Professor Antokoletz

WRT 245

April 5, 2011

Pg. 224, #2

|  |  |
| --- | --- |
| **TO:** | Jaynam Sha  Steven Rodgers |
| **FROM:** | Andrew Ribeiro |
| **DATE:** | April 12, 2011 |
| **SUBJECT:** | Progress on Distributed Architecture for Cracking the Code |

This report details my progress in developing the distributed architecture for Cracking the Code. The architecture is near completion. Components from Jaynam’s database work are needed to fully complete the implementation. On receiving Jaynam’s work, testing can begin.

**Work Completed**

***Crack Protocol design***

* I analyzed the information that needs to be sent from the client to server and developed an application level protocol to coordinate the data interchange.
  + I placed the protocol design in a table in the analysis document on Google docs.

***Client & server design***

* I designed the server and client based on the requirements analysis document. The design diagrams are located in the analysis document on Google docs.

***Crack Protocol implementation***

* I implemented the Crack Protocol as two classes.
  + The class interfaces are in the CrackProtocolClient.h and CrackProtocolServer.h files.
  + The class implementations are in the CrackProtocolClient.cpp and CrackProtocolServer.cpp files.

***Server implementation***

* I implemented the server based on the server design.
  + Concurrency problems were encountered during the implementation. Multiple clients were modifying the same data in parallel. Mutexes, which were not part of the original design, were added for thread synchronization to the implementation. The design was rectified as a result of this implementation issue.

***Client Implementation***

* I implemented the client based on the client design.
  + The client implementation was straightforward.

***Permutation Generation***

* I implemented the distributed permutation generation module.
  + This module has been tested extensively.

***Simple Sockets Library***

* I created this library to simplify socket communication in C++.
  + There were bugs regarding multi-threaded servers in the first version, so I created subversion 1.1 that removed these bugs.
  + This module has been tested quite extensively, but may require additional functionality for greater control over socket communication.

**Work Remaining**

***Integrating database connectivity***

* My client, server, and Crack Protocol implementations require modules from Jaynam’s work.
  + The server implementation needs a module to connect to and execute queries on the database.
  + The Crack Protocol implementation needs a module to transmit information obtained from the database to and from the client.

***Efficiency improvements***

* Some code in the implementations is inefficient. I have also observed heavy memory leakage.
  + The complexity of the environment, in respect to the many levels of concurrency, makes it hard to find memory leaks.
  + Improvements should take at most one day. All modifications are expected to be small.

***Testing***

* After receiving Jaynam’s work, the system must be tested for correctness before integrating it into the graphical user interface.

***Exception handling***

* Some exceptions are not handled properly. This will be fixed before any other work is done.

***Assessment of Progress to date***

* Most of the work is done. A lot of progress has been made in little time.
* Once I Jaynam’s modules, it should take no more than a week to complete this architecture.
* It is hard to debug the implementations in this complex environment. If many bugs are found during the testing stage, it may take me a lot of time to fix the bugs.
* We are on track for meeting the May 1 deadline.

NOTE: Instead of writing for a non-profit organization, I wrote a progress report for a real project I am working on.