

CORBA Development in C++ Environment

Course Summary

Description

This course introduces Common Object Request Broker Architecture (CORBA). CORBA is the technology that allows distributed software system to communicate. Programming the C++ interface to CORBA allows a C++ program to invoke a Java program and receive a response. This course introduces the student to the C++ mapping of CORBA types. Custom mappings are covered, as are exceptions and event programming.

Topics

- Introduction
- The OMG Interface Definition Language (IDL)
- Basic IDL to C++ Mapping
- Client-Side C++ Mapping
- Server-Side C++ Mapping
- The Portable Object Adapter (POA)
- Object Life Cycle
- GIOP, IIOP, and IORs
- Implementation Repositories and Binding
- Dynamic CORBA
- CORBA services

Audience

This course is ideal for the developer working in the C++ language. In addition other members of the development staff such as managers, quality assurance and testing personal would find this course very useful.

Prerequisites

Students of this course should be experienced C++ programmers.

Duration

Four days

CORBA Development in C++ Environment

Course Outline

- I. Introduction**
 - A. Overview of CORBA
 - B. The OMG
 - C. CORBA features
 - D. Request invocation
 - E. Minimal CORBA application
- II. The OMG Interface Definition Language (IDL)**
 - A. Overview
 - B. Compilation
 - C. Source files
 - D. Basic IDL types
 - E. User-defined types
 - F. Interfaces and operations
 - G. Exceptions
 - H. Contexts
 - I. Attributes
 - J. Modules
 - K. Names and scoping
 - L. Forward declarations
- III. Basic IDL to C++ Mapping**
 - A. Mapping for identifiers
 - B. Mapping for modules
 - C. The CORBA modules
 - D. Mapping for basic types
 - E. Mapping for constants
 - F. Mapping for enumerated types
 - G. Variable length types
 - H. The String_var wrapper class
 - I. Mapping for structures
 - J. Mapping for arrays
 - K. User-defined types and _var classes
- IV. Client-Side C++ Mapping**
 - A. Mapping for interfaces
 - B. Object reference types
 - C. Life cycle of object references
 - D. Pseudo-objects
 - E. ORB initialization
 - F. Initial references
 - G. Stringified references
 - H. Mapping for operations and attributes
 - I. Mapping for exceptions
 - J. Mapping for contexts
- V. Server-Side C++ Mapping**
 - A. Mapping for interfaces
 - B. Servant classes
 - C. Object incarnation
 - D. Server main
 - E. Parameter passing rules
 - F. Raising exceptions
 - G. Tie classes
- VI. The Portable Object Adapter (POA)**
 - A. POA fundamentals
 - B. POA policies
 - C. POA creation
 - D. Servant IDL types
 - E. Object creation and activation
 - F. Reference, objectId, and servant
 - G. Object deactivation
 - H. Request flow control
 - I. ORB event handling
 - J. POA activation
 - K. POA destruction
- VII. Object Life Cycle**
 - A. Object factories
 - B. Destroying, copying, and moving objects
 - C. The evictor program
 - D. Garbage collection
- VIII. GIOP, IIOP, and IORs**
 - A. Overview of GIOP
 - B. Common data representation
 - C. GIOP message formats
 - D. GIOP connection management
 - E. Overview of IIOP
 - F. Bidirectional IIOP
 - G. Structure of IOR
- IX. Implementation Repositories and Binding**
 - A. Binding modes
 - B. Direct binding
 - C. Indirect binding via an implementation repository
 - D. Activation modes
 - E. Race conditions
 - F. Security considerations

X. Dynamic CORBA

- A. C++ mapping for type any
- B. Type codes
- C. Type DynAny interface

XI. CORBA services

- A. The OMG naming service
- B. The OMG trading service
- The OMG event service