CMSC433, Spring 2002 Programming Language Technology and Paradigms Distributed Computing & RMI

Alan Sussman

March 12, 2002

Administrivia

- Project 3 due tomorrow
- Commentary for project 2 due next Wed.
  - submit as project 12
- Project 4 posted by Thursday
  - distributed computing with RMI
- Read Ch. 15, pages 973-979, in Eckel
  - Readings web page also has additional distributed computing reading
- Talk about exam Thursday
- Back to project 2 discussion

Is distributing computing different?

- What kinds of distributed computing environments exist?
- Ways in which distributed computing is different
  - Addressing objects
  - Latency
  - Partial failure
  - Concurrency

Distributed computing environments

- Usually refers to multiple CPU's
- Interprocessor communication via shared address space or message passing?
- On same chip, in same room, or across the Internet?
  - latency, failure modes

Existing environments

- Seti @ Home, Entropia
- Server for search engine
- My laptop, PDA, cell phone, MP3 player, digital camera, …

Types of failure

- Machine sleeps
  - wakes up, recovers state
- Machine crash or failure
  - machine may reboot and rejoin
- Network partition
  - network may heal

Uniform view of distributed objects

- Some objects are remote, some are local
  - Doesn't really matter to user of object
  - Objects might transparently migrate
- Design doesn't have to take object distribution into account
- Failure and performance issues don't belong in the design
- The interface doesn't change if an object is remote

Uniform view

- Not appropriate for
  - wide area networks,
  - consumer electronics,
  - portable devices
- Appropriate for some local area networks
  - but robust distributed applications plan for failure
  - even if all objects local
  - means in interfaces, not just implementation

Memory access

- Can we make the fact that an object is remote transparent?
- Perhaps for objects
  - What about int's ?
  - What about char *'s?
- If you can't directly access fields and create pointers to them
  - then it's not transparent

Partial failure

- Computers fail
- OS's crash
- Networks fail
- PDA's get turned off or taken out of the room
- Often no warning, and each hardware/software component can fail independently of all others
  - and no other component may be able to tell exactly what happened

Queue example

- Want to add x to remote queue q
  - q.enqueue(x)
- Operation could fail
- Want to reliably enqueue x

Queue example

```
while (true) {
    try {
        q.enqueue(x);
        break;
        }
    catch (RemoteException e) {}
}
```

Partial failure

- Object was enqueued, but failure occurred during return message
- Could enqueue x multiple times
- Solution?
  - Need a request tag so that duplicate enqueue requests can be detected
- Real question here is how much does it cost to make the remote call look the same as a local call

Concurrency

- Distributed computations mandate concurrency
  - objects at different locations can't be stopped from executing concurrently
  - even less control than with multi-threading
    - no single point of control (hardware, OS, thread scheduler, etc.)