface	7-11
7.2.7	The DynUnion interface	7-12
7.2.8	The DynSequence interface	7-13
7.2.9	The DynArray interface	7-13
7.3	Usage in C++ language	7-14
7.3.1	Dynamic creation of CORBA::Any values	7-14
7.3.2	Dynamic interpretation of CORBA::Any values	7-15
8.	The Interface Repository	8-1
8.1	Overview	8-1
8.2	Scope of an Interface Repository	8-2
8.3	Implementation Dependencies	8-4
8.3.1	Managing Interface Repositories	8-5
8.4	Basics	8-6
8.4.1	Names and Identifiers	8-6
8.4.2	Types and TypeCodes	8-6
8.4.3	Interface Objects	8-7
8.4.4	Structure and Navigation of Interface Objects	8-7
8.5	Interface Repository Interfaces	8-9
8.5.1	Supporting Type Definitions	8-9
8.5.2	IRObjct	8-10
8.5.3	Contained	8-11
8.5.4	Container	8-13
8.5.5	IDLType	8-17
8.5.6	Repository	8-17
8.5.7	ModuleDef	8-19

Contents

	8.5.8	ConstantDef Interface	8-19
	8.5.9	StructDef	8-20
	8.5.10	UnionDef	8-21
	8.5.11	EnumDef	8-22
	8.5.12	AliasDef	8-22
	8.5.13	PrimitiveDef	8-23
	8.5.14	StringDef	8-23
	8.5.15	WstringDef	8-24
	8.5.16	FixedDef	8-24
	8.5.17	SequenceDef	8-24
	8.5.18	ArrayDef	8-25
	8.5.19	ExceptionDef	8-26
	8.5.20	AttributeDef	8-26
	8.5.21	OperationDef	8-27
	8.5.22	InterfaceDef	8-29
8.6		RepositoryIds	8-31
	8.6.1	OMG IDL Format	8-31
	8.6.2	DCE UUID Format	8-31
	8.6.3	LOCAL Format	8-32
	8.6.4	Pragma Directives for RepositoryId	8-32
	8.6.5	For More Information	8-34
	8.6.6	RepositoryIDs for OMG-Specified Types.	8-34
8.7		TypeCodes	8-35
	8.7.1	The TypeCode Interface	8-36
	8.7.2	TypeCode Constants	8-40
	8.7.3	Creating TypeCodes	8-41
8.8		OMG IDL for Interface Repository	8-44
9.		The Portable Object Adaptor	9-1
	9.1	Overview	9-1
	9.2	Abstract Model Description	9-2
		9.2.1 Model Components	9-2
		9.2.2 Model Architecture	9-4
		9.2.3 POA Creation	9-6
		9.2.4 Reference Creation	9-7
		9.2.5 Object Activation States	9-8
		9.2.6 Request Processing	9-9
		9.2.7 Implicit Activation	9-10
		9.2.8 Multi-threading	9-11
		9.2.9 Dynamic Skeleton Interface	9-12

9.2.10	Location Transparency	9-13
9.3	Interfaces	9-13
9.3.1	The Servant IDL Type	9-14
9.3.2	POAManager Interface.	9-14
9.3.3	AdapterActivator Interface	9-19
9.3.4	ServantManager Interface.	9-20
9.3.5	ServantActivator Interface	9-21
9.3.6	ServantLocator Interface	9-24
9.3.7	POA Policy Objects	9-25
9.3.8	POA Interface.	9-30
9.3.9	Current operations	9-38
9.4	IDL for PortableServer module	9-38
9.5	UML Description of PortableServer.	9-46
9.6	Usage Scenarios	9-47
9.6.1	Getting the root POA	9-48
9.6.2	Creating a POA.	9-48
9.6.3	Explicit Activation with POA-assigned Object Ids	9-48
9.6.4	Explicit activation with user assigned Object Ids	9-49
9.6.5	Creating references before activation	9-50
9.6.6	Servant Manager Definition and Creation	9-51
9.6.7	Object activation on demand	9-52
9.6.8	Persistent objects with POA-assigned Ids.	9-54
9.6.9	Multiple Object Ids Mapping to a Single Servant	9-54
9.6.10	One Servant for all Objects.	9-54
9.6.11	Single Servant, many objects and types, using DSI	9-57
10.	Interoperability Overview	10-1
10.1	Elements of Interoperability.	10-1
10.1.1	ORB Interoperability Architecture	10-2
10.1.2	Inter-ORB Bridge Support	10-2
10.1.3	General Inter-ORB Protocol (GIOP)	10-3
10.1.4	Internet Inter-ORB Protocol (IIOP)	10-3
10.1.5	Environment-Specific Inter-ORB Protocols (ESIOPs)	10-4
10.2	Relationship to Previous Versions of CORBA	10-4
10.3	Examples of Interoperability Solutions	10-5
10.3.1	Example 1.	10-5
10.3.2	Example 2.	10-5
10.3.3	Example 3.	10-5

10.3.4	Interoperability Compliance	10-5
10.4	Motivating Factors	10-8
10.4.1	ORB Implementation Diversity	10-8
10.4.2	ORB Boundaries.	10-8
10.4.3	ORBs Vary in Scope, Distance, and Lifetime.	10-9
10.5	Interoperability Design Goals	10-9
10.5.1	Non-Goals	10-10
11.	ORB Interoperability Architecture	11-1
11.1	Overview	11-1
11.1.1	Domains	11-2
11.1.2	Bridging Domains	11-2
11.2	ORBs and ORB Services	11-3
11.2.1	The Nature of ORB Services	11-3
11.2.2	ORB Services and Object Requests	11-3
11.2.3	Selection of ORB Services	11-4
11.3	Domains	11-5
11.3.1	Definition of a Domain.	11-5
11.3.2	Mapping Between Domains: Bridging	11-6
11.4	Interoperability Between ORBs	11-7
11.4.1	ORB Services and Domains	11-7
11.4.2	ORBs and Domains	11-7
11.4.3	Interoperability Approaches	11-8
11.4.4	Policy-Mediated Bridging	11-10
11.4.5	Configurations of Bridges in Networks	11-11
11.5	Object Addressing	11-11
11.5.1	Domain-relative Object Referencing	11-12
11.5.2	Handling of Referencing Between Domains.	11-12
11.6	An Information Model for Object References	11-14
11.6.1	What Information Do Bridges Need?	11-14
11.6.2	Interoperable Object References: IORs	11-14
11.6.3	Standard IOR Components	11-17
11.6.4	Profile and Component Composition in IORs	11-18
11.6.5	IOR Creation and Scope	11-19
11.6.6	Stringified Object References.	11-19
11.6.7	Object Service Context.	11-20
11.7	Code Set Conversion	11-22
11.7.1	Character Processing Terminology.	11-22
11.7.2 Code Set Conversion Framework.	11-25

11.7.3	Mapping to Generic Character Environments . . .	11-33
11.8	
	Example of Generic Environment Mapping	11-34
11.8.1	Generic Mappings	11-35
11.8.2	Interoperation and Generic Mappings	11-35
11.9	
	Relevant OSFM Registry Interfaces	11-35
11.9.1	Character and Code Set Registry	11-35
11.9.2	Access Routines	11-36
12.	Building Inter-ORB Bridges	12-1
12.1	In-Line and Request-Level Bridging	12-2
12.1.1	In-line Bridging	12-3
12.1.2	Request-level Bridging	12-3
12.1.3	Collocated ORBs	12-4
12.2	Proxy Creation and Management	12-5
12.3	Interface-specific Bridges and Generic Bridges	12-6
12.4	Building Generic Request-Level Bridges	12-6
12.5	Bridging Non-Referencing Domains	12-7
12.6	Bootstrapping Bridges	12-7
13.	General Inter-ORB Protocol	13-1
13.1	Goals of the General Inter-ORB Protocol	13-2
13.2	GIOP Overview	13-2
13.2.1	Common Data Representation (CDR)	13-3
13.2.2	GIOP Message Overview	13-3
13.2.3	GIOP Message Transfer	13-4
13.3	CDR Transfer Syntax	13-4
13.3.1	Primitive Types	13-5
13.3.2	OMG IDL Constructed Types	13-10
13.3.3	Encapsulation	13-12
13.3.4	Pseudo-Object Types	13-13
13.3.5	Object References	13-18
13.4	GIOP Message Formats	13-19
13.4.1	GIOP Message Header	13-19
13.4.2	Reply Message	13-24
13.4.3	CancelRequest Message	13-26
13.4.4	LocateRequest Message	13-27
13.4.5	LocateReply Message	13-28
13.4.6	CloseConnection Message	13-29

13.4.7	MessageError Message	13-29
13.4.8	Fragment Message	13-29
13.5	GIOP Message Transport	13-30
13.5.1	Connection Management	13-30
13.5.2	Message Ordering	13-32
13.6	Object Location	13-32
13.7	Internet Inter-ORB Protocol (IIOP)	13-33
13.7.1	TCP/IP Connection Usage	13-34
13.7.2	IIOP IOR Profiles	13-34
13.7.3	IIOP IOR Profile Components	13-37
13.8	OMG IDL	13-37
13.8.1	GIOP Module	13-37
13.8.2	IIOP Module	13-39
14.	The DCE ESIOP	14-1
14.1	Goals of the DCE Common Inter-ORB Protocol	14-1
14.2	DCE Common Inter-ORB Protocol Overview	14-2
14.2.1	DCE-CIOP RPC	14-2
14.2.2	DCE-CIOP Data Representation	14-3
14.2.3	DCE-CIOP Messages	14-4
14.2.4	Interoperable Object Reference (IOR)	14-5
14.3	DCE-CIOP Message Transport	14-5
14.3.1	Pipe-based Interface	14-6
14.3.2	Array-based Interface	14-8
14.4	DCE-CIOP Message Formats	14-11
14.4.1	DCE_CIOP Invoke Request Message	14-11
14.4.2	DCE-CIOP Invoke Response Message	14-12
14.4.3	DCE-CIOP Locate Request Message	14-14
14.4.4	DCE-CIOP Locate Response Message	14-15
14.5	DCE-CIOP Object References	14-16
14.5.1	DCE-CIOP String Binding Component	14-17
14.5.2	DCE-CIOP Binding Name Component	14-18
14.5.3	DCE-CIOP No Pipes Component	14-19
14.5.4	Complete Object Key Component	14-19
14.5.5	Endpoint ID Position Component	14-20
14.5.6	Location Policy Component	14-20
14.6	DCE-CIOP Object Location	14-22
14.6.1	Location Mechanism Overview	14-22
14.6.2	Activation	14-23
14.6.3	Basic Location Algorithm	14-23

14.6.4	Use of the Location Policy and the Endpoint ID	14-24
14.7	OMG IDL for the DCE CIOP Module	14-25
14.8	References for this Chapter	14-26
15.	Interworking Architecture.	15-1
15.1	Purpose of the Interworking Architecture	15-2
15.1.1	Comparing COM Objects to CORBA Objects	15-2
15.2	Interworking Object Model	15-3
15.2.1	Relationship to CORBA Object Model.	15-3
15.2.2	Relationship to the OLE/COM Model	15-4
15.2.3	Basic Description of the Interworking Model.	15-4
15.3	Interworking Mapping Issues.	15-8
15.4	Interface Mapping	15-8
15.4.1	CORBA/COM	15-9
15.4.2	CORBA/Automation	15-9
15.4.3	COM/CORBA	15-10
15.4.4	Automation/CORBA	15-10
15.5	Interface Composition Mappings.	15-11
15.5.1	CORBA/COM	15-11
15.5.2	Detailed Mapping Rules.	15-13
15.5.3	Example of Applying Ordering Rules	15-14
15.5.4	Mapping Interface Identity	15-16
15.6	Object Identity, Binding, and Life Cycle	15-18
15.6.1	Object Identity Issues.	15-18
15.6.2	Binding and Life Cycle	15-20
15.7	Interworking Interfaces	15-23
15.7.1	SimpleFactory Interface	15-23
15.7.2	IMonikerProvider Interface and Moniker Use	15-23
15.7.3	ICORBAFactory Interface	15-24
15.7.4	IForeignObject Interface.	15-26
15.7.5	ICORBAObject Interface	15-27
15.7.6	IORBObject Interface.	15-28
15.7.7	Naming Conventions for View Components.	15-29
15.8	Distribution	15-32
15.8.1	Bridge Locality.	15-32
15.8.2	Distribution Architecture	15-33
15.9	Interworking Targets	15-34
15.10	Compliance to COM/CORBA Interworking	15-34
15.10.1	Products Subject to Compliance.	15-34

15.10.2	Compliance Points	15-36
16.	Mapping: COM and CORBA	16-1
16.1	Data Type Mapping	16-1
16.2	CORBA to COM Data Type Mapping	16-2
16.2.1	
	Mapping for Basic Data Types	16-2
16.2.2	
	Mapping for Constants	16-2
16.2.3	Mapping for Enumerators.	16-3
16.2.4	Mapping for String Types.	16-4
16.2.5	Mapping for Struct Types.	16-5
16.2.6	
	Mapping for Union Types.	16-6
16.2.7	Mapping for Sequence Types	16-8
16.2.8	Mapping for Array Types	16-9
16.2.9	Mapping for the any Type.	16-10
16.2.10	Interface Mapping	16-11
16.2.11	
	Inheritance Mapping.	16-25
16.2.12	Mapping for Pseudo-Objects	16-28
16.2.13	
	Interface Repository Mapping	16-31
16.3	COM to CORBA Data Type Mapping	16-32
16.3.1	Mapping for Basic Data Types	16-32
16.3.2	
	Mapping for Constants	16-33
16.3.3	
	Mapping for Enumerators.	16-33
16.3.4	
	Mapping for String Types.	16-34
16.3.5	
	Mapping for Structure Types	16-36
16.3.6	
	Mapping for Union Types.	16-37
16.3.7	
	Mapping for Array Types	16-39
16.3.8	Mapping for VARIANT	16-40
16.3.9	
	Mapping for Pointers	16-43
16.3.10	Interface Mapping	16-43

16.3.11	Mapping for Read-Only Attributes	16-48
16.3.12	Mapping for Read-Write Attributes	16-48
17.	Mapping: OLE Automation and CORBA	17-1
17.1	Mapping CORBA Objects to OLE Automation	17-2
17.1.1	Architectural Overview	17-2
17.1.2	Main Features of the Mapping	17-3
17.1.3	Mapping for Interfaces	17-3
17.1.4	Mapping for Basic Data Types	17-9
17.1.5	Special Cases of Basic Data Type Mapping	17-11
17.1.6	Mapping for Strings	17-11
17.1.7	ACompleteIDLtoODLMappingfortheBasicDataTypes	17-12
17.1.8	Mapping for Object References	17-16
17.1.9	Mapping for Enumerated Types	17-18
17.1.10	Mapping for Arrays and Sequences	17-19
17.1.11	Mapping for CORBA Complex Types	17-20
17.1.12	Mapping for TypeCodes	17-23
17.1.13	Mapping for anys	17-24
17.1.14	Mapping for Typedefs	17-25
17.1.15	Mapping for Constants	17-25
17.1.16	Getting Initial CORBA Object References	17-26
17.1.17	Creating Initial in Parameters for Complex Types	17-27
17.1.18	MappingCORBAExceptionstoAutomationExceptions	17-29
17.1.19	Conventions for Naming Components of the Automation View	17-36
17.1.20	Naming Conventions for Pseudo-Structs, Pseudo-Unions, and Pseudo-Exceptions	17-36
17.1.21	AutomationViewInterfaceandDispatchInterface(Nondual)	17-36
17.1.22	Aggregation of Automation Views	17-37
17.1.23	DII and DSI	17-37
17.2	Automation Objects as CORBA Objects	17-38
17.2.1	Architectural Overview	17-38
17.2.2	Main Features of the Mapping	17-39
17.2.3	Getting Initial Object References	17-39
17.2.4	Mapping for Interfaces	17-40
17.2.5	Mapping for Inheritance	17-40

17.2.6	Mapping for ODL Properties and Methods . . .	17-41
17.2.7	Mapping for Automation Basic Data Types . . .	17-42
17.2.8	Conversion Errors	17-43
17.2.9	Special Cases of Data Type Conversion	17-43
17.2.10	A Complete OMG IDL to ODL Mapping for the Basic Data Types	17-43
17.2.11	Mapping for Object References	17-46
17.2.12	Mapping for Enumerated Types	17-47
17.2.13	Mapping for SafeArrays	17-47
17.2.14	Mapping for Typedefs	17-48
17.2.15	Mapping for VARIANTS	17-48
17.2.16	Mapping Automation Exceptions to CORBA . .	17-48
17.3	Older OLE Automation Controllers	17-49
17.3.1 Mapping for OMG IDL Arrays and Sequences to Collections 17-49	
17.4	Example Mappings.	17-50
17.4.1	Mapping the OMG Naming Service to OLE Automation 17-50	
17.4.2	Mapping a COM Service to OMG IDL	17-51
17.4.3	Mapping an OMG Object Service to OLE Automation 17-55	
18.	Interceptors	18-1
18.1	Introduction	18-1
18.1.1	ORB Core and ORB Services	18-2
18.2	Interceptors	18-2
18.2.1	Generic ORB Services and Interceptors	18-2
18.2.2	Request-Level Interceptors	18-3
18.2.3	Message-Level Interceptors	18-3
18.2.4	Selecting Interceptors	18-4
18.3	Client-Target Binding	18-4
18.3.1	Binding Model	18-5
18.3.2	Establishing the Binding and Interceptors	18-5
18.4	Using Interceptors	18-6
18.4.1	Request-Level Interceptors	18-6
18.4.2	Message-Level Interceptors	18-7
18.5	Interceptor Interfaces	18-7
18.5.1	Client and Target Invoke	18-8
18.5.2	Send and Receive Message	18-8
18.6	IDL for Interceptors	18-9

19. C Language Mapping	19-1
19.1 Requirements for a Language Mapping	19-2
19.1.1 Basic Data Types	19-3
19.1.2 Constructed Data Types	19-3
19.1.3 Constants	19-3
19.1.4 Objects	19-3
19.1.5 Invocation of Operations	19-4
19.1.6 Exceptions	19-4
19.1.7 Attributes	19-5
19.1.8 ORB Interfaces	19-5
19.2 Scoped Names	19-5
19.3 Mapping for Interfaces	19-6
19.4 Inheritance and Operation Names	19-8
19.5 Mapping for Attributes	19-8
19.6 Mapping for Constants	19-10
19.7 Mapping for Basic Data Types	19-10
19.8 Mapping Considerations for Constructed Types	19-11
19.9 Mapping for Structure Types	19-12
19.10 Mapping for Union Types	19-12
19.11 Mapping for Sequence Types	19-13
19.12 Mapping for Strings	19-16
19.13 Mapping for Wide Strings	19-18
19.14 Mapping for Fixed	19-18
19.15 Mapping for Arrays	19-19
19.16 Mapping for Exception Types	19-20
19.17 Implicit Arguments to Operations	19-21
19.18 Interpretation of Functions with Empty Argument Lists ..	19-21
19.19 Argument Passing Considerations	19-21
19.20 Return Result Passing Considerations	19-22
19.21 Summary of Argument/Result Passing	19-23
19.22 Handling Exceptions	19-26
19.23 Method Routine Signatures	19-29
19.24 Include Files	19-29
19.25 Pseudo-objects	19-29
19.25.1 ORB Operations	19-30
19.26 Mapping for Object Implementations	19-30
19.26.1 Operation-specific Details	19-31

19.26.2	PortableServer Functions	19-31
19.26.3	Mapping for PortableServer::ServantLocator::Cookie 19-31	
19.26.4	Servant Mapping	19-32
19.26.5	Interface Skeletons	19-33
19.26.6	Servant Structure Initialization	19-35
19.26.7	Application Servants.	19-37
19.26.8	Method Signatures	19-39
19.27	Mapping of the Dynamic Skeleton Interface to C	19-40
19.27.1	Mapping of ServerRequest to C	19-40
19.27.2	Mapping of Dynamic Implementation Routine to C 19-42	
19.28	ORB Initialization Operations	19-44
20.	Mapping of OMG IDL to C++.	20-1
20.1	Preliminary Information.	20-3
20.1.1	Overview	20-3
20.1.2	Scoped Names	20-4
20.1.3	C++ Type Size Requirements.	20-5
20.1.4	CORBA Module.	20-5
20.2	Mapping for Modules.	20-5
20.3	Mapping for Interfaces.	20-6
20.3.1	Object Reference Types	20-6
20.3.2	Widening Object References	20-7
20.3.3	Object Reference Operations	20-8
20.3.4	Narrowing Object References.	20-9
20.3.5	Nil Object Reference	20-10
20.3.6	Object Reference Out Parameter	20-10
20.3.7	Interface Mapping Example	20-11
20.4	Mapping for Constants.	20-13
20.5	Mapping for Basic Data Types.	20-15
20.6	Mapping for Enums	20-16
20.7	Mapping for String Types.	20-17
20.8	Mapping for Wide String Types.	20-20
20.9	Mapping for Structured Types	20-21
20.9.1	T_var Types	20-22
20.9.2	T_out Types	20-25
20.10	Mapping for Struct Types.	20-27
20.11	Mapping for Fixed	20-29
20.11.1	Fixed T_var and T_out Types.	20-31

20.12	Mapping for Union Types	20-31
20.13	Mapping for Sequence Types	20-35
20.13.1	Sequence Example	20-38
20.13.2	Using the “release” Constructor Parameter . . .	20-39
20.13.3	Additional Memory Management Functions . .	20-40
20.13.4	Sequence T_var and T_out Types	20-41
20.14	Mapping For Array Types	20-41
20.15	Mapping For Typedefs	20-44
20.16	Mapping for the Any Type	20-46
20.16.1	Handling Typed Values.	20-46
20.16.2	Insertion into any	20-46
20.16.3	Extraction from any	20-49
20.16.4	Distinguishing boolean, octet, char, wchar, bounded string, and bounded wstring	20-52
20.16.5	Widening to Object.	20-55
20.16.6	Handling Untyped Values.	20-56
20.16.7	Any Constructors, Destructor, Assignment Operator 20-57	
20.16.8	The Any Class	20-57
20.16.9	The Any_var Class	20-57
20.17	Mapping for Exception Types	20-58
20.18	Mapping For Operations and Attributes	20-61
20.19	Implicit Arguments to Operations	20-62
20.20	Argument Passing Considerations	20-62
20.20.1	Operation Parameters and Signatures	20-65
20.21	Mapping of Pseudo Objects to C++	20-68
20.22	Usage	20-69
20.23	Mapping Rules	20-69
20.24	Relation to the C PIDL Mapping	20-70
20.25	Environment.	20-71
20.25.1	Environment Interface	20-71
20.25.2	Environment C++ Class	20-72
20.25.3	Differences from C-PIDL	20-72
20.25.4	Memory Management.	20-72
20.26	NamedValue	20-72
20.26.1	NamedValue Interface	20-73
20.26.2	NamedValue C++ Class	20-73
20.26.3	Differences from C-PIDL	20-73
20.26.4	Memory Management.	20-73

20.27	NVList	20-73
20.27.1	NVList Interface	20-74
20.27.2	NVList C++ Class	20-74
20.27.3	Differences from C-PIDL	20-75
20.27.4	Memory Management	20-75
20.28	Request	20-75
20.28.1	Request Interface	20-77
20.28.2	Request C++ Class	20-78
20.28.3	Differences from C-PIDL	20-79
20.28.4	Memory Management	20-80
20.29	Context	20-80
20.29.1	Context Interface	20-80
20.29.2	Context C++ Class	20-81
20.29.3	Differences from C-PIDL	20-81
20.29.4	Memory Management	20-81
20.30	TypeCode	20-81
20.30.1	TypeCode Interface	20-82
20.30.2	TypeCode C++ Class	20-82
20.30.3	Differences from C-PIDL	20-83
20.30.4	Memory Management	20-83
20.31	ORB	20-83
20.31.1	ORB Interface	20-83
20.31.2	ORB C++ Class	20-84
20.31.3	Differences from C-PIDL	20-85
20.31.4	Mapping of ORB Initialization Operations	20-85
20.32	Object	20-86
20.32.1	Object Interface	20-87
20.32.2	Object C++ Class	20-87
20.33	Server-Side Mapping	20-88
20.34	Implementing Interfaces	20-89
20.34.1	Mapping of PortableServer::Servant	20-89
20.34.2	Skeleton Operations	20-90
20.34.3	Inheritance-Based Interface Implementation	20-91
20.34.4	Delegation-Based Interface Implementation	20-93
20.35	Implementing Operations	20-97
20.35.1	Skeleton Derivation From Object	20-99
20.36	Mapping of Dynamic Skeleton Interface to C++	20-99
20.36.1	Mapping of ServerRequest to C++	20-99

20.36.2	Handling Operation Parameters and Results. . .	20-100
20.36.3	Mapping PortableServerDynamicImplementationRoutine	20-100
20.37	PortableServer Functions	20-101
20.38	Mapping for PortableServer::ServantManager	20-102
20.38.1	Mapping for Cookie	20-102
20.38.2	ServantManagers and AdapterActivators	20-102
20.39	C++ Definitions for CORBA	20-103
20.39.1	Primitive Types.	20-103
20.39.2	String_var and String_out Class	20-104
20.39.3	WString_var and WString_out	20-104
20.39.4	Any Class	20-105
20.39.5	Any_var Class	20-107
20.39.6	Exception Class	20-108
20.39.7	SystemException Class.	20-108
20.39.8	UserException Class.	20-108
20.39.9	UnknownUserException Class	20-109
20.39.10	release and is_nil	20-109
20.39.11	Object Class	20-110
20.39.12	Environment Class	20-111
20.39.13	NamedValue Class	20-111
20.39.14	NVList Class	20-111

20.39.15	ExceptionList Class	20-112
20.39.16	ContextList Class	20-112
20.39.17	Request Class	20-112
20.39.18	Context Class	20-113
20.39.19	TypeCode Class	20-113
20.39.20	ORB Class	20-114
20.39.21	ORB Initialization	20-115
20.39.22	General T_out Types	20-115
20.40	Alternative Mappings For C++ Dialects	20-116
20.40.1	Without Namespaces	20-116
20.40.2	Without Exception Handling	20-116
20.41	C++ Keywords	20-118
21.	Mapping of OMG IDL to Smalltalk	21-1
21.1	Mapping Summary	21-3
21.2	Key Design Decisions	21-4
21.2.1	Consistency, Style, Flexibility, Portability, and Implementation	21-5
21.3	Implementation Constraints	21-5
21.3.1	Avoiding Name Space Collisions	21-5
21.3.2	Limitations on OMG IDL Types	21-6
21.4	Smalltalk Implementation Requirements	21-6
21.5	Conversion of Names to Smalltalk Identifiers	21-7
21.6	Mapping for Interfaces	21-8
21.7	Memory Usage	21-8
21.8	Mapping for Objects	21-8
21.9	Invocation of Operations	21-8
21.10	Mapping for Attributes	21-9
21.10.1	Mapping for Constants	21-10

21.11	Mapping for Basic Data Types	21-10
21.12	Mapping for the Any Type	21-12
21.13	Mapping for Enums	21-12
21.14	Mapping for Struct Types	21-13
21.15	Mapping for Fixed Types	21-14
21.16	Mapping for Union Types	21-14
	21.16.1 Implicit Binding	21-14
	21.16.2 Explicit Binding	21-15
21.17	Mapping for Sequence Types	21-15
21.18	Mapping for String Types	21-15
21.19	Mapping for Wide String Types	21-15
21.20	Mapping for Array Types	21-15
21.21	Mapping for Exception Types	21-15
21.22	Mapping for Operations	21-16
21.23	Implicit Arguments to Operations	21-16
21.24	Argument Passing Considerations	21-17
21.25	Handling Exceptions	21-17
21.26	Exception Values	21-18
	21.26.1 The CORBAExceptionValue Protocol	21-19
21.27	CORBA::Request	21-19
21.28	CORBA::Context	21-20
21.29	CORBA::Object	21-21
21.30	CORBA::ORB	21-21
21.31	CORBA::NamedValue	21-22
21.32	CORBA::NVList	21-23
22.	Mapping of OMG IDL to Cobol	22-1
22.1	Overview	22-2
22.2	Mapping of IDL to COBOL	22-2
	22.2.1 Mapping of IDL Identifiers to COBOL	22-2
22.3	Scoped Names	22-3
22.4	Memory Management	22-4
22.5	Mapping for Interfaces	22-5
	22.5.1 Object References	22-5
	22.5.2 Object References as Arguments	22-5
	22.5.3 Inheritance and Interface Names	22-6
22.6	Mapping for Attributes	22-6

22.7	Mapping for Constants	22-7
22.8	Mapping for Basic Data Types	22-7
22.8.1	Boolean	22-8
22.8.2	enum	22-8
22.8.3	any	22-9
22.9	Mapping for Fixed Types	22-10
22.10	Mapping for Struct Types	22-10
22.11	Mapping for Union Types	22-10
22.12	Mapping for Sequence Types	22-11
22.12.1	Bounded Sequence	22-11
22.12.2	Unbounded Sequence	22-12
22.12.3	Sequence Element Accessor Functions	22-12
22.12.4	Nested Sequences	22-13
22.12.5	Sequence parameter passing considerations	22-14
22.13	Mapping for Strings	22-15
22.13.1	How string is mapped to COBOL	22-15
22.13.2	How wstring is mapped to COBOL	22-16
22.13.3	string / wstring argument passing considerations	22-18
22.14	Mapping for Arrays	22-19
22.15	Mapping for Exception Types	22-19
22.16	Argument Conventions	22-19
22.16.1	Implicit Arguments to Operations	22-19
22.16.2	Argument passing Considerations	22-20
22.16.3	Summary of Argument/Result Passing	22-22
22.17	Memory Management	22-23
22.17.1	Summary of Parameter Storage Responsibilities	22-23
22.18	Handling Exceptions	22-25
22.18.1	Passing Exception details back to the caller	22-25
22.18.2	Exception Handling Functions	22-26
22.18.3	Example of how to handle the CORBA-Exception parameter	22-27
22.19	Pseudo Objects	22-29
22.19.1	Mapping Pseudo Objects to COBOL	22-29
22.19.2	Pseudo-Object mapping example	22-30
22.20	Mapping of the Dynamic Skeleton Interface to COBOL	22-39
22.20.1	Mapping of the ServerRequest to COBOL	22-40

22.20.2 Mapping of Dynamic Implementation Routine to COBOL	22-41
22.21 ORB Initialization Operations	22-44
22.22 Operations for Obtaining Initial Object References	22-45
22.23 ORB Supplied Functions for Mapping	22-46
22.23.1	
Memory Management routines	22-46
22.24 Accessor Functions	22-47
22.24.1 CORBA-sequence-element-get and CORBA-sequence-element-set	22-47
22.24.2 CORBA-string-get and CORBA-string-set	22-48
22.24.3 CORBA-wstring-get & CORBA-wstring-set	22-49
22.25 Extensions to COBOL 85.	22-49
22.25.1 Untyped Pointers and Pointer manipulation	22-50
22.25.2 Pointer Manipulation	22-50
22.25.3 Floating point	22-50
22.25.4 Constants	22-51
22.25.5 Typedefs	22-51
22.26 References	22-53
23. Mapping of OMG IDL to Ada.	23-1
23.1 Overview	23-1
23.1.1 Ada Implementation Requirements	23-2
23.2 Mapping Summary.	23-2
23.2.1 Interfaces and Tagged Types.	23-2
23.2.2 Operations	23-3
23.2.3 Attributes	23-3
23.2.4 Inheritance	23-4
23.2.5 Data Types	23-4
23.2.6 Exceptions	23-4
23.2.7 Names and Scoping	23-5
23.3 Other Mapping Requirements	23-5
23.3.1 Implementation Considerations	23-5
23.3.2 Calling Convention.	23-5
23.3.3 Memory Management.	23-5
23.3.4 Tasking	23-5
23.4 Lexical Mapping	23-6
23.4.1 Mapping of Identifiers	23-6
23.4.2 Mapping of Literals	23-6
23.4.3 Mapping of Constant Expressions	23-8

23.5	Mapping of IDL to Ada	23-10
23.5.1	Names.	23-10
23.5.2	IDL Files	23-11
23.5.3	CORBA Subsystem	23-12
23.5.4	Mapping Modules.	23-12
23.5.5	Mapping for Interfaces (Client-Side Specific) .	23-12
23.5.6	Mapping for Types	23-20
23.5.7	Mapping for Any Type	23-29
23.5.8	Mapping for Exception Types.	23-30
23.5.9	Mapping of Operations and Attributes (Client-Side Specific)	23-35
23.5.10	Argument Passing Considerations	23-36
23.5.11	Tasking Considerations.	23-36
23.6	Mapping of Pseudo-Objects to Ada	23-36
23.6.1	NamedValue	23-37
23.6.2	NVList	23-37
23.6.3	Request.	23-38
23.6.4	Context	23-39
23.6.5	Principal	23-40
23.6.6	TypeCode	23-40
23.6.7	ORB	23-42
23.6.8	Object	23-42
23.6.9	Environment	23-43
23.7	Server-Side Mapping	23-43
23.7.1	Implementing Interfaces.	23-44
23.7.2	Implementing Operations and Attributes	23-44
23.7.3	Examples	23-44
23.8	
	Predefined Language Environment: Subsystem CORBA. . . .	23-45
23.8.1	Package CORBA	23-45
23.8.2	Package CORBA.Bounded_Strings;.	23-50
23.8.3	Package CORBA.Context.	23-50
23.8.4	Package CORBA.Environment.	23-51
23.8.5	Package CORBA.Forward	23-51
23.8.6	Package CORBA.Iterate_Over_Any_Elements	23-51
23.8.7	Package CORBA.NVList	23-52
23.8.8	Package CORBA.Object.	23-52
23.8.9	Package CORBA.ORB	23-53
23.8.10	Package CORBA.Principal	23-54
23.8.11	Package CORBA.Request.	23-54

23.8.12	Package CORBA.Sequences.	23-55
23.8.13	Package CORBA.Sequences.Bounded	23-56
23.8.14	Package CORBA.Sequences.Unbounded	23-61
23.9	
	Glossary of Ada Terms	23-65
24.	Mapping of OMG IDL to Java	24-1
24.1	Names	24-2
24.1.1	Reserved Names.	24-2
24.2	Mapping of Module	24-3
24.2.1	Example	24-3
24.3	Mapping for Basic Types	24-3
24.3.1	Introduction	24-3
24.3.2	Boolean	24-8
24.3.3	Character Types	24-8
24.3.4	Octet.	24-8
24.3.5	String Types	24-8
24.3.6	Integer Types	24-8
24.3.7	Floating Point Types.	24-8
24.3.8	Future Fixed Point Types	24-9
24.3.9	Future Long Double Types	24-9
24.4	Helper Classes	24-9
24.4.1	Examples	24-10
24.5	Mapping for Constant	24-10
24.5.1	Constants Within An Interface	24-10
24.5.2	Constants Not Within An Interface.	24-11
24.6	Mapping for Enum	24-11
24.6.1	Example	24-13
24.7	Mapping for Struct.	24-13
24.7.1	Example	24-14
24.8	Mapping for Union.	24-14
24.8.1	Example	24-16
24.9	Mapping for Sequence	24-17
24.9.1	Example	24-17
24.10	Mapping for Array	24-18
24.10.1	Example	24-18
24.11	Mapping for Interface	24-19
24.11.1	Basics	24-19
24.11.2	Parameter Passing Modes.	24-21

24.12	Mapping for Exception.	24-22
24.12.1	User Defined Exceptions	24-23
24.12.2	System Exceptions	24-24
24.13	Mapping for the Any Type	24-26
24.14	Mapping for Certain Nested Types.	24-29
24.14.1	Example	24-29
24.15	Mapping for Typedef	24-30
24.15.1	Simple IDL types	24-30
24.15.2	Complex IDL types	24-30
24.16	Mapping Pseudo Objects to Java.	24-31
24.16.1	Introduction	24-31
24.16.2	Certain Exceptions	24-32
24.16.3	Environment	24-32
24.16.4	NamedValue	24-33
24.16.5	NVList	24-34
24.16.6	ExceptionList	24-34
24.16.7	Context	24-35
24.16.8	ContextList	24-36
24.16.9	Request	24-37
24.16.10	ServerRequest and Dynamic Implementation	24-38
24.16.11	TypeCode	24-39
24.16.12	ORB	24-42
24.16.13	CORBA::Object	24-46
24.16.14	Current	24-47
24.16.15	Principal	24-47
24.17	Server-Side Mapping	24-48
24.17.1	Introduction	24-48
24.17.2	Transient Objects	24-48
24.18	Java ORB Portability Interfaces	24-49
24.18.1	Introduction	24-49
24.18.2	Architecture	24-50
24.18.3	Streamable APIs	24-52
24.18.4	Streaming APIs	24-52
24.18.5	Portability Stub Interfaces	24-55
24.18.6	Delegate	24-57
24.18.7	Skeleton	24-58
24.18.8	ORB Initialization	24-58