**Assignment 2 -** **Multi-threaded Key-Value Store using RPC**

TCSS 558: Applied Distributed Computing, Winter 2015

Team - Leena Singhal, Manali Bhalgat and Amruta Khot

**Executive Summary**

This assignment helps understand the concept of communication between server and client using Remote Procedure calls (RPC). Remote Procedure Call (RPC) is a protocol that one program can use to request a service from a program located in another computer in a network without having to understand the network details. In a client server model, the client is the requesting program and the server is the service providing program. Synchronization is important in RPC as in other local procedure calls when multiple client calls are made to the server. The use of lightweight threads that share the same address space allows multiple RPCs to be performed concurrently.

When RPC code is compiled into a executable program, a stub is included in the compiled code on the client machine which acts as a local procedure call for the client. When the program is run and the procedure is invoked, the stub receives the request and pushes it to the local machine at runtime. The stub has the information of the address to server over the network, which helps client to send the message over the network to the server. In the same way, server runtime has skeleton which acts as a local procedure call for the server. The result will processed by the server and the result is sent back to client.

RPC spans the [Transport layer](http://searchnetworking.techtarget.com/definition/Transport-layer) and the [Application layer](http://searchnetworking.techtarget.com/definition/Application-layer) in the [OSI](http://searchnetworking.techtarget.com/definition/OSI) model of network communication. RPC makes it easier to develop an application that includes multiple programs distributed in a network.

**Technical Impression**

We make use of RMI to implement RPC in Java. Java Remote Method Invocation (RMI) allows you to write distributed objects using Java. All the RMI types are defined in the package java.rmi or one of its subpackages. It provides an interface called “Remote” which denotes that the Interface is to be invoked remotely. The server program implements this interface. Since the server needs to be multithreaded, it extends the thread class. To form the skeleton on the server side, we need to run the “rmic” command against the server class.

Before starting the Server, one should first start the Object Registry, and leave it running in the background. One performs this by using the command:

rmiregistry &

It takes a second or so for the Object Registry to start running and to start listening on its socket.

The Server should then be started; and, like the Object Registry, left running in the background.

The Server will take a few seconds to start running, and to construct and register remote objects. So we need to wait for a few seconds before running any Clients.