

# Implementing Distributed Systems with Java and CORBA

---

Markus Aleksy · Axel Korthaus  
Martin Schader

---

# Implementing Distributed Systems with Java and CORBA

With 27 Figures and 13 Tables

Dr. Markus Aleksy  
Dr. Axel Korthaus  
Professor Dr. Martin Schader

University of Mannheim  
Schloss  
68131 Mannheim  
Germany

markus.aleksy@uni-mannheim.de  
axel.korthaus@uni-mannheim.de  
martin.schader@uni-mannheim.de

Cataloging-in-Publication Data  
Library of Congress Control Number: 2005926835

ISBN 3-540-24173-6 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer-Verlag. Violations are liable for prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media  
springeronline.com

© Springer Berlin · Heidelberg 2005  
Printed in Germany

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Hardcover-Design: Erich Kirchner, Heidelberg

SPIN 11370925

43/3153-5 4 3 2 1 0 – Printed on acid-free paper

# Preface

This book addresses readers interested in the design and development of distributed software systems with Java and CORBA. The programming language Java, first introduced by Sun Microsystems in 1995 in an attempt to remedy some of the deficiencies of C++, has meanwhile pervaded all fields of software development. CORBA, the Common Object Request Broker Architecture, is an industry standard that enables the platform- and programming language-independent implementation of distributed object-oriented systems.

When developing and testing the examples and exercises for this book, we used three different Object Request Broker products (ORBs) that are available free of charge. The first is JacORB 2.2, a Java object request broker originated in the CS department at Freie Universität Berlin, see <http://www.jacorb.org>. The second one is part of Sun's Java™ 2 Platform Standard Edition 5.0 Development Kit (JDK), see <http://java.sun.com>. The third ORB is OpenORB 1.3.1 developed by the Community OpenORB Project, see <http://openorb.sf.net>. Detailed information on downloading, installing, and customizing these ORBs can be found in Appendix E and at the book's website <http://www.wifo.uni-mannheim.de/CORBA> in subdirectory ORB.

Under this URL, one also has the possibility to give feedback, send in corrections, or submit any other suggestions for improvement. In subdirectory *Examples* all the book's examples are provided; in subdirectory *Exercises*, one finds our solutions to the exercises compiled at the end of the chapters. In the examples, we concentrated on the respective CORBA concept to be explained and did not extend and elaborate them to simulate development of real-world applications.

We would like to thank Lisa Köblitz and Michael Schneider for testing the example programs. Lisa also helped us create the figures and illustrations included throughout the book. Colleen Litschke carefully read and reread the text and corrected earlier versions; many thanks for improving our English. Finally, special thanks to Dr. Martina Bihn of Springer-Verlag and her team for the reliable successful cooperation, which we meanwhile have experienced for many years. Financial help from the University of Mannheim's Prechel-Stiftung e.V., who supported us in multiple ways, is gratefully acknowledged.

Markus Aleksy, Axel Korthaus, Martin Schader

Malibu, Mannheim, Paris

June 2005

# Contents

<b>1</b>	<b>Preliminaries</b>	<b>1</b>
1.1	Organization of the Book .....	1
1.2	Additional Material .....	2
1.3	Conventions Used in This Book .....	3
1.4	How to Read This Book .....	3
<b>2</b>	<b>Introduction</b>	<b>5</b>
2.1	Object-Oriented Paradigm .....	6
2.2	Distributed Systems .....	8
<b>3</b>	<b>Concepts of the CORBA Standard</b>	<b>13</b>
3.1	Object Management Group .....	13
3.2	Object Management Architecture .....	13
3.3	Common Object Request Broker Architecture .....	15
3.4	Elements of the CORBA Standard .....	15
3.4.1	Object Request Broker .....	15
3.4.2	Object Adapter .....	16
3.4.3	Interface Definition Language .....	18
3.4.4	Interface Repository .....	18
3.4.5	Dynamic Invocation Interface .....	19
3.4.6	Dynamic Skeleton Interface .....	20
3.4.7	Implementation Repository .....	20
3.5	Procedural Steps in Developing a CORBA-Based Application .....	20
3.6	Remote Invocations .....	21
3.7	Interoperability in the CORBA Standard .....	22
3.7.1	Protocols Defined by CORBA .....	23
3.7.2	Interoperable Object Reference .....	24
<b>4</b>	<b>Introduction to the Interface Definition Language</b>	<b>27</b>
4.1	Lexical Elements of IDL .....	27
4.1.1	Comments .....	27
4.1.2	Identifiers .....	28
4.1.2.1	Excursion: Style Guidelines for IDL Identifiers .....	28
4.1.2.2	Excursion: Additional Formatting Rules .....	29
4.1.3	Keywords .....	29
4.1.4	Punctuation Characters .....	30
4.1.5	Preprocessor Directives .....	31
4.1.6	Syntax Notation .....	31
4.2	IDL Types .....	32
4.2.1	Basic Types .....	33
4.2.2	Constructed Types .....	34
4.2.2.1	Structures .....	34

4.2.2.2	Enumerations .....	36
4.2.2.3	Unions.....	36
4.2.3	Excursion: Named Data Types .....	38
4.2.4	Template Types .....	38
4.2.4.1	Fixed Types .....	38
4.2.4.2	String Types.....	39
4.2.4.3	Sequences .....	39
4.2.5	Arrays .....	40
4.2.6	Native Types.....	40
4.2.7	Interfaces .....	41
4.2.8	Value Types.....	41
4.3	IDL Constants.....	41
4.3.1	Literal Constants.....	41
4.3.1.1	Integer Literals.....	41
4.3.1.2	Floating-point Literals .....	42
4.3.1.3	Fixed-point Literals .....	42
4.3.1.4	Character Literals .....	42
4.3.1.5	String Literals .....	44
4.3.1.6	Boolean Literals.....	44
4.3.2	Declaration of Symbolic Constants .....	44
4.3.2.1	Operators .....	46
4.4	Exceptions .....	47
4.5	Interface Declarations .....	48
4.5.1	Attribute Declarations .....	51
4.5.2	Operation Declarations .....	52
4.6	Value Types.....	53
4.7	Module Declarations.....	56
4.8	Scoping .....	57
4.9	Concluding Remarks .....	60
4.9.1	Interoperability .....	60
4.9.2	Using Anonymous Types .....	60
4.10	Exercises.....	61
<b>5</b>	<b>IDL to Java Mapping</b> .....	<b>65</b>
5.1	Introductory Remarks .....	65
5.2	Names .....	65
5.3	Mapping for Basic Data Types .....	66
5.4	Holder Classes .....	67
5.5	Helper Classes .....	69
5.6	Mapping for Modules .....	70
5.7	Mapping for Constants .....	71
5.8	Mapping for typedefs.....	72
5.9	Mapping for structs .....	72
5.10	Mapping for enums.....	74
5.11	Mapping for Sequences .....	75
5.12	Mapping for Arrays .....	75
5.13	Mapping for Exceptions .....	75
5.14	Mapping for Interfaces.....	78

5.14.1	Regular IDL Interfaces.....	78
5.14.2	Local IDL Interfaces .....	80
5.14.3	Abstract IDL Interfaces.....	80
5.15	Mapping for Value Types .....	80
5.15.1	Regular Value Types.....	81
5.15.2	Abstract Value Types.....	82
5.15.3	Boxed Value Types.....	83
5.16	Mapping for anys.....	84
5.17	Mapping for in, inout, and out Parameters.....	86
5.18	Mapping for Attributes .....	86
5.19	Mapping for Operations.....	87
5.20	Exercises .....	87
<b>6</b>	<b>Important Elements of the ORB Runtime</b>	<b>89</b>
6.1	Initializing a CORBA Application.....	89
6.1.1	Operation ORB_init() .....	90
6.2	Pseudo Interface CORBA::ORB.....	90
6.2.1	Operation list_initial_services() .....	91
6.2.2	Operation resolve_initial_references() .....	91
6.2.3	Operations object_to_string() and string_to_object() .....	92
6.2.4	Thread-Related ORB Operations.....	93
6.2.5	Java Mapping of Pseudo Interface CORBA::ORB.....	93
6.3	Portable Object Adapter.....	97
6.3.1	POA Policies.....	98
6.3.2	Overview on POA Functionality .....	99
6.3.3	POA Manager .....	104
6.3.4	Servant Activators .....	106
6.3.5	Servant Locators .....	106
6.3.6	Java Mapping of Interface POA .....	107
6.4	Pseudo Interface CORBA::Object.....	110
6.4.1	IDL Operations of CORBA::Object.....	110
6.4.2	Java Mapping of Pseudo Interface CORBA::Object.....	111
6.5	Pseudo Interface CORBA::TypeCode.....	113
6.6	Dynamic Invocation Interface.....	114
6.6.1	Pseudo Interface CORBA::NamedValue.....	114
6.6.2	Pseudo Interface CORBA::NVList.....	115
6.6.3	Pseudo Interface CORBA::Request .....	116
6.6.4	ORB Operations for the Dynamic Invocation Interface.....	118
6.6.5	Object Operations for the Dynamic Invocation Interface .....	119
6.6.6	Java Mapping of DII-related Pseudo Interfaces and Operations.....	120
6.7	Dynamic Skeleton Interface.....	124
6.7.1	Pseudo Interface CORBA::ServerRequest.....	124
6.7.2	Java Mapping of the DSI .....	125
6.8	Java Class Servant.....	126
6.9	Exercises .....	128

<b>7</b>	<b>A First Example</b>	<b>131</b>
7.1	JDK's IDL Compiler .....	132
7.2	JacORB's IDL Compiler .....	133
7.3	OpenORB's IDL Compiler .....	134
7.4	Recommended File Organization .....	135
7.5	Implementing Counter Using the Inheritance Approach .....	136
7.6	Implementing the Server Application for the Inheritance Approach.....	138
7.7	Compiling the Server Application.....	140
7.8	Implementing the Client Application .....	141
7.9	Compiling the Client Application .....	143
7.10	Running the Application.....	143
7.11	Implementing Counter Using the Delegation Approach .....	144
7.12	Implementing the Server Application for the Delegation Approach.....	145
7.13	A GUI for the Client Application .....	147
7.14	Using Different ORBs .....	149
7.15	Modules .....	149
7.16	Exercises.....	151
<b>8</b>	<b>Generating Remote Objects</b>	<b>153</b>
8.1	Implementing the CounterFactory Servant .....	154
8.2	Implementing the CounterFactory Server.....	156
8.3	Implementing the CounterFactory Client.....	157
8.4	Running the Application.....	159
8.5	Exercises.....	159
<b>9</b>	<b>Alternatives for Designing IDL Interfaces</b>	<b>161</b>
9.1	Attributes vs. Operations .....	161
9.2	Returning Results From an Operation .....	164
9.3	Exercises.....	168
<b>10</b>	<b>Inheritance and Polymorphism</b>	<b>171</b>
10.1	IDL Definition of DateTimeServer .....	172
10.2	Implementing the Inheritance Approach .....	173
10.2.1	Implementing TimeServer.....	173
10.2.2	Implementing DateTimeServer .....	173
10.2.3	Implementing the Server Application.....	174
10.2.4	Implementing the Client Application .....	175
10.3	Implementing the Example with the Delegation Approach.....	177
10.3.1	Implementing TimeServer.....	178
10.3.2	Implementing DateTimeServer .....	178
10.3.3	Modifying the Server Application .....	179
10.4	An Example for Polymorphism.....	180
10.5	Exercises.....	184
<b>11</b>	<b>Implementing Distributed Callbacks</b>	<b>187</b>
11.1	Defining IDL Interfaces .....	188
11.2	Implementing the Counter Servant .....	188
11.3	Implementing the CBCount Server.....	190
11.4	Implementing the CounterClient Servant.....	191



11.5	Implementing the Client Application.....	191
11.6	Further Usages of the Callback Technique .....	194
11.7	Exercise.....	194
<b>12</b>	<b>Utilizing Value Types</b>	<b>197</b>
12.1	Defining IDL Module PublishSubscribe.....	198
12.2	Implementing Value Type Filter .....	199
12.2.1	Implementing the FilterImpl Class .....	201
12.2.2	Using Class FilterDefaultFactory .....	202
12.3	Implementing Class PublisherImpl .....	204
12.4	Implementing the Server Application .....	205
12.5	Implementing Class SubscriberImpl .....	207
12.6	Implementing the Client Application.....	207
12.7	Exercises .....	209
<b>13</b>	<b>Utilizing Interfaces of the DynamicAny Module</b>	<b>211</b>
13.1	Usage of Anys and TypeCodes.....	211
13.2	DynamicAny API.....	214
13.2.1	DynAnyFactory Interface .....	215
13.2.2	DynAny Interface .....	218
13.2.3	DynFixed Interface .....	221
13.2.4	DynEnum Interface .....	221
13.2.5	DynStruct Interface.....	222
13.2.6	DynUnion Interface .....	223
13.2.7	DynSequence Interface.....	223
13.2.8	DynArray Interface .....	224
13.2.9	DynValueCommon Interface.....	225
13.2.10	DynValue Interface.....	225
13.2.11	DynValueBox Interface.....	226
13.3	Usage of the DynamicAny API in Java.....	226
13.3.1	Implementing Servant and Server Application.....	227
13.3.2	Implementing the Client Application .....	229
13.4	Exercises .....	233
<b>14</b>	<b>Dynamic Invocation Interface</b>	<b>235</b>
14.1	Dynamic Counter Client .....	236
14.2	Dynamic TimeServer Clients.....	239
14.2.1	TimeServer Version 1.....	240
14.2.2	TimeServer Version 2.....	241
14.2.3	TimeServer Version 3.....	242
14.2.4	TimeServer Version 4.....	245
14.2.5	TimeServer Version 5.....	246
14.3	Deferred Synchronous Invocations.....	247
14.4	Exercises .....	251
<b>15</b>	<b>Dynamic Skeleton Interface</b>	<b>255</b>
15.1	Defining IDL Module Bank.....	255
15.2	Implementing the Servant .....	256
15.3	Implementing the Server Application .....	259

---

15.4	Implementing the Client Application .....	260
15.5	Exercises.....	261
<b>16</b>	<b>Implementing Different POAs .....</b>	<b>263</b>
16.1	Counter Example .....	264
16.2	Implementing ServantLocator .....	265
16.3	Implementing the Server Application.....	266
16.4	Exercise .....	268
<b>17</b>	<b>CORBA's Naming Service .....</b>	<b>269</b>
17.1	Basics.....	270
17.2	IDL Definition of the Naming Service .....	272
17.3	Bootstrapping Problem .....	274
17.3.1	URL Schemes.....	274
17.3.2	Standard Command-Line Options.....	275
17.4	Binding and Resolving a Name with the Naming Service .....	276
17.4.1	Implementing the Server Application.....	277
17.4.2	Implementing GUIClient .....	278
17.4.3	Starting Naming Service, Server, and Client Applications .....	279
17.4.3.1	Using the JDK .....	280
17.4.3.2	Using JacORB .....	280
17.4.3.3	Using OpenORB.....	281
17.5	Utilizing Naming Contexts.....	282
17.5.1	Server Implementation Version 1 .....	283
17.5.2	Server Implementation Version 2.....	284
17.5.3	Implementing GUIClient .....	285
17.5.4	Running the Application.....	286
17.6	BindingIterators.....	287
17.7	NamingContextExt Interface.....	289
17.7.1	An Example Using the NamingContextExt Interface .....	291
17.7.2	Server Implementation Version 1 .....	291
17.7.3	Server Implementation Version 2.....	292
17.7.4	Implementing GUIClient .....	293
17.8	Concluding Remarks .....	294
17.9	Exercises.....	294
<b>18</b>	<b>CORBA's Event Service .....</b>	<b>297</b>
18.1	Event Service Basics .....	298
18.2	IDL Specification of the Event Service .....	300
18.2.1	Supplier and Consumer Interfaces.....	300
18.2.2	The Event Channel's Administration Interface .....	301
18.2.3	Proxy Interfaces.....	303
18.3	Using OpenORB's Event Service.....	304
18.3.1	Setup and Start of OpenORB's Event Service .....	304
18.3.2	Using OpenORB's ES with JDK's ORB.....	305
18.3.3	Using OpenORB's ES with JacORB.....	306
18.4	Push-Style Publish-Subscribe Example.....	307
18.4.1	IDL Interfaces for the Example .....	307
18.4.2	Implementing the Event Supplier .....	308

---

18.4.3	Implementing the Publisher Application.....	309
18.4.4	Implementing the Event Consumer .....	312
18.4.5	Implementing the Subscriber Application .....	313
18.4.6	Running the Application.....	315
18.5	Exercises .....	316
<b>Appendix A – IDL Grammar</b>		<b>319</b>
<b>Appendix B – IDL to Java: Mapping of IDL Standard Exceptions</b>		<b>325</b>
<b>Appendix C – Naming Service IDL</b>		<b>327</b>
<b>Appendix D – Event Service IDL</b>		<b>329</b>
<b>Appendix E – ORB Product Installation</b>		<b>331</b>
<b>Acronyms</b>		<b>335</b>
<b>References</b>		<b>337</b>
<b>Index</b>		<b>339</b>