CS 6210 Spring 2014 Final Solution

Name:_____Kishore_____GT Number:

Monday April 28, 2014 (8:00 to 10:50 AM)

Note:

1. **Write your name and GT number on each page.**
2. The test is **CLOSED BOOK** and **NOTES.**
3. Please provide the answers in the space provided. You can use scratch paper (provided by us) to figure things out (if needed) but you get credit **only** for what you put down in the space provided for each answer.
4. For conceptual questions, **concise bullets** (**not wordy sentences**) are preferred.
5. Where appropriate use figures to convey your points (a figure is worth a thousand words!)
6. **Illegible answers are wrong answers.**
7. **DON'T GET STUCK ON ANY SINGLE QUESTION…FIRST PASS: ANSWER QUESTIONS YOU CAN WITHOUT MUCH THINK TIME; SECOND PASS: DO THE REST.**

Good luck!

Question number		Points earned	Running total
1 (0 min)	(Max: 1 pts)		
2 (10 min)	(Max: 10 pts)		
3 (15 min)	(Max: 12 pts)		
4 (15 min)	(Max: 14 pts)		
5 (20 min)	(Max: 16 pts)		
6 (10 min)	(Max: 8 pts)		
7 (20 min)	(Max: 19 pts)		
8 (10 min)	(Max: 10 pts)		
9 (10 min)	(Max: 10 pts)		
Total (110 min)	(Max: 100 pts)		

1. (0 min, 1 point)(circle the right choice)
Who won the Turing Award for 2014?
   a. Dick Lipton
   b. Richard Karp
   c. Juris Hartmanis
   d. John von Neumann
   e. Raj Reddy
   f. Leslie Lamport
   g. Jimmy Falon
   h. None of the above

CS 6210 Spring 2014 Final Solution

Name:_____Kishore_____GT Number:

2. (15 min, 10 points) (**Distributed Objects – Java, Spring Kernel**)

a) (Choose **ONE true statement from the following)** Interface Definition Language (IDL) serves the following purpose

I. It allows expressing a subsystem interface in a language independent manner
II. It is a specific way of writing the header file containing the method prototypes for object-oriented languages such as Java
III. It is purely used for documentation purposes and does not play any part in the software development process

b) (Choose **ONE true statement from the following)** Subcontract in Spring Kernel is a mechanism that serves the following purpose

I. It ensures that there is exactly one implementation of a given service
II. It makes client/server interactions location transparent
III. It is the way marshalling and unmarshalling are done in Spring kernel when the client and server are on the same machine

c) (Choose **ONE true statement from the following)** Spring kernel allows extension of the operating system (i.e., addition of new subsystems and services) via the following method

I. By using C++ as the language to implement the operating system
II. By use of subcontracts as an underlying mechanism for object method invocations
III. By using open language independent interfaces to enable third party software development

d) Choose the **ONE false statement that does not apply** for remote objects in Java

I. References to objects can be passed as args/result
II. Java built-in operators are available for use with them
III. Parameter passing semantics are the same for remote objects as for local objects

e) Choose the **ONE false statement that does not apply** for object-oriented programming

I. They provide strong interfaces allowing 3rd party software development
II. They promote reuse of software components
III. They always lend themselves to superior implementation (i.e., performance) compared to implementation of a subsystem in a procedural or imperative style

CS 6210 Spring 2014 Final Solution

Name:_____Kishore_____GT Number:

3. (15 min, 12 points) (**Global Memory System**)
(a) At the beginning of each Epoch, what information is sent to the initiator node for an epoch by the rest of the nodes? (**bulleted list please**)

- Age information for all the local and global pages: bigger age means older

(+3)

(b) What values are calculated by the initiator node and returned to each node? (**bulleted list please**)

- Compute MinAge: this is the minimum cutoff age above which initiator guestimates M oldest pages will get replaced in the upcoming epoch
- Compute Wi for each node i: Weight Wi is the fraction of the M pages that is expected to be replace at node I in the upcoming epoch
- Pick initiator node for the next epoch: This is node i with MAX(Wi)
- Each node receives {MinAge, Wi for all nodes i)

(+0.5 for each of the first two bullets; +1 for each of the third and fourth bullets)

(c) How are the values returned by the initiator used by each node during the epoch?

- Let y be the page chosen for eviction at node i;
- If Age(y) > MinAge => discard the page y **(+1)**
- If Age(y) < MinAge => send y to peer node j **(+1)**
  - Choose node j using the weight information => probability of choice of node j proportional to Wj **(+1)**

(d) In implementing GMS, what is the technical difficulty of obtaining age information for the pages at a node?

- Hardware does address translation on every memory access. **(+1.5)**
- Therefore, the memory access pattern for the pages backing the virtual memory subsystem is invisible to GMS software. **(+1.5)**
- Thus obtaining the age information for such pages (referred to as anonymous pages) is difficult.
- GMS periodically runs a daemon as part of the VM subsystem to dump the TLB information to keep track of "approximate" age information for such pages

(The last two bullets are just elaboration of the second bullet)

CS 6210 Spring 2014 Final Solution

Name:_____Kishore_____GT Number:

4. (15 min, 14 points) (**Distributed Shared Memory and Treadmarks)**
(a) Consider a distributed shared memory machine. On each processor, **read/write to memory is atomic**. But there is no guarantee on the interleaving of read/writes across processors.

A parallel program is running on processors P1 and P2.

Intent of the programmer:

P1	P2
Modify Struct(A)	Wait for modification Use Struct(A)

The pseudo code that the programmer has written to achieve this intent:

flag = 0; initialization

P1	P2

mod(A);
flag = 1;
	while (flag == 0); //spinwait
	use (A);
	flag = 0;

(i) (3 points) Explain why the above code may not achieve the programmer's intent.

The above code is not guaranteed to work. **(+1 if they get this right)**

Reason: read/writes from P1 happen in textual order. But since there is no guarantee on the order in which writes from a given processor become visible to the other, P1's write to flag may become visible to P2 before the modifications to A by P1 have become visible to P2. This will result in violating the intent of the programmer.

(+2 if they get the reason right; partial credit commensurate with the reasons)

(ii) (2 points) What is the guarantee needed from the memory system to make this code achieve the programmer's intent?

Sequential consistency:

- Read/writes on each processor respect the program order
- Reads/writes from the different processors interleaved consistent with the program order on individual processors

(+1 for each of the above two bullets)

CS 6210 Spring 2014 Final Solution

Name:_____Kishore_____GT Number:

(b)
Consider the following execution happening in Treadmarks. Recall that Treadmarks implements Lazy Release Consistency.
Assume initially that both processors P1 and P2 have copies of the pages X, Y, and Z.

```
P1:
      Lock (L)
            Step 1:  Updates to page X;
            Step 2:  Updates to page Y;
            Step 3:  Updates to page Z;
      Unlock (L)

Subsequently, processor P2 executes the following code:
P2:
      Lock (L)
            Step 4: Read page X;
      Unlock (L)
```

(i) (3 points) Explain what happens on P1 at Step 1

- Create twin for X; call it X' **(+1.5)**
- Write updates to X (original not the twin) **(+1.5)**

(ii) (3 points) Explain what happens on P2 at Step 4

- P2 at the point of lock acquisition invalidates pages X, Y, and Z
- At step 4, P2 goes to P1 and obtains diff(X): created by P1 at the point of unlock(L)
- P2 applies diff(X) to its copy of X and makes X valid and completes read operation.

(+1 for each bullet)

Name:_____Kishore_____GT Number:

(c)(3 points) Explain false sharing in general with a code example.

```
Assume P1's cache:                    Assume P2's cache:
Cacheline contains variables v1, v2   Cacheline contains variables v1, v2

P1's code:                            P2's code:

lock(L1)                              lock(L2)
     update(v1) repeatedely                update(v2) repeatedly
unlock(L2)                            unlock(L2)
```

- Code on P1 and P2 can be executed concurrently
- But cacheline on P1 and P2 contains v1, and v2
- This cacheline will ping pong between P1 and P2
- This is false sharing

(if answer is unclear, here are partial credit guidelines:
+1 for cacheline containing multiple variables
+1 for showing cacheline ping ponging between P1 and P2
+1 mentioning concurrent execution of code on P1 and P2)

5. (20 mins, 16 points) (**Distributed File System - xFS**)
In a centralized file system, the server performs the functions of managing the data blocks, metadata for the files, server-side file cache, and consistency of datablocks of files cached by multiple clients. **The following questions are with respect to how these functions are carried out in xFS.**

Answer True/False and explain why xFS chose to do it (or not do it) that way (If you only say True or False you will not receive full credit.)

(a) Meta data for files are located in the same node as the data.

False. To load-balance metadata management, xFS decouples the location of metadata for a file from the location of the content.

(+1 for false; +3 for reason; partial credit as appropriate)

Name:_____Kishore_____GT Number:

(b) A file is contained entirely on a single disk in the entire system.

False. It uses software RAID to stripe a file on a selected number of disks as determined by the stripe group.

(+1 for false; +3 for reason; partial credit as appropriate)

(c) Small file write problem is solved in xFS.

True. It uses a log structured file system to overcome the small write problem.

(+1 for true; +3 for reason; partial credit as appropriate)

(d) The in-memory cache for a file resides at the same node as the disk copy.

False. xFS uses cooperative caching meaning that the file may be encached at a client node different from the server that hosts the file in its disk.

(+1 for false; +3 for reason; partial credit as appropriate)

CS 6210 Spring 2014 Final Solution

Name:_____Kishore_____GT Number:

6. (10 min, 8 points) (**Failures, LRVM, Quicksilver**)

a) (choose **ONE true statement from the following)** There is **no undo log** in Satyanarayanan's LRVM because

I. LRVM copies the old values for the specified range of addresses in a set-range call into virtual memory
II. Entire virtual memory of a process is made persistent by using LRVM library
III. The transactions as defined in LRVM never abort
IV. LRVM assumes that there is battery backup for the physical memory

b) (choose **ONE false statement from the following)** Satyanarayanan's LRVM is light weight because

I. It does not have to keep undo logs on the disk
II. It provides lazy semantics for reducing the number I/O activities to keep the virtual memory persistent
III. It does not implement the full transactional semantics in the database sense
IV. It is implemented in the kernel

c) (choose **ONE false statement from the following)** The sources of problems in computer systems that lead to unreliability include

I. Power failure resulting in loss of volatile state in physical memory
II. Software crash due to bugs in the code
III. Lack of semantics that allows inferring the state of the system prior to the crash
IV. Not implementing transactional semantics at the operating system level

d) (choose **ONE true statement from the following)** A server (such as a file system) with a recoverable state in Quicksilver will use

I. Two-phase commit protocol
II. One-phase standard protocol
III. One-phase immediate protocol
IV. One-phase delayed protocol

7. (20 min, 19 points) (**Internet Scale Computing**)

(a) (5 points)

DQ Principle: If D is data per query and Q is the number of queries per second, then the product D*Q tends towards a constant for a fully utilized system. How can the DQ principle be used to come up with policies for graceful degradation of the system under excess load?

Q relates to yield => number of successful queries processed
D relates to harvest => fraction of complete data server per query
Graceful degradation policy:

- Admission control knob to reduce Q but keep D the same **(+2)**
- Fidelity control knob to reduce D to allow increasing Q **(+2)**
- These two knobs can be individually adjusted under the constraint that DQ is a constant. **(+1)**

CS 6210 Spring 2014 Final Solution

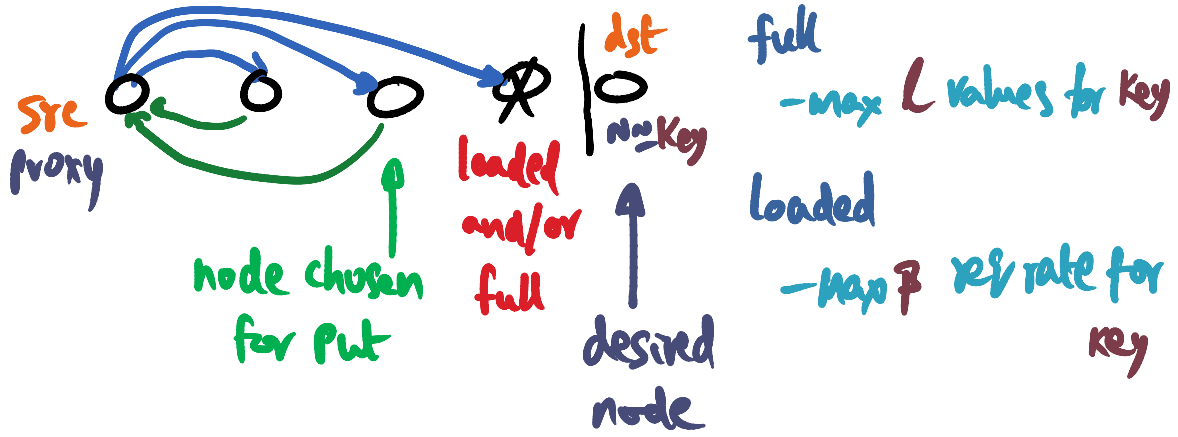
Name:_____Kishore_____GT Number:

(b) (Coral CDN)
If **k** is the key and **n** is the node id, a traditional DHT would try to store **k** at a location **n**, where **n** is equal to **k**.

(i) (5 points) Explain why Sloppy DHT of Coral does not do that.

- Meta data server can become overloaded if too many keys (k) map to the same node (n) **(+2)**
- Nodes en route to node n would also experience too much traffic due to the put/get operations wanting to get to the overloaded meta data server **(+2)**
- Coral avoids both these problems by using k=n as hint not absolute in get/put operations. **(+1)**

(ii) (9 points) Explain how the "put" algorithm of Coral works with figures.



(+2 for figure; partial credit as appropriate for the figure)

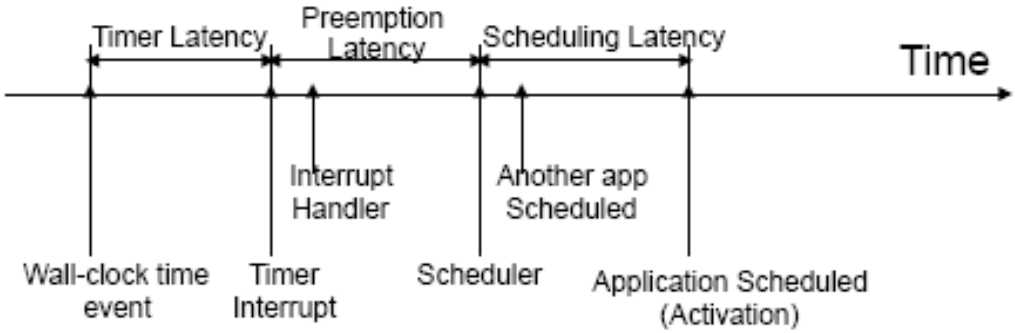
Every node attribute:
- full: max l values for key k **(+1)**
- loaded: max beta requests rate for key k **(+1)**

Two-phase put:
- Forward phase
  - Putter sends RPCs to nodes en route to node n (~=k) using the key-based routing or Coral **(+2)**
  - Put stops at "full" and "loaded" node **(+1)**
- Reverse phase
  - Some nodes that were NOT full or loaded, may have become loaded and/or full **(+1)**
  - Pick a node that is closest to the key as the destination for the put => this avoids the meta server overload and tree saturation **(+1)**

CS 6210 Spring 2014 Final Solution

Name:_____Kishore_____GT Number:

8. (10 min, 10 points) (**RT Linux**)
(a) (6 points) Define the following terms (use figures to help get your point across)
(i) "timer latency"
(ii) "preemption latency"
(iii) "scheduling latency"



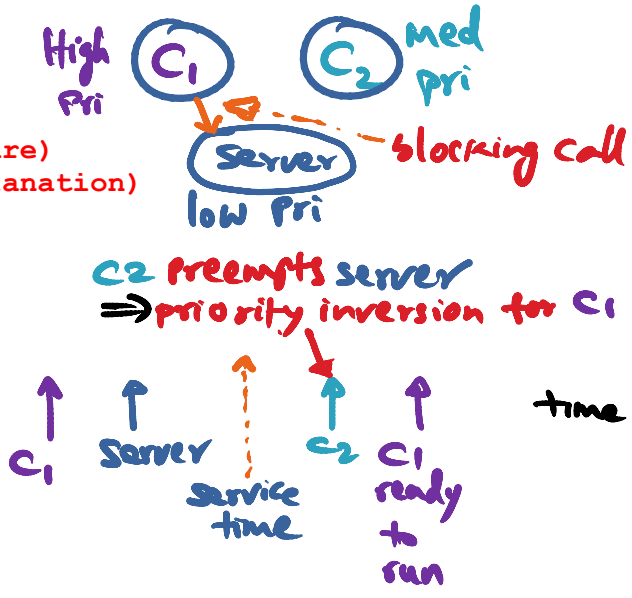
- Timer latency: distance between event occurrence and timer interrupt due to the granularity of the timing mechanism
- Preemption latency: distance between timer interrupt and opportunity to schedule the event due to activity on the CPU being non pre-emptible (e.g., kernel in the middle of an interrupt handler)
- Scheduling latency: distance between when the event becomes schedulable and when it actually gets scheduled due to higher priority applications already in the CPU scheduling queue

+2 for each highlighted phrase or equivalent

(b)(4 points) Explain priority inversion with illustrative figures.

Consider a high pri task C1 calling a low pri server task. It is a blocking call, so server takes over and does the bidding. After the serivice time c1 ready to run again. C2 med prio becomes runnable. C2 preempts server => priority inv wrt C1

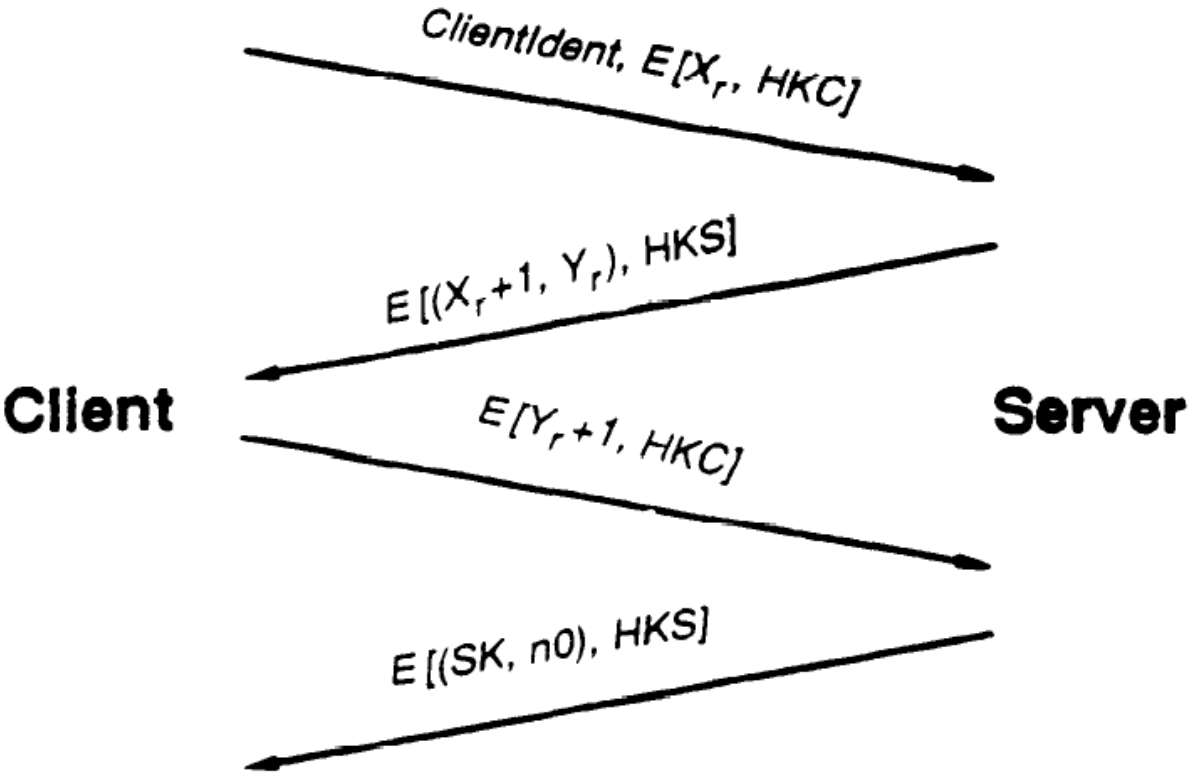
(+2 for figure)
(+2 for explanation)



CS 6210 Spring 2014 Final Solution

Name:_____Kishore_____GT Number:

9. (10 min, 10 points) (**Security and Andrew File System**)



With respect to the above client-server interaction in the Andrew File system, answer the following questions:

(a) (2 points) X_r is a random number generated by the client. What purpose does this serve?

When the client receives the first message from the server E[(Xr+1, Yr), HKS], it will check the field Xr+1. Note that Xr was generated by the client and only if the server was able to decipher the message, Xr+1 field will have the right value. A replay attack will not have the right value for this field.

(+2 if they say this; partial credit as appropriate)

(b) (2 points) Y_r is a random number generated by the server. What purpose does this serve?

When the server receives the second message from the client E[Yr+1,HKC],it will check the field Yr+1. Note that Yr was generated by the server and only if the client was able to decipher the message, Yr+1 field will have the right value. A replay attack will not have the right value for this field.

(+2 if they say this; partial credit as appropriate)

CS 6210 Spring 2014 Final Solution

Name:_____Kishore_____GT Number:

(c) (2 points) What is "ClientIdent" during an RPC session establishment?

ClientIdent will be the user's login "username" and the key for encryption will be the "password" associated with this username (this is known to the user and the system every authorized user of the system).

(+2 if they say this; partial credit as appropriate)

(d) (2 points) Is "ClientIdent" sent in plaintext or encrypted? Explain why.

- "ClientIdent" is sent as plaintext. **(+1)**
- Since the system uses private key encryption, the server needs to know the identity of the requestor to choose the right key for decryption. **(+1)**

(e) (2 points) What is SK? What is its purpose?

- SK is new "session key" generated by the server for the new RPC session that the client has requested. The new RPC session will use SK as the encryption key.
  (+1)
- Generating a new SK for each RPC session ensures that the handshake key (HKC) contained in the cleartoken is not over-exposed on the wire during the login session.
  (+1)