A.I. Logic

Built primarily on a finite state machine, the A.I. uses fuzzy logic to make informed decisions and govern its actions.

Fuzzy logic deals with "ranges" of truth, rather than absolute false and true. Due to this, given the same game state inputs, the A.I. will sometimes make different decisions. The closer the balance of power in the game (territory, buildings, etc), the more likely the A.I. will be to let chance play a role in the decision making, to simulate uncertainty.

There are four states in the state machine; opening, defensive, expanding, and aggressive.

There are currently three different difficulty levels: easy, medium, and hard. The harder the A.I., the quicker the AI begins decision making.



Circuit Generation

Circuits are used in FireWall to transport units between hexes.

Circuits are created by using an A* pathfinding operation. The player specifies specific nodes for the path to travel through, then A* is run on the path to determine the optimal path, making sure to exclude enemy-occupied territory.



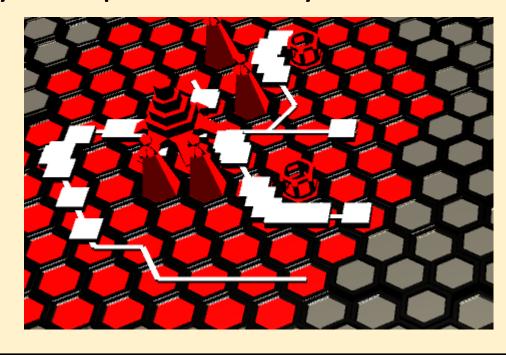
Entry / Less Territory than opponent Do / Expand with circuits going aggressive and unit factories Exit / After a set time Exit / After territory ownership is more even The A.I. will more than likely open up with expanding if territory is uneven The A.I. may go aggressive if, after reinforcing territory the opponent is gaining more territory After taking the oppositions territory, he A.I. may opt to try to expand neutrally

ExpansionState Entry / Less territory The more territory owned by the than opponent Do / Expand with circuits the higher the chance of going defensive and aggregators Exit / After territory ownership is more even

DefensiveState Entry / More territory than opponent Do / Reinforce territory with circuits and hills Exit / After opponent gains more territory

The more territory owned by the opponent, th higher the chance of expanding through aggregato

Creating a synchronized network experience



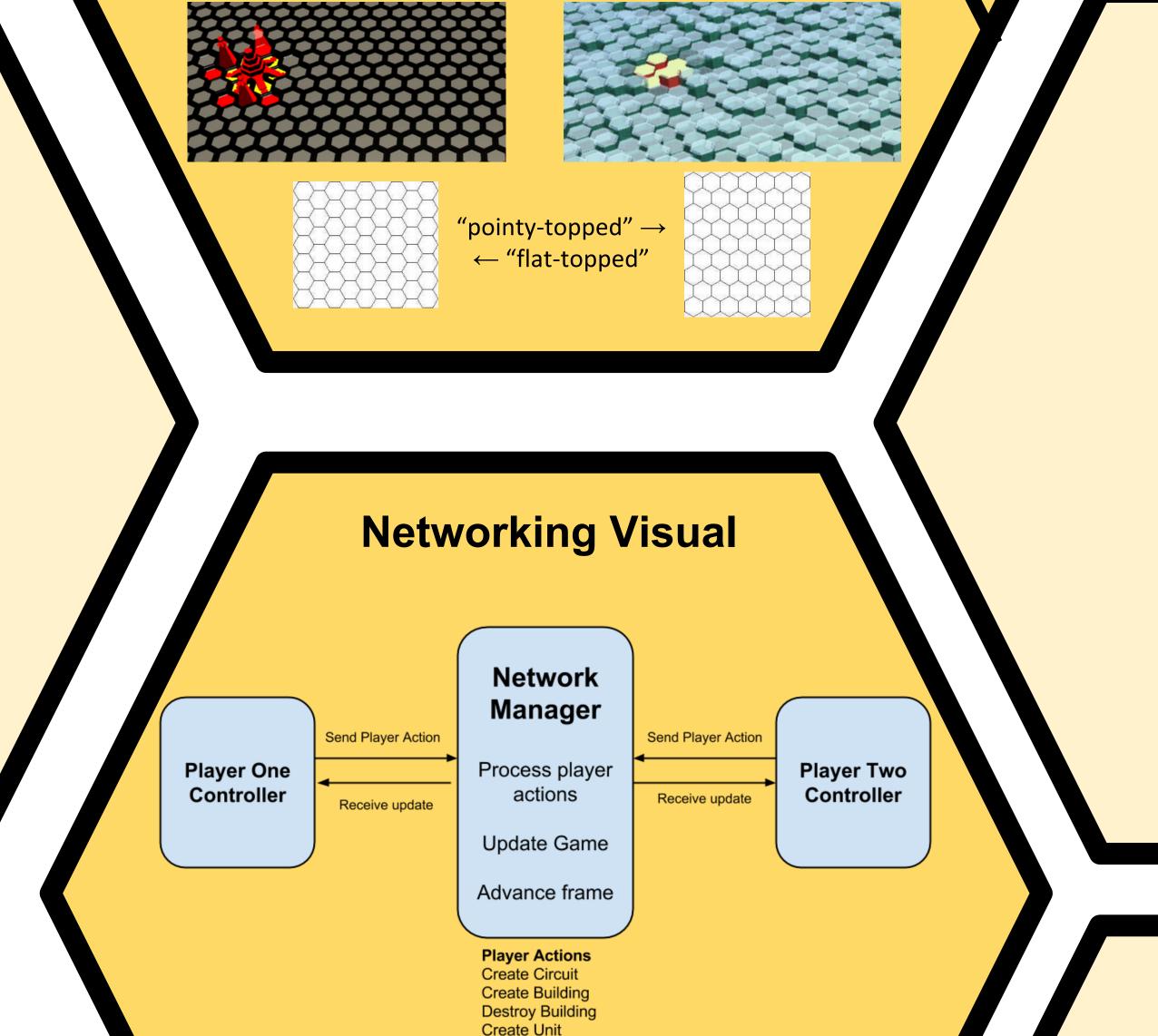
One of the defining characteristics of the game is the hex grid.

Early on, hexes were procedurally generated, but later replaced by a model to reduce complexity.

Used "pointy-topped" orientation of hexes for easier horizontal movement.

Resistances are randomly calculated and reflected by the y-scale of each individual hex.

Early build, very similar to current model

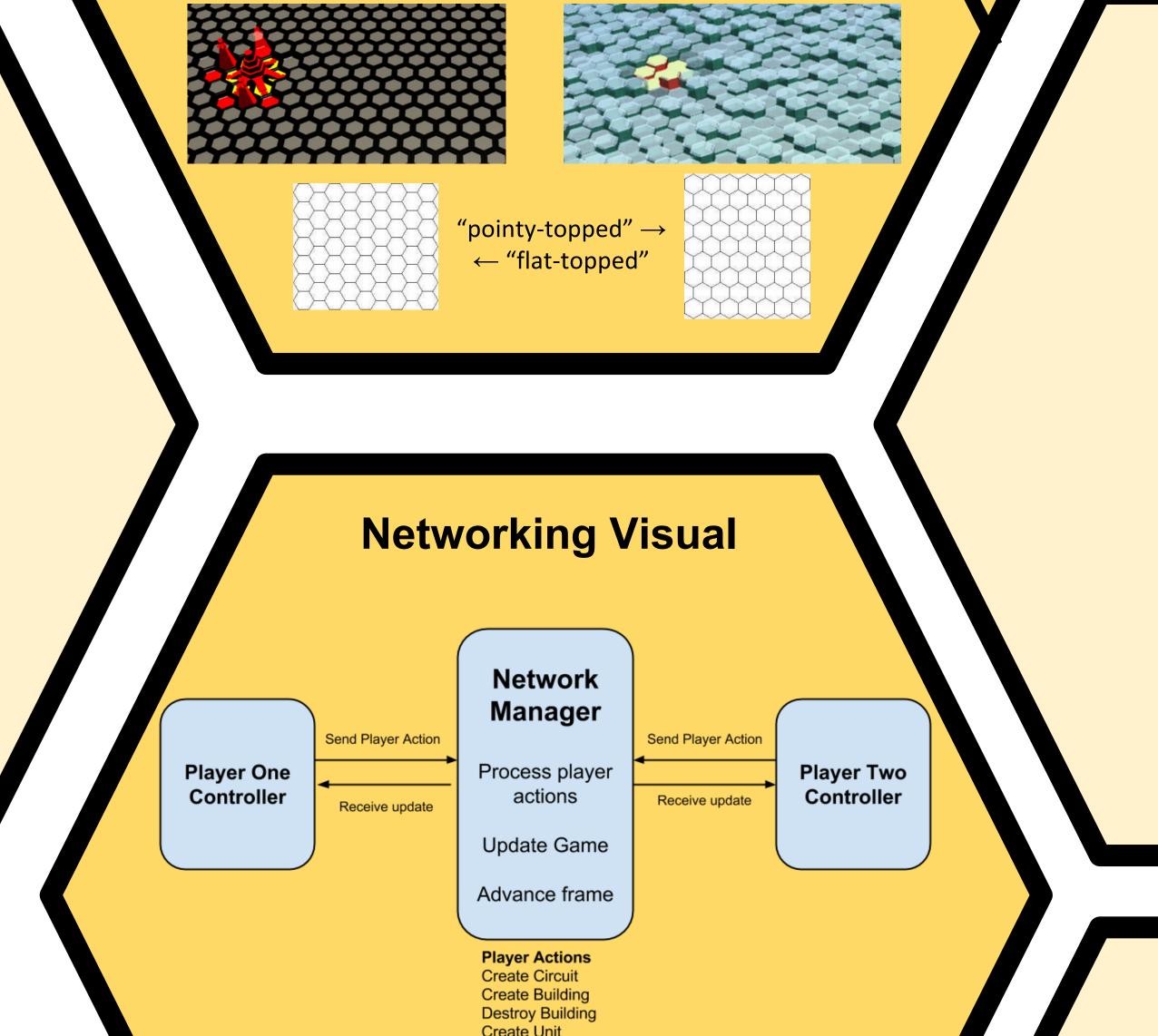


Early build including heightmap

Grid

5

FIO



Developed a base networking class called Action. Any information traveling across the network needed to implement the base class in order to do this.

5L

Verhork

H

When a player performs an action, the controller creates the action and sends it to the network manager. The network manager will not advance the frame until it has an action from everyone.

Other actions were added that don't strictly correspond to some player input:

SeedRandomAction: ensures that random numbers are the same across all instances.

ReadyUpAction: Provides game setup information to both players, including color, name, etc.

Honeycomb Hexertainment