ECE 451/566- Introduction to Parallel and Distributed Programming

Distributed Computing: Distributed Objects/Services

Manish Parashar
parashar@ece.rutgers.edu
Department of Electrical & Computer Engineering
Rutgers University
Distributed Object Computing

- Enable any “agent” in the local system to directly interact with an object that resides on a remote host.
  - CORBA
  - RMI (Java)

- Distributed Objects versus Message Passing
  - Simplicity
  - Robustness
  - Manageability
  - Extensibility
Distributed Objects: Goals

- Let any object reside anywhere in the network, and allow an application to interact with these objects exactly in the same way as they do with a local object.
- Provide the ability to construct an object on one host and transmit it to another host.
- Enable an agent on one host to create a new object on another host.
Remote and local method invocations

Diagram showing remote and local method invocations between nodes A, B, C, D, E, and F.
A remote object and its remote interface
Distributed Objects: Issues

- Creating remote objects
- Remote method calls
  - get class reference
  - call constructor with arguments
  - get reference to new object
  - get method reference
  - call method with arguments
Distributed Objects: Issues

- Data marshalling
- Arguments by value/by reference
- Consistency across multiple clients/servers
- Balancing work-load
Middleware layers

- Applications
- RMI, RPC and events
- Request reply protocol
- External data representation
- Operating System
Middleware: Transparency

- Location
- Communication protocol
- Computer hardware
- Operating system
- Programming language
- ...
Object Interface Specification

- Platform/language independent means for specifying object interfaces
- Compiled to server skeletons and client stubs
  - Interface definition language (IDL)
  - Inter-language Unification (LU)
  - Component Object Model Language (COM)
Example – Interface Definition

// In file Person.idl
struct Person {
    string name;
    string place;
    long year;
};

interface PersonList {
    readonly attribute string listname;
    void addPerson(in Person p);
    void getPerson(in string name, out Person p);
    long number();
};
Object Manager

- Heart of the distributed object system
  - manage object skeletons and object references on the object server
    » CORBA ORB, RMI Registry
  - on remote object creation request
    » locate skeleton, create new object, store new object, return reference
  - on remote method call
    » route request to object, return result
  - on remote object destruction request
    » remove object from store
Object Manager

- Special services
  - dynamic object activation
    » invoke active/deactivate methods provided by the object based on remote request
  - object persistence
    » store and restore object state
- Object manager may reside on server, both client and server, or on a separate host
Registration/Naming Service

- Acts as an intermediary between object client and manager
  - object interface is registered with the service
  - client queries service by name/type/…
  - naming service routes request to the appropriate object server
  - service can also be used for routing method invocations
Other Issues

- Object Communication Protocol
- Development Tools
- Security
The role of proxy and skeleton in remote method invocation

![Diagram showing the role of proxy and skeleton in remote method invocation.]

- **Client**
  - Object A
  - Proxy for B
  - Remote reference module
  - Communication module

- **Server**
  - Remote object B
  - Skeleton & dispatcher for B’s class
  - Remote reference module
  - Communication module

- **Communication**
  - Request
  - Reply
Role of client and server stub procedures in RPC
CORBA

- Common Object Request Broker Architecture
  - Generic framework for developing distributed object systems
  - Platform independent/language independent
CORBA

- Object Request Broker (ORB)
  - enable clients and server objects to interact
  - provide servers
    » naming services, security services, …
- Interface Definition Language (IDL)
- Dynamic Invocation Interface
- Internet Inter-Orb Protocol (IIOP)
CORBA Architecture

CORBA overall architecture
CORBA components

- Client Idl stubs
- Interface definition language
- Dynamic invocation interface
- Static Idl skeletons
- dynamic skeletons
- Interface repository
- Object request broker
Some CORBA services(i)

- Life-cycle service
- Persistence service
- Event service
- Naming service
- Concurrency control service
- Relationship service
Some CORBA services(ii)

- Externalisation service
- Query service
- Licensing service
- Properties service
- Time service
- Security service
- Trader service
module Tester{

   interface Single{
      attribute string exname;
      readonly attribute string location;
      string returnsVals(in string point);
   }

}


A CORBA application

- Client code
- Server code
- Stub code
- Skeleton code
- ORB
- Transport mechanism
Developing a CORBA application

- Write the IDL descriptions
- Process IDL
- Develop the remote object classes
- Develop server code
- Develop client code
- Deploy
The main components of the CORBA architecture
Java RMI

- Remote Object Interfaces
- Server Implementations
- RMI Registry
- Client Stubs and Server Skeleton
- Serializing Objects
RMI V/s CORBA

- Language barrier
- Virtual Machine
- Efficiency
- Elegance
- Ease-of-use
- Communication model
- Support