Distributed Mutual Exclusion
A project for cs2510, fall 2011

1 Introduction

The objective of this project is to put in practice the concepts seen in the distributed operating system course. This project uses some of the infrastructure developed in project 1.

In this project you will implement TWO distributed mutual exclusion algorithms, namely Ricart & Agrawala (tokenless) and a Tree-Based (like Raymond’s token-based). This system lets users communicate among themselves through updating a shared resource (e.g., drawings or text) across domains, such as different machines, different processes, etc). The different users will communicate through the communication facilities from project 1. There are two types of operations:

1. When reading, each user will have a consistent view of the state of the resource (i.e., all users see the same contents at each read). For performance and fault tolerance purposes, you may want to replicate the data (locally?). Note that even though there may be delays in making the resource consistent across platforms, each user $u_1$ that reads often enough will see the same sequence of value that another user $u_i, 1 < i < n$ sees.

2. When writing, ensure that your project provides fault tolerance and good performance overall.
Issues

Many threads should be dispatched in the same AND in different machines via some script; you will decide whether threads declare their shared resources at dispatch time or at run time (dynamically). For some algorithms, there might be a need for a name server type of entity to register the users for a particular resource, and make the member connections known to other members. Analogously, when a user requests withdrawal from the group, the other members should be notified. You may use the structures you developed for project 1, either centralized or distributed.

If a node fails (assume all links are reliable), you will need a certain type of protocol to handle the token and the tokenless protocols. There could be failures such as: a node holding the token dies, a node in the critical section dies, a node waiting for the CS dies, a node in the tree dies, the root of the tree dies, etc. YOU should use your creativity to decide which faults you will deal with, why, and how to recover from the faults.

2 Requirements

2.1 Project Design, deadline Nov 4, 15% of the grade

As part of the design document you are required to describe the model that you intend to implement. This 2-3 page report should contain the design and implementation issues, and the justification of each decision you have made. The design document should include your initial idea of the module decomposition of the system, the interface between each pair of components, and the justification of design choice. For example, you should discuss the fault model and its advantages and disadvantages. Another example is to discuss how to implement the protocol for a member to join/leave the group and the messages exchanged to enable this. Another example is your conjec-
ture of what is each part of the project good for: is it good for high/low load (of reads, writes, both, neither), what type of faults is it good for (transient, permanent, etc), etc.

Careful consideration should be given to the design, since the design choices will directly affect implementation. Each group may make an appointment with the TA to discuss the choices and explain the design. All members of the group should be able to discuss all the aspects of the project. Appointments for demos will also be scheduled later.

2.2 Project Implementation, deadline Nov 16, 85% of the grade

In addition to the design document above, the project will be divided into several parts:

1. (25%) the implementation of the mutual exclusion algorithms
2. (25%) the implementation of the fault detection and handling
3. (20%) the report with analysis of the performance of the implementation for different workloads and different number and types of faults
4. (15%) the demo