Assignment 7 – Solution

Problem 1.

a) Call ReadTop(r) to read the request at the top of the queue and then run a transaction to process r. After processing r, Dequeue it. If the system stays up long enough, the top request will be processed at least once. If there's a failure after the ReadTop and before the Dequeue, then after recovery the request will be processed a second time giving at least once semantics.

b) Call Dequeue(r) and then run a transaction to process r. If there's a failure that causes the latter transaction to abort, then r may not be processed. But since it's dequeued only once, it will be processed at most once.

c) Call ReadTop(r). If r has already executed (we can find out because r is testable), then dequeue(r), else run a transaction to process r. After the latter transaction commits, dequeue(r).

Problem 2.

a) True. If the containing transaction commits, ProcessRequest() is executed and its reply is durable. Recovery code ensures that it is not executed again. If the containing transaction aborts, ProcessRequest() is undone. Each request is processed exactly once.

b) False. If the server fails during recovery after it dequeues x and before it enqueues reply, it will never enqueue a reply.

c) True. Adding a reply is coupled with removing the corresponding request in an atomic operation. Exactly one reply is enqueued when the request dequeued.

Problem 3.

a) False. If the transaction containing ProcessRequest() aborts after the call (for example we run out of disk space or hit a software bug), and then the server crashes, then ProcessRequest() will be called again after recovery.

b) Same as problem 2.b.

c) True. Adding a reply is coupled with removing the corresponding request in an atomic operation. Failures may cause a request to be executed several times in

ProcessRequest(r) generating multiple replies. But exactly one reply is enqueued when the request id dequeued.