**Northeastern University – Silicon Valley**

CS 6650 Scalable Dist Systems

**Homework Set #2** [100 points]

***INSTRUCTIONS: Please provide clear explanations I your own sentences, directly answering the question, demonstrating your understanding of the question and its solution, in depth, with sufficient detail. Submit your solutions [PDF preferred]. Include your full name. Do not email the solutions.***

1. **Java Threads – Single and Multi-threading implementation** [20 points]

This small programming project is preparation toward the next set of programming assignments.

It focuses on implementing a simple Java server using single threading first and multithreading later.

Study the necessary sections from Buyya Ch. 14

http://www.buyya.com/java/Chapter14.pdf

Implement simple client – server communication to demonstrate:

1. Single threading
2. Multi-thredaing

See files and README.

Study **Chapter 4 and 5, Chapters 6 and 9 from** Coulouris Book to answer the below:

1. Answer the following questions using explanation and diagrams as needed. No implementation needed.
2. 4.2 [5 points]

Some design issues:

1. How does the client reach out to server? The client needs to know port and IP address of service. Client can use a binder to map the text to the certain port, or it can use a commonly acknowledged port id, OS can also cache the server address.
2. There might have some efficiency problem. The client might use local identifiers for ports. When server creates a non-public port number, the local name would be useless, and it must be translated to global identifier.
3. The same port can be used for different locations. If client uses a binder, then it needs to reconsult to find the new location.
4. 4.15 [10 points]

For point 1, sender needs to have a sequence number for each message. Receivers will record sequence number and check those numbers on each messages received.

For point 2, as only small amount of data loss, receiver will requests a resend for missing messages, rather than ACK for all messages. For this to work, sender must store all the messages recently sent, so it can retransmit as a unicast datagram.

For point 3, as there is no time limit, there is no ACK for replying. So sender don’t know when to drop the stored message, as it may need to resend. One possible solution is to discard message in the queue after a set amount of time.

1. 5.11 [5 points]

Vote: input. candidate name, voter’s number

Result: output. Candidate name, number of votes.

1. 5.13 [5 points]

For input arguments, CORBA IDL is similar to Java RMI.

For output, java needs to pack all returned value in a class, so it is less convenient than CORBA.

1. 5.22 [10 points]
2. Single-thread: 5 + 4\*0.5 + 3 + 4\*0.5 + 10 + 3 = 25ms

For two requests, 2\*25ms = 50ms

1. Multi-threads:

Client: 5+ 0.5 + 0.5 + 5 + 0.5 + 0.5 = 12ms then waits

Server: 6 + 3 + 0.5 + 0.5 = 10 ms

10 + 0.5 + 0.5 = 11ms

10 + 11 = 21 ms

21 + 11 = 32ms

32 + 3 + 0.5 + 0.5 = 36 ms

So total is 36ms for two calls.

1. Answer the following questions using explanation and diagrams as needed. No implementation needed.
2. 6.8 [5 points]

The client can access the interface provided by mailbox service, so client will register interest. Client needs to know the RemoteEventListener provided, so when the event generator pass the event to mailbox service, then mailbox service will pass that to client through listener.

Also, client must have a way to contact mailbox, so it can turn on and off the notification service. A register() function is needed.

So the client register the mailbox, and get a registration object from mailbox, it stores it somewhere. So even if the client crashes, it can restore the object from the file.

1. 6.14 [5 points]

The consumer needs to only subscribe to one address, so the listener is only point to one publisher.

1. 6.15 [5 points]

It is unsuitable because it is not for heterogeneous working. And for each client, we need a shared memory space, which is not worthy.

It might be useful if clients share the response from server.

1. 9.1 [5 points]

Request reply vs client server.

Client server in SOAP is a more rigid protocol for web service. The rules and standards are important for clients and servers to follow.

Request reply is REST is more flexible. The passing can be XML, JSON, etc. Client server can only use XML.

Two reasons:

SOAP is not bonded to any protocol, so it is easy to work asynchronously.

SOAP is easy to extend, so the asynchronous would be easy to performed.

They are both based on acknowledged rules for message passing. So they both work under HTTP protocol.

1. 9.8 [5 points]

The servlet container provides the servlet to access the HTTP request easily. When the servlet is called, the web server passes the HTTP request to the servlet container. The container will then pass the request to the servlet.

1. See the Coding Tutorial PDF provided, and the below references. Do your own study of RMI Java examples to implement this. Implement a Java RMI Application in which the Client object is sending a list of 10 integers to the Server, and a remote method ['sort()'] o the server returns a sorted version of the same list back to the Client. [20 points]

**References**

Java RMI

<https://www.cs.uic.edu/~troy/fall04/cs441/rmi/calc/index.html>