MathServer

A Sample Multithreaded RPC System supporting Mathematical Functions

433-678 - Cluster and Grid Computing

Scott Beca (sbeca) – 267 362

s.beca@ugrad.unimelb.edu.au

Introduction

We were tasked with creating a multithreaded server application which would allow a client to connect and ask for the answer to a mathematical function and the server would respond with the right answer. The server needed to be able to listen for connections on any port supplied as a command-line argument and needed to be able to respond to numerous clients at once using multithreading.

Architecture

We chose to implement our solution in Java as its inbuilt libraries and Object Oriented nature were very conducive to completing this assignment in the most effective and efficient way possible.

We choose to implement the MathServer using the Master Worker Thread Model. The server is started by launching the MathServer class which will then start listening on a user-supplied port for client connections. When a new client tries to connect, the MathServer creates a new Socket, creates a new MathThread thread, supplies it with the newly created Socket and starts it. The MathThread implements all the code for communicating with the client as well as the relevant code for calculating the various mathematical functions. After the connection has been made, the client sends the MathThread a String containing the mathematical function it would like calculated, the MathThread calculates it if it knows the function (if not, it replies with an error) and sends the client the result. After this has been completed, the Socket is closed and the MathThread finishes and is destroyed.

The client is implemented in a single class called MathClient. Through command-line arguments, the user is able to choose which server they wish to connect to and on which port. MathClient contains code to receive input from a user on which mathematical function they would like calculated and with what input value. After the user has supplied the MathClient with this information, the MathClient attempts to form a connection with the MathServer using Sockets. If it is successful, it sends the String detailing the mathematical function and waits for the answer. When it receives the answer, it prints the result to the user's screen and asks if they would like to calculate another. If so, the process is repeated. If not, the MathClient closes.

Protocols

To handle the network communications required for this assignment, we used Sockets. A Socket describes one end of the connection between two programs over a network. A Socket consists of an address that identifies the remote computer and a port for both the local and remote computer. The same port cannot be used by two applications. If one application tries to create a Socket using a port that is already in use, it will fail and throw an exception.

The process of client/server communication is described in the Architecture section so does not need to be explained here again.