

Objectives

1. Implement the Ring Coordinator Election Algorithm (i.e. Chapter 6)
2. More experience with socket programming for inter-process communication

Due: April 16, 11:59 P.M.

Project Specification

1. You are to implement the Ring Coordinator Election Algorithm over sockets. More specifically, your program needs to simulate the following scenarios:
 - a. Initially, all processes start up and communicate in a ring organization. They should display the messages being passed so that we can observe the process. They should pass a “token” around the ring. You should build delays into the process so that you can clearly demonstrate the working of the algorithm. Processes are numbered as in the textbook (i.e. Chapter 6). Each process should receive the token at some small, predetermined interval. Make this interval different for each process so that they do not initially all start an election at the same time.
 - b. When a process does not see the token in the expected interval it will initiate an election. After the system elects a coordinator you will need to demonstrate the following scenario:
 - i. Manually stop communication by the current coordinator.
 - ii. An election will be initiated by a process that does not see the token when expected.
 - iii. The election should proceed according to the ring algorithm given in Chapter 6.
 - c. The crashed process should be brought back up and it should initiate an election.
 - d. Set the timers so that two elections (by processes 2 & 5) run at the same time and show that they still work.
2. You need to clearly show the communication between processes.

Other Specifications

1. This is an individual project.
2. Your program can be run on any platform that you can provide access to for the TA.
3. Either a console or GUI program is acceptable.
4. Your source code should contain your name and login ID.

Write-up:

Your write-up should include instructions on how to compile and run your program. Ideally it should be complete enough that the TA can test your program without your being there. Your write-up should include any known bugs and limitations in your programs. If you made any assumptions such as limits on the size of a user name you should document what you decided and why. This write-up should be in text format and should be submitted along with your code. If you use code found on the Internet or in a book then the write-up must give a reference to the source so we can tell that you did not copy the code from another student.

Submission Guidelines:

Submit your code by the due date via Blackboard. You should zip your source files and other necessary items like project definitions, classes, special controls, DLLs, etc. and your write-up into a single file and include it with your Blackboard submission. The name of this file should be your last name and the last four digits of a student ID number in the form name_number.zip. Be sure that you include everything necessary to unzip this file on another machine and compile and run it. This might include forms, modules, classes, configuration files, etc.

Make sure your name and your student ID are listed in your write-up, and in comments in your source code. You may resubmit the project at any time. Late submissions will be accepted at a penalty of 10 points per day. This penalty will apply regardless of whether you have other excuses. In other words, it may pay you to submit this project early. If the TA cannot run your program based on the information in your write-up then he will email you to schedule a demo. The TA may optionally decide to require all students to demonstrate their labs. In that case we will announce it to the class.

If your program is not working by the deadline, send it anyway and review it with the TA for partial credit. Do not take a zero or excessive late penalties just because it isn't working yet. We will make an effort to grade you on the work you have done.

DO NOT POST YOUR CODE ON PUBLICLY ACCESSIBLE WEBSITES UNTIL AFTER THE DEADLINE

Grading

- 20 – Using sockets for inter-process communication
- 10 – All processes start up correctly
- 10 – Shut down the coordinator communication
- 10 – Election initiated when the coordinator stops communicating
- 10 – The correct process wins the election
- 10 – Correct messages communicated and shown on each process
- 20 – Multiple elections work correctly
- 05 – GTA discretion to fit the lab intention
- 05 – Required documentation including comments

To receive full credit for comments in the code you should have brief headers at the start of every module/ subroutine/ function explaining the inputs, outputs and function of the module. You should have a comment on every data item explaining what it is about. (Almost) every line of code should have a comment explaining what is going on. A comment such as `/* Add 1 to counter */` will not be sufficient. The comment should explain what is being counted.

Extra credit: (5 points) - Develop this project using a GUI

Deductions for failing to follow directions:

- 2 Including absolute/ binary/ executable module in submission
- 2 Submitting write-up in other than plain ASCII text format
- 2 Submitted file has a name other than as described in the Submission Guidelines
- 10 Submitting Java runtime system with code
- 2 Source program does not contain your name and login ID in a comment

Important Note:

You may discuss the problem definition and tools with other students. You may discuss the lab requirements. You may discuss or share project designs. All coding work must be your own. You may use any book, WWW reference or other people's programs (but not those of other students in the class) as a reference as long as you cite that reference in the comments. If you use parts of other programs or code from web sites or books YOU MUST CITE THOSE REFERENCES. If we detect that portions of your program match portions of any other student's program it will be presumed that you have collaborated unless you both cite some other source for the code. You must not violate UTA, state of Texas or US laws or professional ethics. Any violations, however small, will not be tolerated.