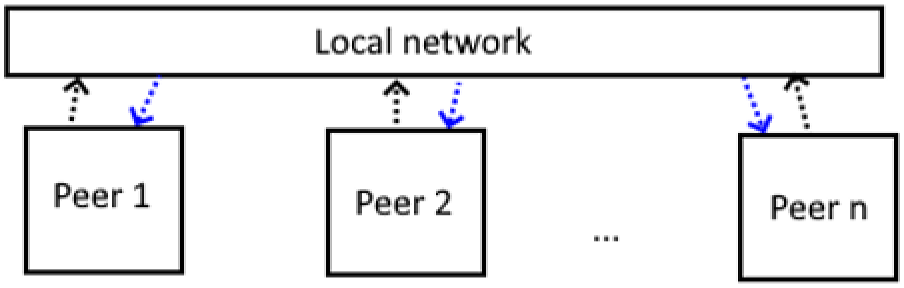
COMP 3010
Network Design Document
Colton Dietterle - 7822763

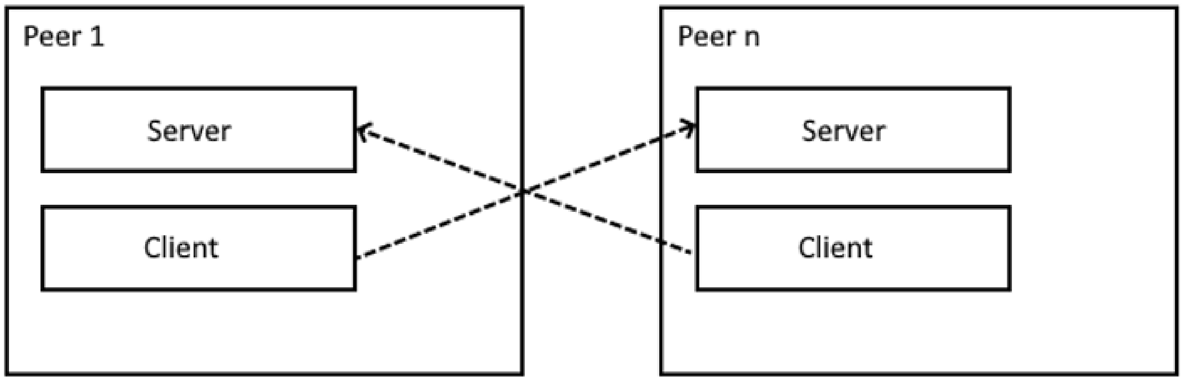
The networking of my project will follow 3 phases. mDNS discovery, initialize the connection, and playing the game. The specific technologies I will be using are Java, libGDX, and JmDNS. To recap my project proposal idea, I will be creating a peer to peer multiplayer game of tag. Users who are 'it' will move their players around a map in an attempt to tag other players.

Phase 1: mDNS Discovery



All peers that wish to connect to the game will send a multicast UDP packet into the local network (Black arrows). Each peer will then listen into the network to discover other peers. Once a peer is discovered, save its information (blue arrow). At the end of this phase every peer will have a list of every other peer.

Phase 2: Initialize the connection



Establish a TCP connection between each peer. Following the definition of a peer-to-peer network from [1], each peer will act as both a client and a server. The client component of peer 1 will connect to the server component of every other peer, and the client component of every other peer will connect to the server component of peer 1 (and so on for all n peers). Following the general idea of [2, pg.223] the network topology will be structured, consisting of every peer being connected to every other peer.

Before moving on to phase 3 some integrity checks will be made here. This includes ensuring that every peer has the same copy of the game. This will stop users from using 'hacked' clients with values that are edited in their favor. For example the map that the players move around in is stored locally, so a player may be able to edit it so that the map they move around in is different from everyone elses. To check this I will serialize important data and have each peer compare their data with everyone else. The game will reject any players who do not have the data matching the majority of other peers.

Phase 3: Playing the game
Following the topology above, the peers will begin playing the game. Each peer's server will listen in for packets from other peer's clients, and each peer's client will send packets to each peer's server when an action is performed. When a peer performs an action in the game, their client component will send 'player n performed action x' to every other peer's server component. Every other peer will then update player n on their games state. To handle what happens when a peer joins or leaves the network, I will be following [2, pg.223].

Example messages that I plan to be passed frequently are: (not in their final formats)
Player n moved to position x,y
Player n tagged player m, player m is 'it'
Player n has joined
Player n has left (+ reason)
etc.

References:

[1] Schollmeier, R. "A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications." *Proceedings First International Conference on Peer-to-Peer Computing*, IEEE, 2001, pp. 101–02, https://doi.org/10.1109/P2P.2001.990434.

[2] Shen, Xuemin., et al. *Handbook of Peer-to-Peer Networking*. 1st ed. 2010., Springer US, 2010, https://doi.org/10.1007/978-0-387-09751-0.