Homework 2

CS550 (V01)

Amit Nikam (A20470263)
anikam@hawk.iit.edu
Fall 2020

---

1. **(10 points)** Consider a chain of processes P1,P2,...,Pn implementing a multitiered client-server architecture. Process Pi is client of process P(i+1), and Pi will return a reply to P(i-1) only after receiving a reply from Pi+1. What are the main problems with this organization when taking a look at the request-reply performance at process P1?

   Ans: The main problem in this type of layered client-server architecture is the dependence on other Nodes(sometimes on same machine). So in general the performance can be bad for large number of Nodes. When taking a look at the request-reply performance at P1 (with P2) which happens to be the very last node in the chain, the requests coming through the N - 2 nodes also needs to be considered, since any unreachable or poorly performing node in these layers can make the whole process delay and the performance will decline for client at top.

2. **(10 points)** Describe precisely what is meant by a scalable system. Scalability can be achieved by applying different techniques. What are these techniques?

   Ans: Scalability is when a system can grow without any major loss in performance with respect to either its geographical size, number of components or number and size of its administrative domains. Techniques of scalability are caching, distribution and replication. In caching, the distributed cache across machines stores the similar data request information, this makes the resources available quicker and globally. In distribution, the system is distributed as per the application which can make scaling better. And finally in replication, the same system is replicated at different sites, this makes it possible for scalability, specially geographical.

3. **(10 points)** If a client and a server are placed far apart, we may see network latency dominating overall performance. How can we tackle this problem?

   Ans: In this particular case the performance degradation is experienced due to the client waiting for the response from the server. Thus, by changing the synchronous client to a asynchronous type can improve the performance significantly as the client will then move on to the next task after while the server responds. Alternatively, by properly modularizing the client it becomes possible to improve performance as modules might be able to do their tasks normally while other waits for a response from the server.

4. **(10 points)** Imagine a Web server that maintains a table in which client IP addresses are mapped to the most recently accessed Web pages. When a client connects to the server, the server looks up the client in its table, and if found, returns the registered page. Is this server stateful or stateless?

   Ans: This happens to be a stateless server. The process the server performs happens to be more of a convenience feature and not a crucial step. Even if the above mentioned process was to be skipped or removed, it wouldn't significantly tamper with the goal. If a process like authentication were to be implemented

which checked user permissions for the particular page, then it would be a stateful server.

5. **(10 points)** Consider a BitTorrent system in which each node has an outgoing link with a bandwidth capacity *Bout* and an incoming link with bandwidth capacity *Bin*. Some of these nodes (called seeds) voluntarily offer files to be downloaded by others. What is the maximum download capacity of a BitTorrent client if we assume that it can contact at most one seed at a time?

   Ans: Since BitTorrent follows a tit-for-tat policy the total *Bin* data chucks a client will receive from other nodes in the network is proportional with its *Bout.* So part of its total download is *Bout.* Since the S number of seeders are nodes in the system, they have an output of *Bout.* So now to understand the data that the client will receive from one seeder, we need to take in to consideration that the seeder would be sending data to other nodes too. So the client will only receive *Bout* / N data from a single seed where N is the total number of nodes in the network.

   So ultimately even though total max capacity to download of one client is [(S x *Bout*) / N + *Bout*], client that connects to only one seeder has a download rate of [(*Bout* / N) + *Bout*].

6. **(10 points)** In many layered protocols, each layer has its own header. Surely it would be more efficient to have a single header at the front of each message with all the control in it than all these separate headers. Why is this not done?

   Ans: One of the reasons why one big header is not implemented is because the data is in different forms at different network layers. So to make a single big header, the header would need to be decoded, modified and then again encoded to be sent ahead. This adds unnecessary cost to the system.

   Secondly this type of implementation would remove the transparency between a layer from other layers, which goes against the very transparency goal of Distributed Systems. The desired implementation is one in which one layer is invisible from another. In the current implementation the header is padded with the body and passed to the lower layers as one offload, this is the desired implementation and so it should not be changed.

7. **(10 points)** Consider a procedure incr with two integer parameters. The procedure adds one to each parameter. Now suppose that it is called with the same variable twice, for example, as incr(i, i). If i is initially 0, what value will it have afterward if call-by-reference is used? How about if copy/restore is used?

   Ans: When passing the variable by reference, the location to the variable in memory is passed to the function as a pointer and so the same variable will be incremented twice and answer value will be 2.

   But while using copy/restore, a copy of the variable is sent to the function. Since the same variable is passed twice, two copies of the same variable will be each incremented in the function and the latest copy will overwrite any previous values so finally the value will be 1.

8. **(10 points)** Describe how connectionless communication between a client and a server proceeds when using sockets.

   Ans: Socket happens to be a logical endpoint through which the client-server communicate. On the server this endpoint is binding to a port and is actively blocking read for any client communications. The client socket does a blocking write to the server socket using its address, i.e. location on network and port number. The connection need not be closed in this case.

9. **(10 points)** Suppose that you could make use of only transient asynchronous communication primitives, including only an asynchronous receive primitive. How would you implement primitives for transient *synchronous* communications?

   Ans: To implement a synchronous communication using transient asynchronous communication I would implement a read task after sending communication to the server that would poll until a communication is received. Alternatively if the low levels in the communication support storing of responses, I would block the caller until the lower layer serves the response after which the caller would continue with further tasks.

10. **(10 points)** Assume a client calls an asynchronous RPC to a server, and subsequently waits until the server returns a result using another asynchronous RPC. Is this approach the same as letting the client execute a normal RPC? What if we replace the asynchronous RPCs with synchronous RPCs?

    Ans: This would not exactly be the same. In this implementation the client would receive an acknowledgement from the server because of the asynchronous RPC that a request has been accepted and but the result is only guaranteed if the communication is reliable which is generally not the case. But it would be similar if the result is received after the acknowledge is received. So to answer, it would depend on the network reliability and is not same in most cases.