CS 461 Distributed Computing

Assignment 2 Help Session
Rege Colwell
October 9, 1997

What we give you

- a working copy of ReliableNI (I think)
- ServiceAddress & PacketTypeCode
  - A NetAddress (machine+port) & Service #
  - Constants for packet Types
- Service, ServiceThread, EchoService
  - Code to start your own service and an Example
  - New Service = extend service & override servConn

You must Implement

- ClientServerNI
  - Lets many clients connect() to services
  - binds() servers so clients can connect()
- Connection
  - 2 Ends connected by 2 streams

New Features

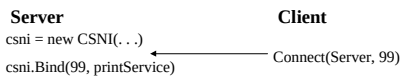
- Connection Based
  - No more mailboxes, 2 way connection
- Services
  - Create new threads for each connection
- Dead Connection Detection - You Still there?!
  - Ping - pong

Client Server Network Interface

- Do **NOT** inherit from ReliableNI, NetworkInterface, or anything else!
- Use a ReliableNI as a **private** variable
- Do bookkeeping on *services* offered and manage *connections*
- Each CSNI has many services and connections

Public CSNI Methods (Required)

- *Constructors, getAddress*
- *bind/unbindService*
  - Manipulate Data Structure that keeps track of bindings
- *Enable* - Avoids false rejects in server startup



  - **Common Mistake: Don't forget to call enable**
- *Run* - Get packet, deal with it, get packet, …

Connections

- One end of a phone connection
- Anything put into connection comes out the other end
- unique id number (per CSNI)
- Each two-way connection has 2 id numbers (different number on either side)
  - Protocol says: use recipient's ID (accepts and connect slightly different)

Connection Methods

- Constructors - Does not create 2 way connection
- getInputStream() - stream to get data from other end
- getOutputStream() - stream to send stuff out to other end
  - Don't forget to Flush!
- getPeerAddress() - address of other end of the connection

Data Streams

- Make A B C D look like "ABCD"
- or "WXYZ" into W X Y Z
- Need to make new kind of InputStream that can read from multiple packets
- Output stream must be able to break data into packet sized chunks

Useful Java Things

- Classes: hashTable, Vector
  - Both in java.util
  - Vector is "dynamic array"
  - Check out java API under misc. on homepage
- Class variables are global to a class
  - *static* int iNum;
- Static Initializers - run when class is init'ed
  - static { }

Keep your grader happy

- index.html
- How to compile
- How to run your test(s)
- Links to source files