Java Distributed Computing

IS 313
5.27.2003
Outline

- Homework #3
- Distributed computing overview
- Programming RMI
Homework #3

- Good news
  - Ave. 13.8
- Bad news
  - Some people didn’t turn in
Distributed computing

- **Sockets**
  - Messages from A to B

- **Distributed computing**
  - A is doing some computation P
  - A asks B to compute some function F
Sockets
Distributed Computing

Server

Codebase

function F

Client

Application

?
Questions to ask

How does...

1. P know how to call F (method signature, host identity)
   A. When does user/programmer of P need to know

2. P send its message to B and B send the answer back

3. B handle F so it can be invoked

4. The system provide quality of service, security and management
Distributed comp. approaches

- (RPC)
- RMI
- DCOM
- CORBA/JIDL
- Web services
RMI

1. P and F are written in Java
   - F’s interface is available to P at compile time
   - Host determined at run-time
2. Standard part of Java API
3. RMI registry must be running on B
4. Not much
RMI
DCOM

1. P and F are implemented in COM
   - A type library for F is available at compile time
   - Destination is encoded in registry at run time
2. Part of Windows OS
3. Loads and runs autonomously
4. Some MS tools
DCOM
CORBA

1. An IDL description of F is available at compile time
   - The ORB uses inter-ORB communication to find host dynamically
2. Request sent to ORB
3. Destination ORB activates and runs
4. ORB vendors supply
CORBA
Web services

1. P gets XML description of F
   - This can happen at run-time
   - Service registry knows what hosts offers the service
2. HTTP/SOAP interaction
3. Web application server invokes F
4. Emerging
Web services
Trade-offs

- RMI
  - Java only
- DCOM
  - Windows only
- CORBA
  - Must buy ORB software
- Web services
  - ??
RMI

Server

Java object
remote method
RMI registry

Client

Stub Application

RMI protocol
Elements of an RMI application

- Remote object
  - contains the methods we want to call
- RMI server
  - a program that exposes the remote object
- RMI registry
  - built-in intermediary
- RMI client
  - program that accesses the remote object
RMI Elements
Remote Object

- Must implement a remote interface
- UnicastRemoteObject convenience class
Remote Interface

- Remote object’s signature
- Must extend java.rmi.Remote
- All methods must throw RemoteException
RMI Registry

- Distributed with the JDK
- No application-specific arguments
RMI Server

- Program creates instance of Remote Objects
- Names and publishes them in the RMI registry
- Must set security policy to allow remote connection
RMI Client

- Compiled using the Remote Interface
- Contacts remote registry
- Downloads stub for remote object
- Calls methods on the stub
Stub

- conforms to the remote interface, but
- connects to remote JVM
- marshals the parameters for F
- waits for the result of F,
- unmarshals the return value or exception returned, and
- returns the value to the caller.
Registration
Lookup
Method Call
Serialization

- Alternative to remote object
- Transmits the object itself
- Objects passed as parameters to remote methods
- `implement java.io.Serializable`
  - no code necessary
RMI Example

- WeatherInfo
  - serialized class
- IWeatherData
  - remote interface
- WeatherData
  - remote object
- WeatherServer
  - RMI server
- WeatherInfoClient
  - RMI client
Deployment

- Distribute
  - remote interfaces
  - definitions of serializable classes
- Run rmic to generate stubs for remote objects
- Run registry
- Run RMI server