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.