**Project Proposal**

**2/26/16**

**Team Name: Grad #1**

**Team Members:** Isaac Pfleegor, Matthew Perry, Olin Anderson

As a team, we would like to work on the problem of synchronizing multiple clients in real-time over a network. The goal would be to create a rudimentary GUI game that encompasses the Java-thread and socket communication problem and demonstrates our solution to the problem. We feel this project would highlight a number of themes and techniques covered throughout the quarter. This project also presents challenges not covered in class. We would explore techniques for coordinating the remote clients’ viewers, and techniques to hide network lag and recover from lost UDP packets.

**1) What problem you will solve:**

Synchronized multi-client network game using multiple threads to handle game logic, game rendering, and networking communication.

**2) What your goal is and what your team will do?**

The goal is to implement a working multi-threaded solution of the client-server problem using Java threads and Java network tools.

**2.5) What functionalities your team will implement?**

Our team would work on the solution by designing and implementing the server, client, and GUI renderer.  The server would have a different thread associated with each socket input stream. Each input stream would receive the world positions from each client. Another server thread would serialize the state of a shared array storing all of the client positions and broadcast the serialized array to all of the clients at a fixed interval. Each client could have a thread dedicated to receiving the array, parsing it and updating its viewer based on the player coordinates.

The game logic begins with one client represented as a square and all other clients represented as circles. When the square intercepts a circle, the circle becomes a square as well. The objective of the game is to be the last circle remaining.

The world position array would need to be locked during the serialization process from other threads updating it. The server could use a thread-pool to handle client connections which would limit the number of possible players in the game.

**3) What kind of tools and software you will use in your project, such as Amazon AWS S3 or EC2 or MapReduce?**

For this project we will be using java threads on multiple computers. In addition, we will use Network Sockets in a client-server network arrangement to connect the computers, and a basic Java GUI using either Swing or JavaFX.

**4) Please clearly describe what results or outcome you will expect to achieve after project is done?**

The expected outcome is a basic functional GUI game that demonstrates use of synchronized network operations using sockets and threads in real-time.