Slide 1

Administrivia

- Reminder: Homework 5 due Monday.
- Possible info on optional project Monday (or earlier by e-mail).

Slide 2

Example Continued: Generic Master/Worker

- Last time we looked at several versions of a generic master/worker program in Java:
  - Sequential.
  - Multithreaded.
  - Distributed using sockets.
- Today look at distributed version using RMI.

Slide 3

Distributed-Memory Programming in Java Using RMI

- Motivation: For client/server applications, can be annoying to have to design your own protocol.
- Instead, idea is to define "remote objects" that can be treated (at program level) like any other objects — invoke methods.
- Typical use in client/server program:
  - Server creates some remote objects and "registers" them.
  - Clients look up server's remote objects and invoke their methods.
  - Both sides can pass around references to other remote objects.

Slide 4

Java RMI — A Short How-To

- Define a class for remote objects:
  - Define interface that extends `Remote`
  - Define class that implements that interface, instances of which can be remote objects. (So, either have the class extend a remote-object class or use a static method of a remote-object class.) Notice that the class can also include other methods, only available locally.
  - Write code using classes: If using as remote object, reference interface; otherwise can reference class.

Slide 5

Java RMI — A Short How-To, Continued

- Compile as usual.
- Make `.class` files network-accessible. (There are other options, but this is simplest.)
- Start `rmiregistry` — or, with more recent versions, call `LocateRegistry.createRegistry()` within process.
- Run server and clients as usual.

Slide 6

Example: Generic Master/Worker Revisited

- Version using RMI is also straightforward, again except for code to shut down properly. Notice use of `synchronized` in code to ensure thread-safe access to shared variables; experiment suggests that RMI may use multiple threads to process concurrent requests.
- (Caveat: These programs were developed under Java 1.5 so do not necessarily reflect best practice for later releases.)

Slide 7

Distributed-Memory Java and *Implementation Mechanisms*

- Very similar to MPI, really — UE management is outside the scope of the libraries, synchronization is implicit. For sockets, communication is explicit; for RMI, implicit.

Slide 8

Minute Essay

- None really — just sign in, unless questions?
- And best wishes for a pleasant holiday! (To recycle something someone said to me long ago: I hope it's just as productive as you want it to be!)