Technical Specification

Team Members:

David Gershuni, Shiyao Wu, Kaiwen Shi, Woye Lin, Jon Koehmstedt, Emily Pham

Based on our last high-level requirements, we have the following technical specification:

GUI

Each player will be displayed a GUI containing following information:

• Global-Map. The whole map will consist the following: bases of two teams, with each team’s turret nearby; Tiles in which tanks can locate; Resource pool where the tank can collect resources to upgrade its weapon; Barriers that prevent tank from locating in certain tile. The global map will be a image.

• Mini-map. While all players battle on the same global map, each player will have its unique Mini-map representing his own visual field. The Mini-map will be a subclass of JPanel and will be drawn on that panel . That Mini-map should be dynamically changed while the player’s tank is moving around the global map.

• Base. During the game, each player will be able to see the base health (int) of both teams on the screen. However, the player will only be able to see the base resources in his own team.

• Tank Info. Each player will be able to see his own tank health and tank resources. We will use integer to represent a tank’s health. Initially, all tanks will have a health of 100. During the game, when a tank is shot by the opposing tank’s ammo, it will lose health equals to the ammo’s damage. A tank is died when its health becomes less or equal to 0.

• Purchase Menu. During the game, each player will be able to upgrade its tank weapon in the game store. To do that, the GUI will display a Purchase Menu that shows all the products with the price that player can purchase. If the player don’t have enough money to afford certain product, the game won’t allow the user to buy that product.

• Chat. The user should be able to see all players in the current game on the GUI, and be able to select anyone from those to communicate with. The player should also be able to do the team chat, in which all team members should be able to receive message from this player.

Tank

• Tank health (int health). Each tank initially will have a health value

100. When it gets shot, the health value will be decreased. When the health value is equal or below 0, the tank will die.

• Tank’s Location (int x, int y). Each tank will have a specific location at a specific time represented by (int x, int y). These values will correspondingly change when the tank moves on the map.

• Tank Speed (int speed). Each tank will have a specific speed on a certain time. The speed may increase once the tank gets upgraded.

• Tank Direction (int direction). Each tank initially will have 4 directions to move. We use integer to represent its direction: North = 1, South = 2, East = 3, West = 4. Once the tank gets updated during the game, it may have more directions, such as North\_East, North\_West, South\_East, South\_West.

• Ammo. Each tank will be equipped with ammo to destroy opposing tanks

and bases. There should be different types of ammo. The following things will be needed to represent each type of ammo: type name (String), damage (int), speed(int), and shot-rate per second (int), color(Color). We use color to give

a different appearance for each type of ammo. At the beginning, all tanks will have the basic type of ammo. Once the tank collects enough resources, the ammo should get updated, with an increasing in its damage, speed and

shot-rate per second.

• Weapons Enabled (boolean enable, int time). When the tank collected the resources on the map, the tank won’t be able to attack opposing tanks for 5 seconds. This would be the cost for a tank to collect resources. After 5 seconds, the tank’s weapons will be toggled to enable again.

Team (ArrayList<Player> team)

• There will be two teams in the game. Each team will be made up with one or more players driving tank. Each player will be a class containing the following: Player’s ID(String), Tank(Class), and its own field of view in the map. Each team will be an ArrayList of all players belong to that team.

Turret

• The Turret will be built near the base of every team. Initially a turret will have a health of 100. When it gets shot by opposing tanks, it will lose corresponding health as tanks do. Turret will be able to use ammo to attack opposing tanks when they are nearby. The Turret will die once its health is below 0.

Repair Kits

• Each repair kits will contain a health point (int) that can be applied to the base. When tank apply a repair kits to its base, the health point of that kits will be added up to the existing base health.

Win/Loss

• The base health score (int) will be used to determine the winner. The game will be at most 10 minutes. The base score will be the sum of health of all tanks on the same team plus the health of turret on that team. Each tanks will try to attack opposing tanks and turret. When a tank or a turret get shot, the base on that team will lose corresponding health score. Whenever a base’s score is equal or below 0, that team will lose the game. If after 10 minutes, both base still have health score remaining, whoever has the most health score remaining wins the game.

Keyboard Handling

• During the game, the user should be able to use keyboard to control the tank. The four arrow key in the keyboard will be used to move the tank to certain location, which is to change the tank location into another

tile. Another key will be used for a tank to attack opposing tanks/turrets.

Server

• The server sill add multiple players into one game. We’ll use database to keep track of each player’s game record. After each game, the server should also record the game result statistic and put those records into

database. After all the players are added in, the server will start the game. There should be a timer displayed on the game which would allow at most 10 minute. The server will use that timer to keep track of the game time. When it reaches 10 minutes and still no base are completely destroyed, the server should end the game automatically and export each players record to the database.

Client

• Each player will be a client. The client can request server to add him into a game at the beginning. Different clients should be able to communicate with each other during the game.