

## Design Document

### Tools and Technologies

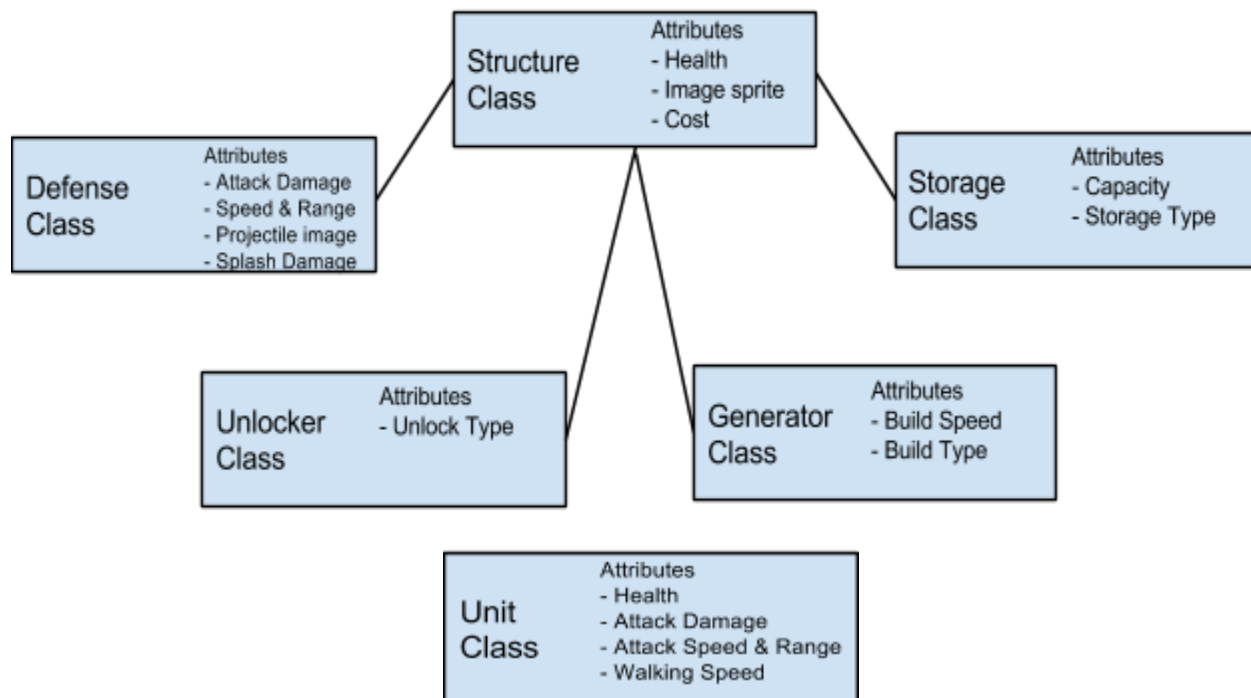
- CreateJS
  - Client side framework for creating all graphics and animations
- Socket.IO
  - Client socket connections
- NodeJS
  - All server code
- HTML5
  - Views
- CSS3
  - Styling

### Data Structures

Grid square - a class containing information on whether or not the square is occupied and if so what type of object is on it (defensive or offensive).

Game grid - two dimensional array of grid squares.

Structure class hierarchy:



### WebSockets Implementation

All communication will take place using Socket.IO and NodeJS. A number of listeners will be set up on the clients and server, which will handle the following:

#### Nodejs/Socket.io listeners

- Game lobby listener
  - Player joins game
  - Player readies up
- Game state listener
  - Player status
    - In lobby
    - In game
    - In victory/defeat screen
- Unit performed action
  - Unit location changed
    - x, y pos
  - Unit attribute changed
    - Health
    - Alive/Dead

## Algorithms

- **A\***

All zombie movement will be controlled by the A\* pathfinding algorithm. The game grid will be broken up into a graph structure where squares that are occupied by structures will be blocked and all other spots will be free. This algorithm will allow the zombies to find the shortest path to their target and how to get there without moving through walls and buildings.

- **Opponent Targeting**

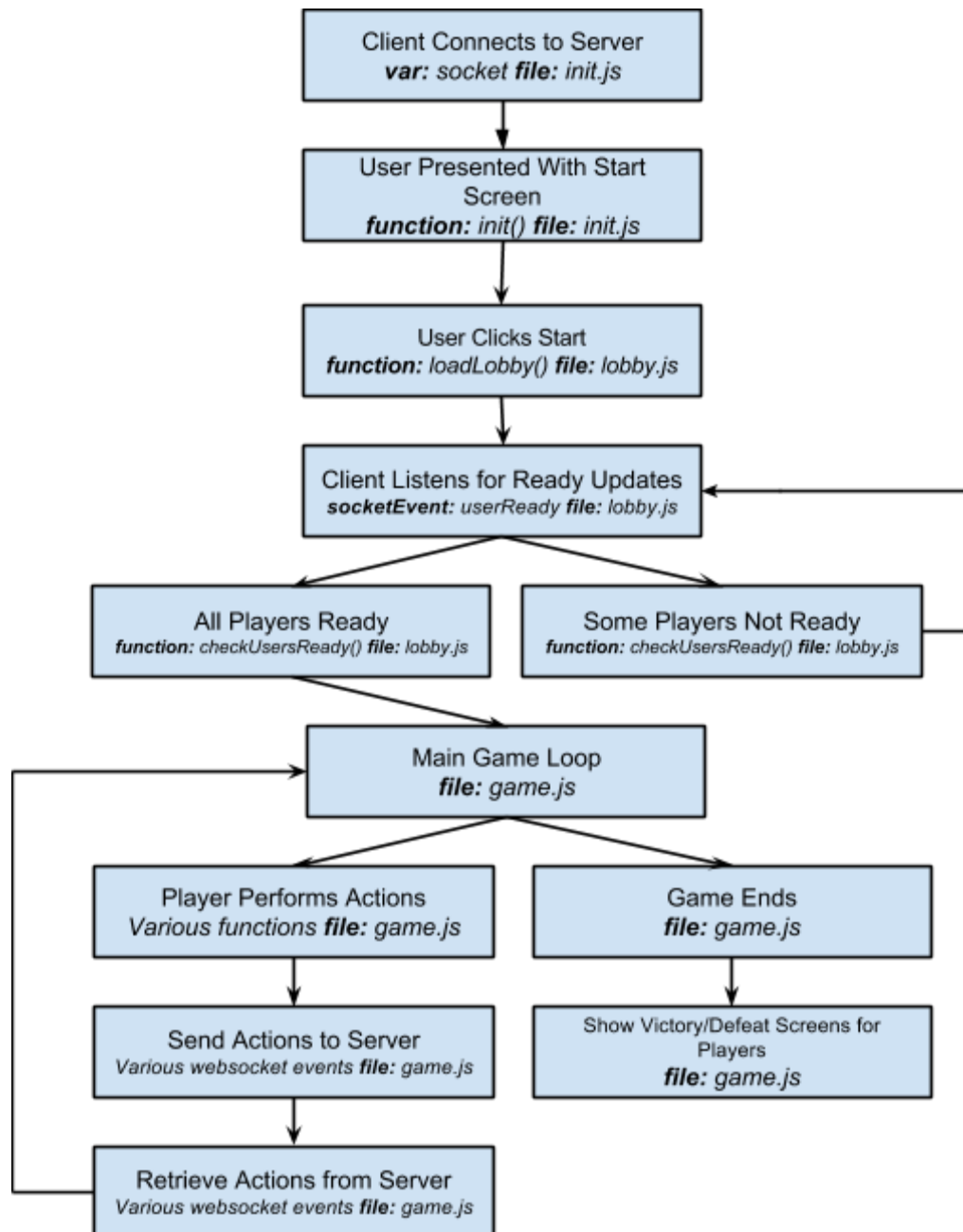
Zombies and defense structures will choose their targets based on euclidean distance. They will swarm to their nearest opposing target and strike. This allows zombies and defense turrets to attack different targets and not just focus on a single goal.

**Hosting:** <http://students.cse.tamu.edu/devint1/outbreak/>

**Repo:** <https://github.com/clement360/Outbreak>

**Development Log:** <https://github.com/clement360/Outbreak/commits/master>

## High-Level Process Flow



When the player first starts the game, he or she will be presented with the splash screen. From there, the player will be able to either start the game or watch a tutorial. If the user starts the game, the client will connect to the server and listen for ready up events until all players are ready. Once all players are ready, the client will enter the main game loop. Each player will perform his or her actions, which will get sent to the server. At the same time, the client will also be listening for game events from the server and will synchronize each player's game state accordingly. Once a victory condition is met, the game will end and victory/defeat screens will be shown to the players.

## **Gameplay**

When the game starts, each player will be presented with the game grid. The first action players must take is to build various structures or zombies in their base. Buildings will perform various tasks, such as unlocking zombies, increasing income, and defending against enemy attacks. Zombies will be placed in a zombie cage on the player's base, and will move toward the enemy team when the player clicks the attack button. The zombies will begin attacking enemy units that are in range, and if not killed, will eventually attack the enemy's base. Once a team's base is destroyed, that team will lose, and the opposing team will win. The victory condition makes the game worth playing for players, and contributes to meaningful play.