

distributed Systems – Lab 2 Report

Distributed Systems – Lab 2 Remote Method Invocation Report



Submitted: October 14, 2014

By: Devan Shah 100428864

Submitted to: Weina Ma & Ying Zhu

**Lab Report Questions:**

1. What did you use in your implementation to ensure that the Election service records a vote whenever a client thinks they have cast a vote?
2. How did you achieve the guarantee that all votes are safely stored even when the server process crashes?
3. Outline your implementation for ensuring that the records remain consistent when it is concurrently accessed by multiple clients. You should include snippets of your program to help with your explanation.

**Task #1:**

Define the interface to the Election service in Java EMI.

1. The Election Interface can be found in the attached folder **DistributedSystems - Lab 2 - Task 1 Define Election Interface** under **src**. (~/DistributedSystems - Lab 2 - Task 1 Define Election Interface\src\ElectionInterface.java). The Election Interface contains function definitions for *vote* and *result* functions. The vote function takes in a string (candidate’s name) and int (voter number), also returns a Boolean to identify if the vote was casted successfully or not. The result() function takes in no parameters but returns a vector of objects (ElectionResults objects). I have an ElectionResults class that is used to store the election results as a serialized object in a vector. The ElectionResults.java file can be found in folder **DistributedSystems - Lab 2 - Task 1 Define Election Interface** under **src** (~/DistributedSystems - Lab 2 - Task 1 Define Election Interface\src\ElectionResults.java) More details regarding the files and functions can be found in the source files ElectionInterface.java and ElectionResults.java.

**Task #2:**

Implement the Election service in Java RMI. Your implementation should ensure that a vote is recorded whenever a client thinks they have sent in a vote.

Write a client program and run multiple instances of the client to case votes. A client may also query the server for the results and display them.

The Server can be started with command: “java ElectionServer localhost”

The Client can be started with command: “java ElectionClient vote localhost 1099” for voting

The Client can be started with command: “java ElectionClient results localhost 1099” for results

The source can be found under: **DistributedSystems - Lab 2 - Task 2 Implement Election\src** in the submitted zip file.

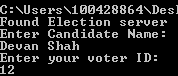
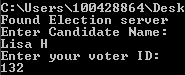
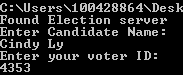
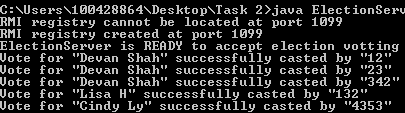
Following are outputs from the client and server when votes are casted/results are retrieved:

Figure Client 5 casting vote

Figure Client 4casting vote

Figure Client 3 casting vote

Figure Client 2 casting vote

Figure Client 1 casting vote

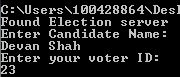
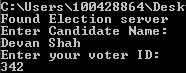


Figure while clients cast vote’s server prints as votes come in



Figure Client querying the results from the server

**Task #3:**

Now modify, if necessary, your implementation to ensure that the votes are recorded even when the server process crashes.

Using the same votes that the 5 clients casted in Task 2, the server stored the casted results in a file “ElectionResultsRawData.ser” and supports restoring the results when server is started again.

The Server can be started with command: “java ElectionServer localhost”

The Client can cast a vote with command: “java ElectionClient vote localhost 1099”

The Client can retrieve results with command: “java ElectionClient results localhost 1099”

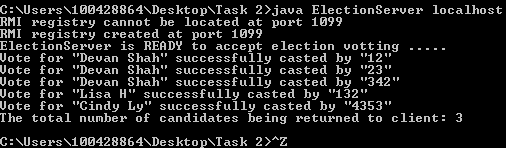
The Server can be killed with ctrl + z

Restart the Server and the server will ask if you would like to restore using the file.

Enter yes and then use the client to retrieve results, the results will be the same as that were retrieved previously.

The source can be found under: **DistributedSystems - Lab 2 - Task 3 Save Results\src** in the submitted zip file.

Following is the sample output following the procedure mentioned above:



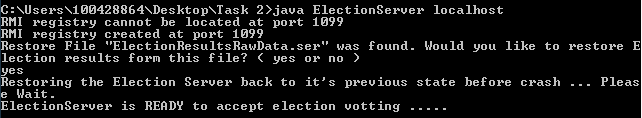
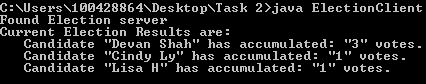


Figure Client querying the server for results after server was restored

Figure After server is terminate and server process is restarted, it will be restored if user wants.

Figure Server was terminated using ctrl + z this has saved the results that were previously sent to the server.

**Task #4:**

Modify your implementation, if necessary, to ensure that the records remain consistent when it is concurrently accessed by multiple clients.