Administrivia

- Reminder: Project proposals due Thursday.

- Sample solutions for all assignments to be on Web soon.

Slide 1

Example Application — Generic Master/Worker Program

- (Finish looking at code.)

Slide 2

A Little About Distributed-Memory Programming in Java

- Java doesn't exactly provide explicit support for distributed-memory parallel programming.

- However, similar effects can be achieved with multiple Java programs on different machines communicating via socket-to-socket connections and with RMI.
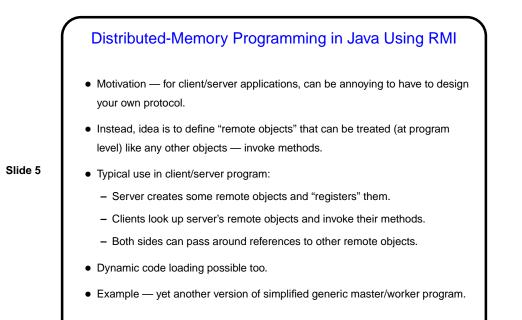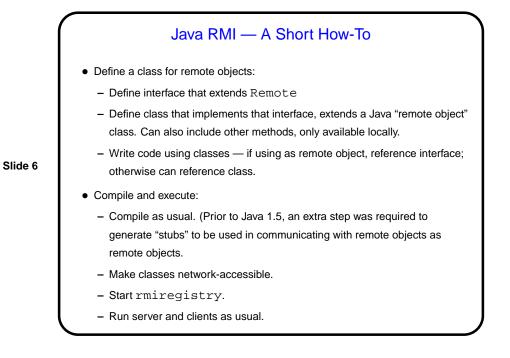
Slide 3

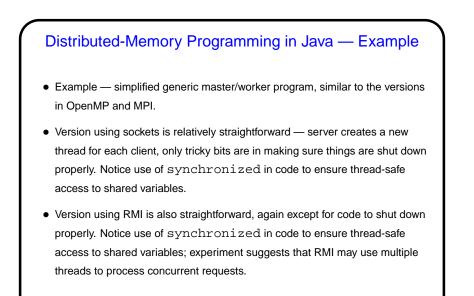## Distributed-Memory Programming in Java Using Sockets

- Client/server model:
  - Server sets up "server socket" specifying port number, then waits to accept connections. Connection generates socket.
  - Client connects to server by giving name/IPA and port number — generates a socket.
  - On each side, get input/output streams for socket. Program must define protocol for the two sides to communicate.

Slide 4

Distributed-Memory Programming in Java Using RMI

Slide 5

- 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.

- Dynamic code loading possible too.

- Example — yet another version of simplified generic master/worker program.

Java RMI — A Short How-To

Slide 6

- Define a class for remote objects:
  - Define interface that extends `Remote`
  - Define class that implements that interface, extends a Java "remote object" class. Can also include other methods, only available locally.
  - Write code using classes — if using as remote object, reference interface; otherwise can reference class.

- Compile and execute:
  - Compile as usual. (Prior to Java 1.5, an extra step was required to generate "stubs" to be used in communicating with remote objects as remote objects.
  - Make classes network-accessible.
  - Start `rmiregistry`.
  - Run server and clients as usual.

Slide 7

Distributed-Memory Programming in Java — Example

- Example — simplified generic master/worker program, similar to the versions in OpenMP and MPI.

- Version using sockets is relatively straightforward — server creates a new thread for each client, only tricky bits are in making sure things are shut down properly. Notice use of `synchronized` in code to ensure thread-safe access to shared variables.

- 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.

Slide 8

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.

Minute Essay

- None — sign in.

Slide 9