Application Servers
G22.3033-011

Session 4 - Sub-Topic 3
Introduction to CORBA, DCOM, and RMI

Dr. Jean-Claude Franchitti

New York University
Computer Science Department
Courant Institute of Mathematical Sciences

Contents

- Introduction to CORBA, DCOM and RMI
- Overall architectural framework
- Architectures
- Programming layers
- Benefits of each technology
- Conclusions
- References

CORBA

- Object-oriented middleware architecture for heterogeneous distributed systems
- Governed by OMG, > 700 companies
- Standardized specification
- Large number of commercial and free implementations
- Revision 2.0 (1995) allowed implementations to interoperate
CORBA Components:

- Object model
- IDL
- Language mappings: C, C++, Smalltalk, Cobol, Ada, Java (soon)
- ORB for basic services
- Interoperability through GIOP and IIOP
- Services and Facilities: useful standard interfaces

COM (Component Object Model)

- Microsoft’s Object Framework
  - Widespread use on desktop (Windows)
- Binary standard, complicated
- Functionality provided in an interface
- Interface pointer is like a C++ virtual function pointer
- An object can support multiple interfaces
- COM+ on the works (simpler to program)

COM Interface pointer
DCOM

- Distributed extension to COM
- Governed by Microsoft (Open Group)
- Emerged with NT 4.0
- Built on top of DCE RPC
- Software AG port to non-Window platforms

RMI

- Remote Method Invocation for Java objects
- Standard from JDK 1.1
  - Not all Java licensees support it...
- CORBA-like standard services to come later
- Applet communication

RMI’s Stated Goals:

- Support seamless remote invocation on objects in different virtual machines.
- Support callbacks from servers to applets.
- Integrate the distributed object model into the Java language in a natural way while retaining most of the Java language’s object semantics.
- Make differences between the distributed object model and local Java object model apparent.
- Make writing reliable distributed applications as simple as possible.
- Preserve the safety provided by the Java runtime environment.
RMI Architecture

Virtual Machine 1
- `stub`/`skeleton`
- Remote referenced
- Transport
- Security

Virtual Machine 2
- `stub`/`skeleton`
- Remote referenced
- Transport
- Security

Bootstrap registry

---

Programming Layer (CORBA, DCOM)

- **CORBA**
  - Define interface in IDL
  - Generate stub/skeleton
  - Derive and implement server skeleton, typically by inheritance
  - Server main program creates new object and signals CORBA with
    `IDL.generate()`
  - Client binds to remote object and makes the call

- **DCOM**
  - Define interface in IDL
  - Inherit from `IDeclare`/`IServiceProvider`
  - Generate proxy/ID
  - Implement server by class factory
  - Take care of reference counting
  - Server main program creates new class factory, sends an event and
    waits until objects get deleted by
    COM Server
  - Client uses `CoCreateInstance()` to create interface pointer
  - QueryInterface() to access additional interfaces
  - Make calls through interface pointer
Programming Layer (RMI)

- Define interface & Java (extend java.rmi. Be use of interface)
- Java implementation
  - stateless, stateful, debate RemoteObject
  - implement away from code
  - install server/manager
  - create one or more instances
  - register remote object in local registry
- Write client/app or application
  - Get reference from registry
  - Compile source files
- Generate stubs and skeletons with idjc
- Debug and deploy

Benefits of CORBA

- Wide support for heterogeneous computing
  - Vendors
  - Operating systems
  - Programming languages
- Easy programming model
- Supports multiple interface inheritance
- Proven interoperability

Useful standard service specifications:
- Naming
- Events
- Life Cycle
- Persistence
- Transactions
- Concurrency
- Externalization
- Security
- Etc.
- (But adoption by vendors is slow)

Benefits of DCOM

- Large user base and component market
- Binary software integration
  - Large-scale software reuse (no source code)
  - Cross-language software reuse
  - COFF tools integration
- On-line software update
  - Allow updating a component in an application without recompilation, rebuilding, or even restarting

- Multiple interfaces per object
- Wide selection of programming tools available
  - Automation of standard code
Benefits of RMI

- Single language
- Smooth integration into Java
- Free
- Easy programming model
- Cross-platform execution in browsers
- Multiple interfaces per object
- True polymorphic computing capability
  - client can transfer code for execution on the server

Conclusions

- Basically similar architectures
- Programming is different
  - Will become quite similar with COM+
  - Multiple inheritance is different
- CORBA allows different ways to implement the specification
- Strong areas and markets exist for every three technologies
  - CORBA: enterprise
  - DCOM: desktop
  - RMI: internet
- Object wars continue
References