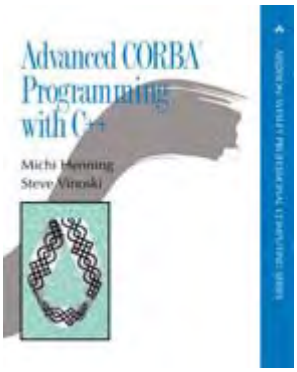

Advanced CORBA® Programming with C++

Michi Henning
Steve Vinoski

Publisher: Addison Wesley
First Edition February 12, 1999
ISBN: 0-201-37927-9, 1120 pages

Book for “Rain Manager”, IT-SC

Enjoy the life together.

Review

Here is the CORBA book that every C++ software engineer has been waiting for. *Advanced CORBA® Programming with C++* provides designers and developers with the tools required to understand CORBA technology at the architectural, design, and source code levels. This book offers hands-on explanations for building efficient applications, as well as lucid examples that provide practical advice on avoiding costly mistakes. With this book as a guide, programmers will find the support they need to successfully undertake industrial-strength CORBA development projects.

The content is systematically arranged and presented so the book may be used as both a tutorial and a reference. The rich example programs in this definitive text show CORBA developers how to write clearer code that is more maintainable, portable, and efficient. The authors' detailed coverage of the IDL-to-C++ mapping moves beyond the mechanics of the APIs to discuss topics such as potential pitfalls and efficiency. An in-depth presentation of the new Portable Object Adapter (POA) explains how to take advantage of its numerous features to create scalable and high-performance servers. In addition, detailed discussion of advanced topics, such as garbage collection and multithreading, provides developers with the knowledge they need to write commercial applications.

Other highlights:
In-depth coverage of IDL, including common idioms and design trade-offs
Complete and detailed explanations of the Life Cycle, Naming, Trading, and Event Services
Discussion of IIOP and implementation repositories
Insight into the dynamic aspects of CORBA, such as dynamic typing and the new DynAny interfaces
Advice on selecting appropriate application architectures and designs
Detailed, portable, and vendor-independent source code

Library of Congress Catalog-in-Publication Data
Henning, Michi
Advanced CORBA® Programming with C++ / Michi Henning, Steve Vinoski.
p. cm. — (Addison-Wesley professional computing series)
Includes bibliographical references and index.
ISBN 0-201-37927-9
1. C++ (Computer program language) 2. CORBA (Computer architecture)
I. Vinoski, Steve. II. Title. III. Series.
QA76.73.C153 H4581999
005.13'3—dc21
98-49077
CIP

Published simultaneously in Canada.
Text printed on recycled and acid-free paper.
4 5 6 7 8 9 10—CRS—0302010099
Third printing, August 1999

Dedication

To Anni and Harry, for setting me on the path.
To Jocelyn, for letting me follow it.
—Michi
To Cindy, my wife and best friend—for your sacrifices, your support, your patience, and your love.
—Steve

Preface

For years, both of us have been (and still are) teaching CORBA programming with C++ to software engineers all over the world. One of the most frequently asked questions in our courses is, "Where can I find a book that covers all this?" Although many books have been written about CORBA, most of them focus on high-level concepts and do not address the needs of software engineers. Even though CORBA is conceptually simple, the devil lies in the detail. Or, more bluntly, books focusing on high-level concepts are of little use when you must find out why your program is dumping core.

To be sure, there are resources available about CORBA, such as newsgroups, Web pages, and the Object Management Group (OMG) specifications. However, none of them really meets the needs of a programmer who must get the code to work (and preferably by yesterday). We wrote this book so that there would finally be a tutorial and reference that covers CORBA programming with C++ at the level of detail required for real-life software development. (And, of course, we wrote it so that we would have a good answer for our students.)

Writing such a book is a tall order. Explaining the CORBA specification and APIs is one thing, and it's a necessary part of the book. However, knowing the various APIs will not, by itself, make you a competent programmer (only a knowledgeable one). To be competent, you need not only knowledge of the mechanics of the platform but also an understanding of how the different features interact. You must combine them effectively to end up with an application that performs and scales well and is maintainable, extensible, portable, and deployable.

To help you become competent (as opposed to merely knowledgeable), we go beyond the basics in a number of ways. For one thing, we provide advice as to what we consider good (and bad) design, and we make no attempt to hide problems with CORBA (which, like any other complex software system, has its share of wrinkles). Second, we go beyond the APIs by explaining some of CORBA's internal mechanisms. Even though you can use an ORB without knowing what goes on under the hood, it is useful to understand these mechanisms because they have a profound influence on how well (or how poorly) an application will perform. Third, we devote considerable space to a discussion of the merits of various design decisions; typically, when a design provides a gain in one area it also involves a loss in another. Understanding these trade-offs is crucial to building successful applications. And fourth, where appropriate, we make recommendations so that you are not left without guidance.

Inevitably, our approach required us to make value judgments, and, just as inevitably, a number of people will disagree with at least some of the recommendations we make. Whether you agree or disagree with us, you should still profit from our approach: if you agree, you can stick to the advice we give; if you disagree, the discussion will have at least encouraged you to think about the topic and form your own opinion. Either way,

you are better off than you would be with a book that just dumps the facts on you without providing the deeper insight required to use them.

Prerequisites

This book is not a beginner's book, in the sense that we do not devote much space to explaining the structure of the OMG or the specification adoption process. We also do not provide a high-level overview of the architectural goals of CORBA or all its services and facilities (see [31] for a high-level overview). Instead, we assume that you want to know how to write real CORBA applications with C++. Despite the lack of overview material, you should be able to follow the material even if you have never seen CORBA before. If you have experience in network programming or have used another RPC platform, you will find it easy to pick things up as you go.

Much of this book consists of source code, so we expect you to be literate in C++. However, you do not need to be a C++ guru to follow the code. We have avoided obscure or little-understood features of C++, preferring clarity to cleverness. If you understand inheritance, virtual functions, operator overloading, and templates (not necessarily in intricate detail), you will have no problems. Some of the source code uses the Standard Template Library (STL), which is now part of the ISO/IEC C++ Standard. We have limited ourselves to very simple uses of this library, so you should be able to understand the source code even if you have never seen STL code before.

If you have never written threaded code, you will find the chapter on writing threaded servers tough going. Unfortunately, there was not enough room to provide an introduction to programming with threads. However, the Bibliography lists a number of excellent books on the topic.

Despite our best efforts to show realistic and working source code, we had to make a number of compromises to keep code examples understandable and of manageable size. When we demonstrate a particular feature, we often use straight-line code, whereas in a realistic application the code would better be encapsulated in a class or helper function. We have also minimized error handling to avoid obscuring the flow of control with lots of exception handlers. We chose this approach for didactic purposes; it does not imply that the code pretends to reflect best possible engineering practice. (The Bibliography lists a number of excellent books that cover source code design in great detail.)

Scope of this Book

OMG members are continually improving CORBA and adding new features. As a result, available ORB implementations conform to different revision levels of the specification. This book covers CORBA 2.3. (At the time of writing, CORBA 2.3 is being finalized by the OMG.) Throughout the text, we indicate new features that may not yet be available in your ORB implementation; this allows you to restrict yourself to an earlier feature set for maximum portability.

Despite its size, our main regret is that this book is too short. Ever-increasing page counts and ever-closer deadlines forced us to drop chapters on the Dynamic Invocation Interface (DII), the Dynamic Skeleton Interface (DSI), and the Interface Repository (IFR). Fortunately, the vast majority of applications do not need those features, so dropping these chapters is not much of a loss. If your application happens to require the dynamic interfaces, the background we provide here will enable you to easily pick up what you need from the CORBA specification.

Another feature notable by its absence is Objects-By-Value (OBV). We chose not to cover OBV because it is too new for anyone to have any substantial engineering experience with it. In addition, at the time of writing, there are still a number of technical wrinkles to be ironed out and we expect the OBV specification to undergo further changes before it settles down.

Size and time limitations also meant that we could not cover every possible CORBA service. For example, we did not cover the Transaction Service or Security Service because each of them would require a book of its own. Rather than being complete, we have restricted ourselves to those services that are most essential for building applications: the Naming, Trading, and Event Services. We cover those services in more detail than any other publication we are aware of.

An important part of this book is the presentation of the Portable Object Adapter (POA), which was added in CORBA 2.2. The POA provides the server-side source code portability that was missing from the (now deprecated) Basic Object Adapter. The POA also provides a number of features that are essential for building high-performance and scalable applications. We have therefore paid particular attention to showing you how to use the POA effectively in your designs.

Overall, we believe this book offers the most comprehensive coverage to date of CORBA programming with C++. We have arranged the material so that you can use the book both as a tutorial and as a reference. Our hope is that after the first reading, you will have this book open at your side when you are sitting at your terminal. If so, we will have achieved our goal of creating a book that is used by real engineers to build real applications.

Acknowledgments

As with any book, the authors are only part of the story, and this is the place to thank the large number of people who have contributed to making this book possible. At Addison Wesley Longman, Mike Hendrickson and our editor, Deborah Lafferty, believed us when we told them that this book needed to be written. Without their faith in us, you would not be reading this. Brian Kernighan reviewed several drafts and made us redo the job where necessary. His clarity of thought and critical eye have greatly improved this book. John Fuller and Genevieve Rajewski, our production editors, put up with all our naive questions and enabled two amateurs to take a book to camera-ready stage. Our copy editor, Betsy Hardinger, edited every page in this book with meticulous attention to detail. Her efforts taught us more about clarity of style than we thought possible.

Particular thanks go to Colm Bergin, Jonathan Biggar, Bart Hanlon, Jishnu Mukerji, and Doug Schmidt, our expert reviewers. They read the entire manuscript and spotted many problems that would have otherwise gone unnoticed. Their comments kept us honest throughout. Alan Shalloway reviewed the book from the perspective of a newcomer and made valuable suggestions on how to improve the presentation of some of the more difficult topics.

Todd Goldman and Tim Gill from Hewlett-Packard gave us permission to draw on earlier ORB training material written by Michi. John Vinoski and Dan Rabideau of Green Bay Engraving take credit for designing the Möbius strip on the cover.

We are grateful to Steve's employer, IONA Technologies, for allowing us to use the next generation of their Orbix product (called "ART") to develop and test our code examples. Their generosity provided us with the opportunity to make sure that our examples were correct and functional. The fact that ART conforms to CORBA 2.3 allowed us to target the most recent version of the CORBA specification available as of this writing.

We also would like to thank the many contributors to comp.object.corba and the corba-dev mailing list. The discussions there have influenced much of the content of this book.
A number of people have provided feedback, corrections, and constructive criticism since the first printing of this book. Rather than list them all here (and have to keep updating this Preface for each new printing), we have placed a list of everyone who contributed at <http://www.awl.com/cseng/titles/0-201-37927-9>. Our thanks go to all these people for helping to make this a better book.

Michi's Acknowledgments

I would like to thank my former employer, DSTC Pty Ltd, for providing me with an environment that was conducive to writing. Joachim Achtzehnter, Martin Chilvers, Wil Evers, Ted McFadden, and Michael Neville reviewed parts of the manuscript and made valuable suggestions for improvement. Particular thanks go to David Jackson, who read all my drafts and made sure that loose ends were not allowed to remain hanging. Finally, I would like to thank my wife, Jocelyn, and our son, Tyson, for their love and

encouragement. Without their support and patience, this book would have never been written.

Steve's Acknowledgments

I would like to thank my employer, IONA Technologies, for supporting my efforts to write this book, which occasionally kept me away from the office. In particular, I would like to thank Barry Morris for his support and encouragement, Stephen Keating for taking up the slack when I had to miss work because of all-night writing sessions, and the whole IONA Boston product development team for their patience and support.

I would also like to thank Bart Hanlon, who not only reviewed this book but also was my manager at my former employer, for continually encouraging me for several years to tackle this project and for teaching me a lot about tackling projects in general. In the technical realm, I have learned from many people over the course of my career, but I owe much to John Morris, Craig Bardenheuer, Denis deRuijter, Dale LaBossiere, Tom Moreau, and Bob Kukura, who at one time or another greatly influenced my education in the realms of distributed systems and engineering in general. I would also like to thank my *C++ Report* co-columnist, Doug Schmidt, a true technical visionary whose work in object-oriented network programming, C++ frameworks, and CORBA has paved the way for books such as this one. He not only helped review this book, but also agreed to let me use material from our columns in writing it.

Finally, without the support of my family, I would have never been able to even consider writing this book. I'd like to thank my wife, Cindy, and our children, Ryan and Erin, for putting up with my extremely long hours and days of writing and working. Thanks also to my parents, Ed and Dooley, who have always supported me with their seemingly limitless patience and love. I also owe my brother, John, a special thanks for his wonderful artwork on our book cover.

Michi Henning and Steve Vinoski

October 1998

Chapter 1. Introduction

1.1 Introduction

CORBA (Common Object Request Broker Architecture) is now well established in the mainstream of software development and has found phenomenal industry acceptance. CORBA is supported on almost every combination of hardware and operating system in existence. It is available from a large number of vendors (even as freeware), supports a large number of programming languages, and is now being used to create mission-critical applications in industries as diverse as health care, telecommunications, banking, and manufacturing. The increasing popularity of CORBA has created a corresponding increase in demand for software engineers who are competent in the technology.

Naturally, CORBA has had to evolve and grow (sometimes painfully) to reach its current levels of popularity and deployment. When the first version of CORBA was published in 1991, it specified how to use it only in C programs. This was a result of building CORBA from proven technology. At that time, most production-quality distributed systems were written in C.

By 1991, object-oriented (OO) languages such as Smalltalk, C++, and Eiffel had been in use for years. Not surprisingly, many developers thought it strange that a language-independent distributed OO system such as CORBA could be programmed only using C, a non-OO, procedural language. To correct this short-coming, several development groups at companies such as Hewlett-Packard, Sun Microsystems, HyperDesk Corporation, and IONA Technologies started developing their own proprietary mappings of CORBA to the C++ language. These proprietary mappings of CORBA to C++, all invented independently, differed in many ways. As most C++ programmers know, C++ is a multiparadigm language that supports varied approaches to application development, including structured programming, data abstraction, OO programming, and generic programming. The proprietary C++ mappings reflected this diversity; each of them mapped different CORBA data types and interfaces into different (sometimes very different) C++ types and classes. The mapping differences reflected not only the varied backgrounds of the developers but also the ways they intended to use CORBA to build systems as diverse as software integration middleware, operating systems, and even desktop tool kits.

When the Object Management Group (OMG) issued a Request For Proposals (RFP) for a standard mapping of CORBA to C++, these developers and other groups submitted their mappings to the standardization process. As is common for OMG RFP submissions, the submitting groups joined forces to try to reach consensus and arrive at a single C++ mapping specification that would draw from the strengths of all the submitted mappings. The process of producing a single standard C++ mapping for CORBA took approximately 18 months, lasting from the spring of 1993 until the fall of 1994. For technical reasons, such as the richness of C++ and its support for diverse programming styles, the consensus-building process was not an easy one. At one point, because of the competitive spirit and political nature of some of the parties involved (both characteristics are inevitable in any industry standards group), the C++ mapping standardization effort fell apart completely. However, the need for a standard C++ mapping eventually overcame all obstacles, and the standardization was completed in the fall of 1994.

The C++ mapping was first published with CORBA 2.0. Since its adoption, the mapping has been revised several times to fix flaws and to introduce minor new functionality. Despite this, the mapping has remained surprisingly stable and portable even while the C++ language was undergoing its own standardization process. The standard C++ mapping removed a major obstacle to broad acceptance of CORBA because it created source code portability, at least for the client side. The server side still suffered from portability problems until CORBA 2.2.

CORBA 2.0 also removed another major obstacle by providing the Internet Inter-ORB Protocol (IIOP). IIOP guarantees that system components developed for different vendors' ORBs can interoperate with one another, whereas before CORBA 2.0, different system components could communicate only if all of them used the same vendor's ORB.

The C++ mapping and IIOP were key features that initiated CORBA's move into the mainstream and made it a viable technology for many commercial companies. This increased popularity of CORBA also meant an increased demand for extensions and bug fixes. As a result, the specification has been revised three times since the publication of CORBA 2.0. CORBA 2.1 was largely a cleanup release that addressed a number of defects. CORBA 2.2 added one major new feature: the Portable Object Adapter (POA). The POA, together with an update to the C++ mapping, removed the server-side portability problems that existed to that point. CORBA 2.3, the most recent release, as of this writing, fixed many minor bugs and added one major new feature, Objects-By-Value.

The OMG has now grown to more than 800 members, making it the world's largest industry consortium, and CORBA has become the world's most popular and widely used middleware platform. In our estimation, C++ is the dominant implementation language for CORBA (although Java is making some inroads for client development). Demand for CORBA-literate C++ programmers continuously outstrips supply, and it seems likely that CORBA will remain the dominant middleware technology for at least several more years. This book is all about making you CORBA-literate and giving you the information you need to be able to write production-quality CORBA-based systems.

1.2 Organization of the Book

The book is divided into six parts and two appendices.

Part I, Introduction to CORBA, provides an overview of CORBA and presents the source code for a minimal CORBA application. After reading this part, you will know the basic architecture and concepts of CORBA, understand its object and request dispatch model, and know the basic steps required to build a CORBA application.

Part II, Core CORBA, covers the core of CORBA with C++: the Interface Definition Language (IDL), the rules for mapping IDL into C++, how to use the POA, and how to support object life cycle operations. This part introduces the case study we use throughout this book; following each major section, we apply the material presented there to the case study so that you can see how the various features and Application Programming Interfaces (APIs) are used for a realistic application. After reading this part, you will be able to create sophisticated CORBA applications that exploit many CORBA features.

Part III, CORBA Mechanisms, presents an overview of the CORBA networking protocols and shows the mechanisms that underpin CORBA's object model, such as location transparency and protocol independence. After reading this part, you will have a good idea of what goes on beneath the hood of an ORB and how design choices made by various vendors influence a particular ORB's scalability, performance, and flexibility.

Part IV, Dynamic CORBA, covers dynamic aspects of CORBA: type `any`, type codes, and type `DynAny`. After reading this part, you will know how you can use these CORBA features to deal with values whose types are not known at compile time. This knowledge is essential for building generic applications, such as browsers or protocol bridges.

Part V, CORBAservices, presents the most important CORBA services, namely the Naming, Trading, and Event Services. Almost all applications use one or more of these services. The Naming and Trading Services allow applications to locate objects of interest, whereas the Event Service provides asynchronous communication so that clients and servers can be decoupled from each other. After reading this part, you will understand the purpose of these services and you will be aware of the architectural consequences and trade-offs implied by their use.

Part VI, Power CORBA, discusses how to develop multithreaded servers and presents a number of architectural and design issues that are important for building high-performance applications.

`Appendix A` shows the source code for an instrument control protocol simulator that you can use if you want to experiment with the source code in this book.

`Appendix B` contains a list of useful resources you can use to get more information about various aspects of CORBA.