CSEP590SG - Assignment #3

This assignment is due in class (before class begins) on Tuesday, Feb 17, 2004. Please do not turn in a late assignment.

Before starting in on this assignment, I strongly recommend that you read through Butler Lampson's alternate explanation of Paxos. A link to this paper is:

http://www.cs.washington.edu/education/courses/csep590sg/CurrentQtr/assignments/paxos_lampson.pdf

I've also added this link to the class webpages.

The explanation of the algorithm is in section 6.4 of the paper. You should focus on sections 1, 2, 3, and 6. Sections 4 and 5 are useful to understand some of the terminology that Lampson uses in 6. But, 6.4 is the key section.

One word of warning: there is a bug in the paper. In section 6.4, when Lampson is talking about the examples, he says:

"If the leader hears only from a and b or a and c, it knows that round 3 is dead but does not know that round 2 is dead, and hence must choose 8. If it hears only from b and c, it does not know that round 3 is dead and hence must choose 9."

The corrected version of this is:

"If the leader hears only from a and b, it knows that round 3 is dead but does not know that round 2 is dead, and hence must choose 8. If it hears only from a and c or b and c, it does not know that round 3 is dead and hence must choose 9."

A similar bug exists in the the "Leader's choices for round 4" quadrant of the example.

So…with this, your assignment is….

Building Paxos

Your goal is to implement the "basic" Paxos single-decree synod protocol, i.e., section 2.3 of the Lamport paper. There are three main components you must build:

1. Implement an object that when instantiated, acts as a parliament member during a single decree. The object accepts messages from leaders[s], and processes them according to the single-decree synod protocol.

2. Implement an object that when instantiated, acts as a leader and attempts to propose and pass a decree [i.e., leads one round in the protocol]. You should pass the communications address of the parliament members in the constructor of this object.

3. Implement an object that attempts to determine whether or not the decree has been passed. [hint: you should reuse a lot of the code you implemented in object 2.]

You only need to worry about a single decree- don't implement the multidecree procotol.

Feel free to either build your system as either i. or ii. below:

i. A collection of objects within a single process. The addresses you pass to the constructors in 2 and 3 are pointers to the objects of type 1. Messages are just method calls and return values.

ii. A collection of processes. The addresses in 2. and 3. are IP addresses and port numbers of objects of type 1. Messages are now TCP or UDP messages.

Test your implementation, and submit:

1. Your source code.
2. A (brief) design document.
3. A description of how you tested your implementation, including a justification of the test cases you thought were important.