CS 677 Spring 2019: Homework 1

Solutions

March 4, 2019

1. **What is replication transparency? Give an example where it is beneficial for the system to be replication transparent to users.**

   Replication transparency means the fact that resources may be replicated is hidden from the users. For example, in a Content Distribution Network (CDN), the contents of a website are replicated in geographically distributed servers. When users visit a website hosted on a CDN, the CDN automatically redirects them to the contents hosted on the nearest server and the users are unaware of the other replicas.

2. **What is the difference between the processor pool model and the traditional client server model that were discussed in Lecture 1?**

   In the traditional client server model, the clients are local workstations that have certain processing power. User tasks are usually preformed partly on local clients and partly on remote servers.

   In the processor pool model, the clients are thin clients that have little processing power (e.g. diskless terminals). In this model, clients have to rely on the server for everything because they have nearly no processing power.

3. **Why is the event-based architecture also referred to as the publish-subscribe architecture?**

   In the event-based architecture, different system components do not communicate directly; instead, they communicate by sending/receiving events over a event bus. There are two types of components: 1) publisher who sends events to the event bus when certain events happen, and 2) subscriber who subscribes to certain events and gets notified when those events are ready on the events bus. Therefore, the event-based architecture is also referred to as the publish-subscribe model.

4. **For multi-tiered architecture, give an example of when you would put the user interface layer at the client, the data layer at the server and split the application layer between the client and the server.**

   A scenario where this kind of division can be employed are client-based online games. The user interface is completely rendered on the client side to reduce the response time on user

interaction. Some part of the application logic are executed on client side for better performance while other part are performed on server side to prevent cheating. The user data are stored completely on the server side so that users can access their data seamlessly from different machines.

5. **In Chord, explain what happens when a node joins the ring at ID i?**

   If there is a previous node before i (the ID for that node k is lower than i), the new joined node will take on the answering responsibility from the previous node to itself (k + 1, ... , i). If there is no previous node before i the new joined node will answer 0, ... , i.

6. **Not every node in a peer-to-peer network should become super- peer. What are reasonable requirements that a super-peer should meet?**

   Since super-peers take up additional responsibility in the network on top of the work that a normal node carries out, super-peers should have enough resources to accommodate this new responsibility. Examples are processing power, enough memory, ability to handle more network requests etc.

7. **Give one advantage of an edge server system.**

   Reduced latency and bandwidth for clients geographically close to the edge server system. This is because communication does not have to go all the way to a central server which could potentially be in a different geographical location further away from the client.

8. **Explain why multi-threaded processes can achieve true parallelism, and not just concurrency, in a multiprocessor system?**

   In multi-threaded processes, each thread has its own instruction stream independent from other threads. Therefore different threads can be run on multiple processes in parallel. Also, when one thread makes blocking system calls, although that thread goes into block state, other threads can continue to run so the whole process will not block. If the number of processes is more than the number of cores, if a thread on core goes into block state, other threads that are waiting can still use that core. Therefore multi-threaded process can achieve true parallelism rather than just concurrency.

9. **What are some advantages of kernel level threads over user- level threads? Do lightweight processes share the advantages of kernel level threads?**

   Kernel level threads have the following advantages over user level threads:

   (a) Scheduler can make better scheduling decisions because it has full knowledge of all the threads, while when user level threads the scheduler has no information about the threads within a process.

   (b) With kernel level threads a process can achieve true parallelism on multiprocessor machines because different threads within a process can run on multiple cores simultaneously. With user level threads only one thread within a process can run at any given time.

(c) With kernel level threads developers don?t need to avoid blocking system calls because if one thread goes into blocking state other threads can continue to run. With user level threads blocking system calls have to be avoided because if one thread block the whole process will also block.

Lightweight processes also share the advantages of kernel level threads because it creates a many-to-many mapping between the threads within a process and the scheduler.

10. **Statically associating only a single thread with a lightweight process is not such a good idea. Why not?**

    Because such an association effectively reduces to a pure kernel level thread implementation, which loses all the advantages of user level threads that LWPs bring. In fact, you incur a slight overhead of matching LWPs to threads which you would not have in a pure kernel level thread implementation.