CS454

Assignment 3

Design by: Johnson Chan and Andrew Jenkins

System Manual

Marshalling Data

The protocol for sending and receiving data between clients, server, and binder is as follows: first send an integer representing the length of a message to follow, then send an integer representing the type of request, then proceed to send the proceeding messages, which depend on the request/response type.

The length message is used to receive 'x' number of arguments for a function in the form of 'x' messages. From the request type, the binder or RPC library will decipher, via hard-coded switches, the number of messages to follow and their argument types. In other words, data marshalling and unmarshalling is hard-coded and dependent on the type of request.

rpcInit: sends/receives no more data than initial length and type messages.

rpcRegister: sends: hostname (char\*), port (int), function name (char\*), argTypes (int\*)

receives: response code (int)

rpcCall: sends: function name (char\*), argTypes (int\*)

receives: function location response (int)

sends: execute request (int), function name (char\*), argTypes (int\*), arg1 (int), arg2 (int) … argn (int)

receives: response code (int)

rpcExecute: receives: function name (char\*), argTypes (int\*), argTypes (int\*), arg1 (int), arg2 (int) … argn (int)

sends: response code (int)

Binder Database

Function Overloading

Scheduling

Termination Procedure

When the binder receives a TERMINATE request, it proceeds through the list of server sockets and sends a TERMINATE request to every server individually.

If a server is in the middle of execution, and receives a TERMINATE request, the server will wait for all threads to finish execution (wait until numThreads < 0 and do not allow creation of any new threads), then stop listening for requests, and cease execution.