Networking Plan

Whenever we explore into the future possibilities of our game network support, we must first grasp the notion of a game network and the principles of the internet network's structure. We learned through our research that our game's existing design pattern does not enable game networking. For example, even though we have multi-player games, there are a lot of grey areas (2-4 player). We also didn't consider ((2) inconsistent design architecture (e.g., bad error handling, etc.) which restricts our rendition of the game's ability to be played online.

Given this constraint, our game may still be networked. Initially, because of the strength and simplicity of java programming, it has the potential to be used in future cross-platform multiple game development. Scripting in Java also allows us to offer simple network support. So, for our game networking, we'd go with the industry standard of client-server configuration because it's easier to implement. Because it needs a little amount of bandwidth. So, we'd employ a socket to allow numerous players to connect to a game's host server, which would then allow data to be transmitted across the networks. So, relying on our code architecture, we'd have to create a host server that manages the data and is in charge of connection assistance and game play. The client-server, on the other hand, will oversee presenting the GUI and will be used by the player to connect to the host and play the game. Aside from implementing the server-client sockets, we'd have to make a few tweaks to the game we've built.

In terms of the design for the offline game we've produced and the online networking version we'd want to build. These only change in a few ways, but for the most part, the game is the same. It would operate in the same way. One of its changes would be the definition of a variable that tells the player(s) from the clients-servers end who's turn it is, while also preventing other players (s) from using any game capabilities till the player has taken his/her turn. While it will be the client-job servers to notify the host server when other player turns are available. Although this is still the most basic theoretical method to game networking, it is probable that real implementation will result in complications that are not accounted for in the description. Lastly, if that any player decided to leave the game in the middle, our code would smartly replace the one less player with an AI player, so the gaming experience is the same at all times.