Honeycomb Hexertainment

FIREWALL

Design Document

Zach Atwood
Taylor Eedy
Ross Hays
Peter Kearns
Matthew Mills
Camoran Shover
Ben Stokley

Table of Contents

Introduction

The goal in designing this game was to create a game which would draw no perfect comparisons. We wanted to avoid creating a game which would be constantly heralded as a “clone” or a “remake”. Early on, we focused on holding long brainstorming sessions to really flesh out the type of game we wanted to create. We used these sessions to create an effective team dynamic and develop a game in which we could represent the sum of our interests. In the midst of all this, Firewall was created.

So what is *Firewall*?

Firewall is a real time strategy game focused primarily on territory control. As of now, it is intended that the game will feature one-on-one networked play where players will struggle for control of the map. Each player attempts to destroy the other players main hub by constructing circuits on the map over which units travel to capture territories and generate more energy. Energy is the primary currency of the game, allowing players to create towers and units which can do a range of things, including assaulting enemy territory, increasing energy output, or raising defenses against enemy attacks. The goal of the game is to expand rapidly and force your enemy to concede territory. However, rapid expansion could spread your forces too thin and leave you vulnerable to counterattack. In the end, the player that loses their main hub will lose the game, or whoever controls the most territory as time expires will win the game.

Style

Firewall derives its style from that of computer hardware and technology. To this effect, the entire game centralizes its theme around different circuit and computer jargon. All towers, units, and building names are derived from this idea, including capacitors, resistors, and aggregators. Attempts are made to let the identity and style of each object represent its purpose within the game, but is not consistent throughout the game. We are planning to stylize these elements, however, to add a digital theme to the game.

This includes adding shaders and highlights that are reminiscent of Tron or similar titles. We also acquired a holographic shader pack to add to the digital theme of the game.

Setting

Firewall is set upon something akin to a virtual circuit board, represented as a hexagonal grid. In this regard, the setting has no physical real world space, and should be more viewed through abstraction. Players take control by building circuits and towers which will serve as roads for units in order to promote expansion. Additionally, every hexagon on the grid has resistance, which determines the amount of energy a player must put into it in order to neutralize and capture it. Players are able to modify the terrain by altering the resistance of tiles in order to create walls to prevent enemy advancement.

Rules

Victory conditions

- Destroy enemy base.
- Control the majority of the map as time expires.

Resources

- Energy will be the primary resource, earned by controlling territory
- Certain buildings capture territories increasing income
- Buildings have a cost to build and maintain.

Expansion

- Each player may expand out from their base using circuit paths.
- There is a limit on the circuits each player can create.
- Territory will take time to build along depending on its starting value.
- Buildings may be placed along circuits.

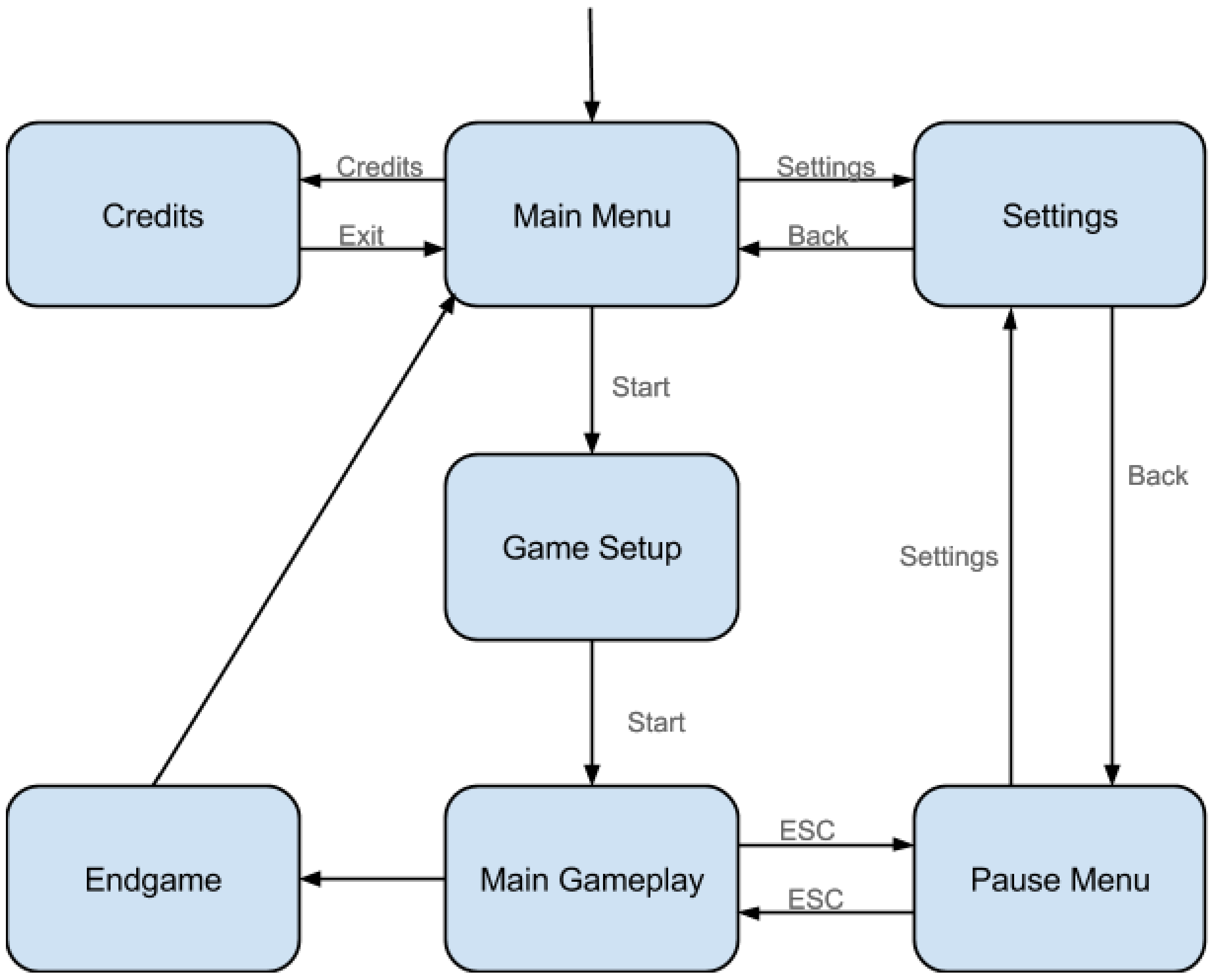
Buildings

- Constructing buildings consumes energy
- Some buildings serve a defensive purpose others allow unit production.

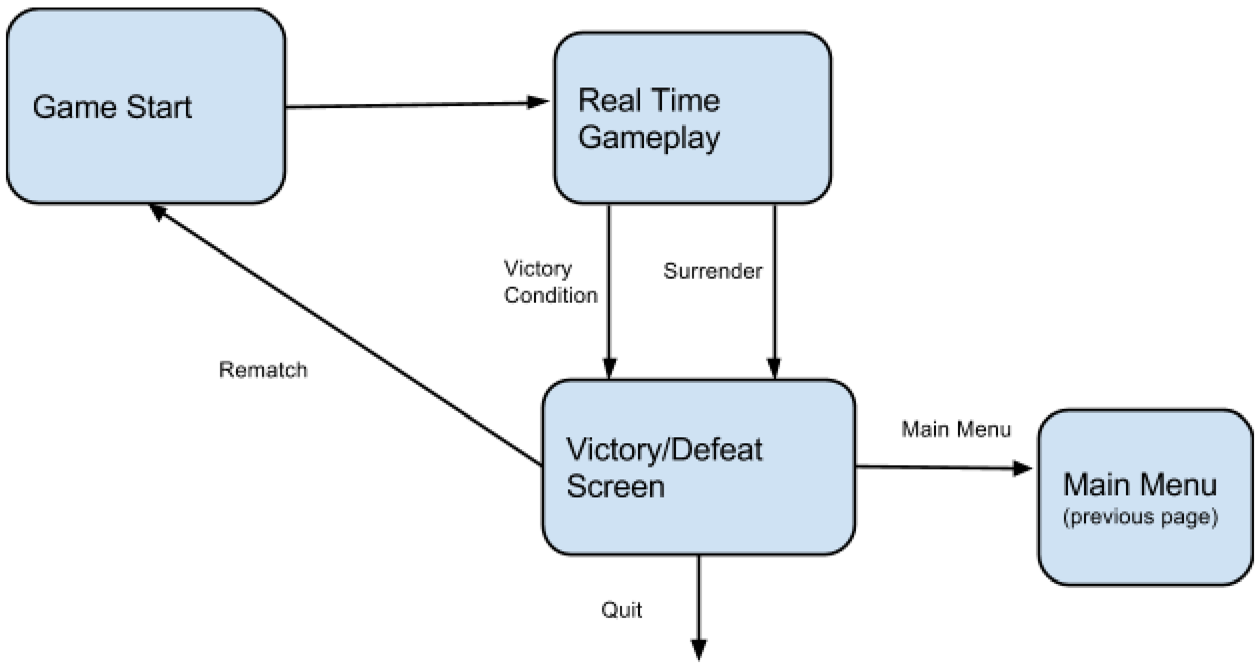
Units

- Created with energy.
- Used to conquer territory, or destroy enemy structures.
- Different units excel at different tasks.

Game States


Credits
Main Menu
Settings
Credits
Exit
Settings
Back
Start
Game Setup
Back
Settings
Start
Endgame
Main Gameplay
Pause Menu
ESC
ESC

Main Gameplay



Controls

Currently controls are implemented with the mouse giving the majority of the commands. Tiles can be clicked on to build circuits or buildings but we have plans to expand these to keyboard hotkeys in the future.

Mouse Controls

- **Left click**: Select a hexagon
- **Right click**: Deselect a hexagon

The UI contains most of the build commands and circuit placement controls. The game can be played entirely with the mouse but keybindings make gameplay faster.

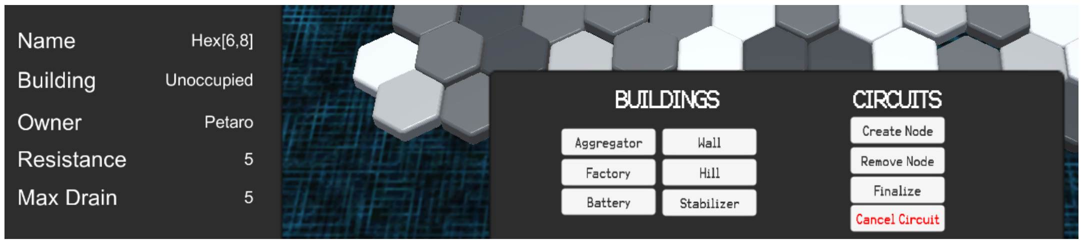
Keybindings

- **F**: Quick build Factory on selected hex
- **R**: Quick build Resistor on selected hex
- **A**: Quick build Aggregator on selected hex
- **S**: Quick build Stabilizer on selected hex
- **H**: Quick build Hill on selected hex
- **C**: Enter circuit mode
    - **Left click**: Set path node
    - **Right click**: Finalize Path
    - **Double click**: Place building

Keybindings can be modified in the options menu of the main menu.

UI

The User Interface contains 2 sub menus. The Hex information menu contains information about the currently selected hexagon (Building, Location, Owner) and the build menu that contains controls for constructing buildings and circuits.



Name	Hex[6,8]
Building	Unoccupied
Owner	Petaro
Resistance	5
Max Drain	5

The command card (pictured in the center) allows players to interact with different aspects of the game. It is a context sensitive menu that determines its state based on the selected hex. The states are:

Circuit Adjacent: The basic function of this state is to allow the player to create buildings. The hex must be adjacent to a circuit and not have a building on it.

Building Occupied: This includes hexes which have a building, and allows players to upgrade or destroy the building. In a special case, selecting a factory will also open the menu to select the production type of the factory.

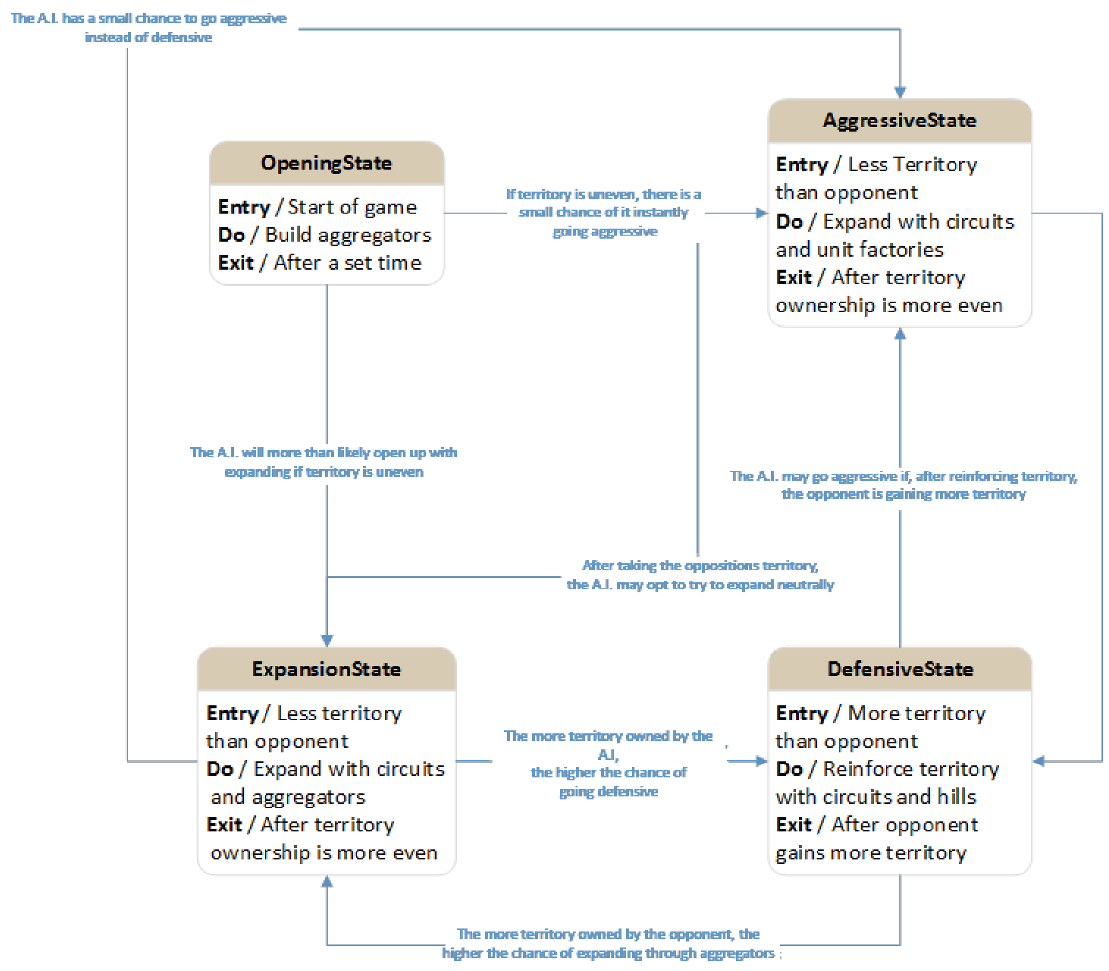
On Circuit: Allows the user to start a circuit. From here, the game switches in to circuit builder mode.

Circuit Builder: The only state not dependent on a selected hex. Once circuit builder is activated, players can create circuit nodes by left clicking on their territory. Once the circuit is finished, the player finishes the circuit by right clicking on the final node.

AI

Our game will have have a single player mode in addition to the multiplayer options. There are several levels of AI difficulty, allowing varying degrees of challenge for a new player of the game.

The AI uses fuzzy logic and a finite state machine to implement several strategies that respond to the state of the game. After the initial build order the AI analyzes the board and switches with some probability to an expansive or aggressive state.

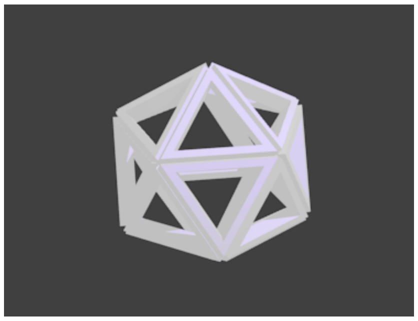

The A.I. has a small chance to go aggressive instead of defensive
AggressiveState
Entry / Less Territory than opponent
Do / Expand with circuits and unit factories
Exit / After territory ownership is more even
OpeningState
Entry / Start of game
Do / Build aggregators
Exit / After a set time
If territory is uneven, there is a small chance of it instantly going aggressive
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, the A.I. may opt to try to expand neutrally
ExpansionState
Entry / Less territory than opponent
Do / Expand with circuits and aggregators
Exit / After territory ownership is more even
The more territory owned by the A.I, the higher the chance of going defensive
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, the higher the chance of expanding through aggregators

UNITS

Firewall has a variety of units, all with different abilities to make them practical for different types of attacks. Units are self-controlled. Upon being built, they progress along a placed path and attack enemy territory upon arriving.

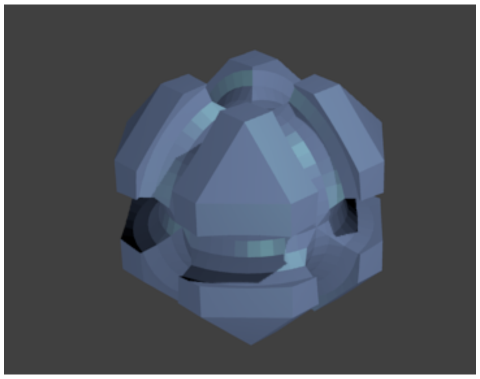
Packet-- utility

The base unit is called a packet. Packets travel along wires and randomly attack enemy territory when they encounter it. It also may disperse energy to buildings.



Bomber -- assault

The bomber is a variation of the basic unit which detonates on impact with enemy territory. The detonation damages all enemy territory adjacent to the detonation point with relatively low damage.

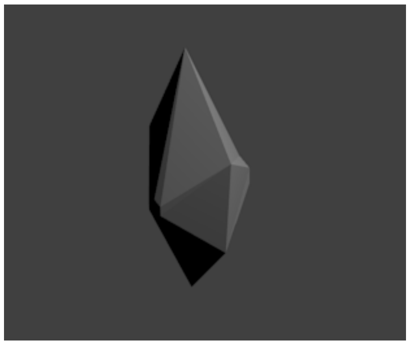


Disperser -- assault (//Not Implemented)

The advanced variation of the bomber has the same functionality, but its explosion now carries through multiple objects, damaging all enemy territory within its radius with damage relative to the distance from the explosion.

Spiker-- assault

The spiker is a variation of the basic unit which directly assaults a single enemy hex at the end of the wire. It hits for a particular hex for a more significant amount of damage. If the spiker neutralizes a hex, that hex is captured and a circuit is placed on it.



Emitter -- assault (//Not Implemented)

The advanced version of the spiker has the same functionality, but its direct attack now hits all enemy territory in a straight line from the attack, with damage dealt being relative to distance from center.

Engineer -- defense (//Not Implemented)

The engineer is a defensive fortification unit. At the end of the path, it provides a resistance boost to any tiles it is adjacent to.

Trojan -- viral (//Not Implemented)

The trojan is a viral unit which can be injected into enemy territory to instantly capture a hex and its surrounding area.

Logger -- viral (//Not Implemented)

The logger is a viral unit which can be injected into enemy territory to track player actions in that area.

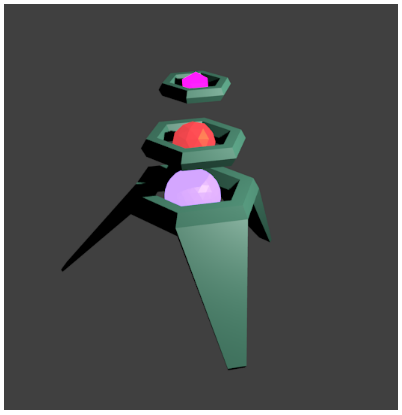
Spy -- viral (//Not Implemented)

The spy is a viral unit which can be injected into enemy territory in order to lift the fog of war from an area around it.

BUILDINGS

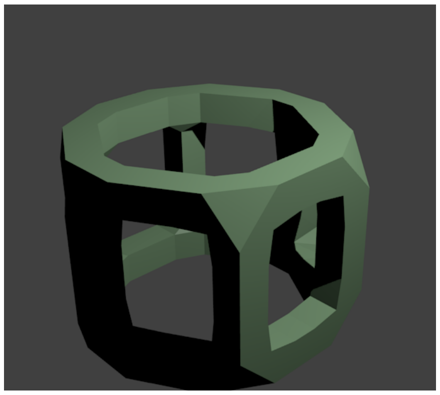
Main Base

The only building you have at the beginning of the game. The Main Base has 6 circuit nodes for initial path construction, and controls a depth of 2 hexagons from its center. The main base also produces the Packet unit type, this can not be changed.



Factory

The producer manages all the assault and defense units. The player selects which unit the building should be producing and the building produces those units automatically as time passes and energy is generated.

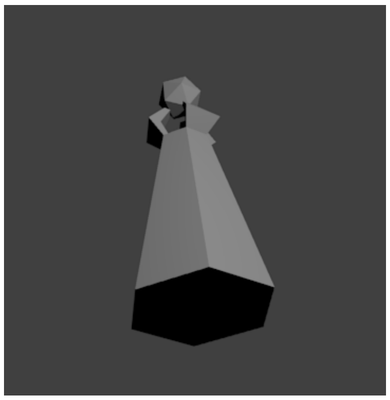


Injector -- (//Not Implemented)

The injector produces all the viral units as well as being the staging area for their placement. A player will select a virus to build from the building menu. When the virus is complete, the player selects an enemy hex which is within the active radius of the building, and the buildings injects the virus into that location.

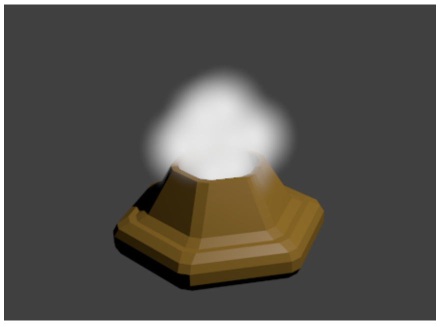
Aggregator

This building allows players to capture surrounding neutral territory over time. After being placed, the aggregator will slowly branch and lower the resistance of surrounding neutral territory. It has a max radius and can be upgraded into a collector after capturing all the territory possible.



Stabilizer

This building increases energy output of any lower resistance tiles around it. When created, it lowers its tile to zero resistance, effectively making it easy to take with any form of attack. Any tiles with a resistance lower than 0.2 within its active radius will double in energy output.



Resistor

The resistor allows for the of the territories around it to be permanently raised. It comes in two primary flavors:

Wall (//Implemented but removed)

Upon placing the resistor node, the player can select four nodes to build up. The four selected nodes scale their new resistance based on their distance from the wall node. If a player creates two wall nodes within four units of each other and connects one wall to the other they will all raise to max resistance.

Hill

Upon placing the resistor node, the central hex raises to mid resistance, and then it branches out, with some resistance loss at each level. This resistor is used to terraform and provide some higher level resistance against enemy units that may eventually be moving through the territory. Upgrades to the hill resistor will cause the central resistance to raise.

Networking

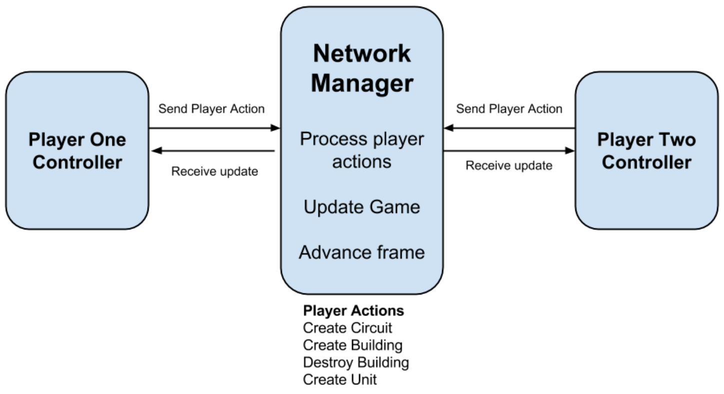
For a RTS style game having a synchronized connection is critically important. To this end we implemented lockstep networking, which ensures that both parties send actions to a network manager that then waits to advance the frame until it has actions from both parties. Actions were a base class created to pass the information across the network connection.

All of the networking was implemented inside Unity, while this reduced dependencies on 3rd party plugins we believe that using a plugin might have made the development go smoother.

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



Economy

The economy of the game was one of the hardest elements to create and balance. Initially we planned to make the players income be solely dependent upon the number of owned and unoccupied hexes. We decided to add another layer of complexity, and developed the system or resource sources and drains.

The way the new economy system works, there are resource sources and resource drains, which are outlined as follows:

Rate
The rate defines how much money the player is receiving per second. Simply put, the rate is defined by the amount of resource sources against the amount of resources drain.

Resource Source
A resource source is any hex grid object. It provides a standard amount of resources per second based on it’s resistance. The higher the resistance, the lower the resource output.

Resource Drain
A resource drain is an object built on the grid which pulls resources from the player rate. It includes all structures the player can build. Some buildings, like the stabilizer, act by countering the resource drain within in area.

As an addendum, the resource system in it’s current state still needs a great deal of balancing. Players gain resources at an extremely fast rate, but adjustments are being made.