Artificer MOBA Build Design Document

# Objectives

1. Add network connections between local and remote clients, simple verse maps
2. Update Menu UI to MOBA Replace single and multi player with ‘Find a game’ and ‘Quick Connect’

# Description

// Create new MOBA description

# Build Overview

Artificer MOBA will have the following play style

* 16 a side ship battles online
* Each team spawns with a large station on one side of the map
* Teams can set up base construction (A ship will travel from base to the build site and needs to be protected
* Ships spawn from main and forward bases
* Garage to build ships
* Player has free-look and ship follow, free look allows the mouse to drag the screen
* AI pirate groups and cargo escort groups
* Free floating wreckage generation
* Asteroid fields
* When player zooms out go to map view
* 10 templates to pick from when picking a ship before match

## After GreenLight

# Design Plan

## June

Following an unsuccessful greenlight and some UNET + Steamworks/.Net tutorials, I have decided to implement online functionality over June including matchmaking with steamworks and using UNET for creating peer to peer networking.

### Week Four - Creating Player ships on client/host

* Rethink ship structure
  + Make shipdata able to sync fully (20/06/16)
  + Create functional ships on all clients with ship generator (21/06/16)
  + Sync move and rotation (22/06/16)
  + Sync input keys across server (or make listener activation and deactivation as a [command] for visual effects (23/6/16)
    - Only localhost will show visible engines (23/6/16)
  + Fix mouse follow issue (23/6/16)
  + Show rotor effect syncing when turning rotor (23/6/16)
  + Firing weapon spawns projectile on server
    - Create projectilespawn obj with ProjectileSpawner script(23/6/16)
    - Create static function that accepts bullet prefab and weapon data for firing bullets (23/6/16)

## July

### Week One & two - Creating Player ships on client/host

* Create functional ships online
  + Firing weapon spawns projectile on server
    - Beam weapon is visible on client and server – add other projectiles later (4/7/16)
    - Instantiate and network spawn muzzle and impact sounds (4/7/16)
    - Projectiles successfully damage others. (7/7/16)
    - Sync damage (8/7/16)
      * Performance boost, see if components are damaged more than once (11/07/16)
    - Sync component destruction
      * Create a script that loops through each destroyed comp and physically remove them using the instance ID (12/7/16)
      * Create clientRPC and command to pass int array that then uses script mentioned above (12/7/16)
      * Create WreckagePrefab for network spawning (12/7/16)
      * Spawn WreckagePrefab to server (able to destroy) (12/7/16)
      * Add sync ability for child components on wreckage(12/7/16)
      * Performance boost and movement syncing(13/7/16)
  + Place projectiles in their own namespace(14/7/16)
  + Place ship scripts in their own namespace (14/7/16)
  + Network spawn to run on command function (15/07/16)
  + Write xml comments and regions in scripts (15/07/16)
  + Projectile sounds are applied at their correct position for volume
  + Clean ship attributes (15/07/16)
  + DPC implement Hit virtual functions (15/07/16)
  + Fix Projectiles hitting ship but not damaging individual components (15/07/16)

### Week Three – Custom Player Spawning

* Change PlayerSpawn to TeamASpawn and TeamBSpawn and alinement labels custom spawn functionality
  + Create team spawner manager (takes team sizes as a variable) base class for different teams
  + Team spawner consists of three functions
    - Small teams (up to 32 players) – two station at opposite ends 200km away from each other, create TeamA and TeamB Spawns around these (don’t bother with two other functions yet)
      * Add Network Identities to station generator
    - Team Spawner Manager will store a dictionary<short, Transform> for current ships
    - And store <short, ShipData> for player accounts. Shot being client ID
  + Don’t sync alignment labels so some will be enemies and others will be friendly
* Create faction managers again?
* Before spawning allow player to choose side (Create new popup)
* Store ship data within space attributes for player (ShipData PlayerShip)
* Delete any AI waypoints, contract specific station data
* Override on OnServerAddPlayer(NetworkConnection conn, short playerControllerId) that sends a player pending to the space generator. A third parameter network reader will be used to pass the player’s steam name and shipdata to pass to the space generator.
* When the ship is generated, the game manager will add that ship to server using: NetworkServer.AddPlayerForConnection() be sure to store player id to assign to that object.
* Use for respawning the player ship in the event of the ship being destroyed NetworkServer.ReplacePlayerForConnection()

### Add Networking to player Ships

### Matchmaking

* Create chat message window
* Receive and process chat messages (ON HOLD)
* Test matchmaking with two computers
* Set every UI widget that displays user information to register for the PersonaStateChanged\_t callback, and update on any change.
* Make sure UNET can then take over with peer to peer networking.
* Add ability to kick players

### PlayerData and Stat Storage

* Create StatTracker– controller component with stat storage that’s responsible for keeping track of stats and updating steam. Get and sets
* Add player info in lobby list to display player stats

## Extra

### Server

* Create a new Artificer Project called Artificer Server (NetworkManager.StartServer()) This should possibly have all the same game assets? Possibly keep on same project with different scene.
* <http://www.alanzucconi.com/2015/09/02/a-practical-tutorial-to-hack-and-protect-unity-games/>
* <http://en.unity3d.netobf.com/Video_help>
* https://www.hackthis.co.uk/articles/game-hacking-chapter-1-unity3d-attack-by-reverse-engineering
* Create normal map and assign it when creating component
* Not all sockets within proximity light up when connecting components in unity editor
* Add multi language support (Russian)
* Repair self components