  
  
  
CS 319 - Object-Oriented Software Engineering  
Analysis Report  
  
Tanks Game  
Yryskul Turduev

Hamza Saeed Khan  
Soleh Ruziboev  
Seyedehshahrzad Seyfafjehi

Table of Contents

[1. Introduction 2](#_Toc310454890)

[2. Overview 4](#_Toc310454891)

[2.1. Tank 4](#_Toc310454892)

[2.2. Shells 4](#_Toc310454893)

[2.3. Mines 4](#_Toc310454894)

[2.4. Life Bar 4](#_Toc310454895)

[2.5. Health 5](#_Toc310454896)

2.6. Obstacles..........................................................................................................................5

2.7. Map..................................................................................................................................5

2.8. Controls............................................................................................................................5

[3. Requirement Specificaiton 6](#_Toc310454897)

[3.1. Functional Requirements 6](#_Toc310454898)

[3.1.1. Play Game 6](#_Toc310454899)

[3.1.2. Change Settings 6](#_Toc310454900)

[3.1.3. Pause Game 6](#_Toc310454902)

[3.1.4. End Screen 6](#_Toc310454903)

[3.1.5. View Credits 6](#_Toc310454904)

3.1.6 Multiplayer...…..……………………………………………………………………………………………………………..7

3.1.7 Exit Game....……………………………………………………………………………………………………………………7

3.1.8 Help....…………………………………………………………………………………………………………………………….7

[3.2. Non-Functional Requirements 7](#_Toc310454905)

[3.2.1. Game Performance 7](#_Toc310454906)

[3.2.2. User-Friendly Interface 7](#_Toc310454908)

[3.2.3. Extendibility 7](#_Toc310454909)

[4. System Model 8](#_Toc310454910)

[4.1. Use Case Model 8](#_Toc310454911)

[4.1.1. Play Game 9](#_Toc310454912)

[4.1.2. View Help 9](#_Toc310454913)

[4.1.3. View Credits 9](#_Toc310454914)

[4.1.4. Change Settings 10](#_Toc310454915)

[4.1.5. Exit 10](#_Toc310454916)

[4.2. Dynamic Models 11](#_Toc310454917)

[4.2.1. Sequence Diagrams 11](#_Toc310454918)

[4.2.1.1. *Game*](#_Toc310454919) *Setup*

[4.2.1.2. *Game*](#_Toc310454920)

[4.2.1.3. *Main*](#_Toc310454921) *Menu Options*

[4.2.2. Activity Diagram 14](#_Toc310454922)

[4.3. Object and Class Model 15](#_Toc310454923)

[5. User Interface 18](#_Toc310454924)

[6 Glossary 19](#_Toc310454925)

**1. introduction**

The Tank Wars is an arcade game with a built map in which tanks attempt to shoot at each other in order to win the game. It is a multiplayer game; to be exact it has 2 players who control the tanks and their movements in the map using specific keys or the separate controllers connected to the computer. The map consists of various objects each with a specific purpose. It has two tanks managed by players, 3 types of obstacles, the breakable, unbreakable and transparent ones. Moreover, at random times of the game the health power-ups and mines will appear, also at random positions. The tanks also have an option to set the mines in the map.

Each player will control the tanks The Tank Wars is going to be a desktop application developed using object-oriented models. The main programming language used will be JAVA; however, there might be some details implemented in different OO languages. The IDE for JAVA programming is going to be Eclipse (several versions such as Mars 1, NEON 3).

Different to other similar arcade games, the Tank Wars will have diverse types of effects built in using several available tools such as JAVA effects.

**2. Overview**

Tank Wars is a two-person action game. The objective of the game is to beat the opponent by destroying their tank. The tanks spawn on the opposite ends of a map that comprises obstacles, which block the tanks’ way, and landmines that appear randomly on the map. The map also occasionally shows health objects that can be picked up to reduce the damage to a tank. To move through the map, the tanks can blast through one kind of obstacles but not others. Each of the two users get a tank with a supply of ammunition in the form of shells and mines. The shells can be shot at the opponent’s tank and the mines can be planted for the opponent to tread over. The users have life bars that decrease by the level of damage that the tanks incur. When the life bar for a tank reaches zero, the remaining tank is declared the winner and the game ends.

**2.1. Tanks**

The tanks are the objects that the users manipulate. A user can move the tank, shoot and can plant mines with it. The tank can move in all four directions on a two-dimensional map. The tanks have life bars that show the extent of the damage that has been received. Each tank has an unlimited supply of shells but a limited number of mines.

**2.2. Shells**

These are the objects that the tank shoots. They do a constant amount of damage and move in a single direction in a straight line. The shells are shot in the direction the tank is facing.

**2.3. Mines**

Mines do more damage than the shells. A tank can plant mines on the map but not shoot them. In addition to the mines that the tank plants. The game itself randomly props up mines on the map. A mine blows up if a tank treads over them or if the mine timer ends.

**2.4. Life Bar**

Each tank has a life bar that shows on top of the screen. The life bar how much more damage the tank can sustain. Shells and mines decrease this life bar. A mines effect on the bar is four times that of the shell.

**2.5. Health**

Health objects very occasionally pop up on the map. If the tank picks up a health object, the life bar increases by a certain amount. The health objects can be picked by the tanks by treading over them. The health objects are destructible. They can be destroyed by shooting them with shells. The health objects disappear of not picked in a certain amount of time

**2.6. Obstacles**

Obstacles are wall like objects that make it difficult for the tanks to move. Obstacles are of two kinds, breakable and unbreakable. Shells have no effect on unbreakable obstacles but can destroy a limited part of the breakable obstacles, so that the tank can move through an unbreakable obstacle. Mines have no effect on either type of obstacles.

**2.7. Map**

The map contains all the other objects and determines their positions. The map size is constant and equal to the screen size of the device. The tanks cannot move out of the screen size as no new map portion is rendered.

**2.8. Controls**

The users control the tanks through keyboard buttons. The arrow keys control the movement of one tank and the spacebar shoots its shells. The keys A, S, D, W control the movement of the other tank and C shoots its shells.

**3. Requirement Specification**

**3.1. Functional requirements**

**3.1.1 Play Game**

Tank War is a basic type of classic “Tank” game. The main purpose of the game is to destroy your opponent by shooting his tank. The game is played by two players. In the beginning of the game, Players have 100% lives and 3 mines. Playground has 3 type of obstacle such as breakable, unbreakable and transparent. Two players try to kill each other and who first destroys opponent he wins. Players can drop mines while driving the tank. The mine lives for some time. If no one touched it, mine automatically explodes. In order to provide tanks by mines and lives, they are produced automatically and placed randomly along the playground. Lives and new mines also have own timer, if no one picks it up, it disappears automatically. These extra features make game much more challenging and enjoyable.

**3.1.2 Change Settings**

Player can change some of the game settings according to his own demands. The settings that can be changed by the Player are listed below:

* Music
* Sound
* Background Picture

Player can turn off or turn on game sounds and music from Change Settings. Player can select background picture before starting the game from Change Settings.

**3.1.3 Pause Game**

The game can be paused during gameplay and Player can continue game from where he paused. When game is paused, closing application will cause any game progress to be lost.

**3.1.4 End Screen**

When one player wins, system pops up an “End screen” which displays the winner, its health and mine amounts. In addition, the screen has home button to go to Main menu and restart the game.

**3.1.5 View Credits**

Player can reach contact information of the game developers which included in View Credits in order to give Player a chance to communicate with designers and help the development of the game by sharing his own ideas and suggestions.

**3.1.6 Multiplayer**

It is two player game. Every tank has its unique color and mines. Players control the tanks by keyboard. In the future, we want to make it four multiplayer game which will be controlled by joysticks.

**3.1.7 Exit the game**

Main menu has exit button. When player presses it, it closes all game and exits. Additionally, player can exit from the game by using exit button in the top right of the game screen.

**3.1.8** **Help**

Player can get all kind of information about the game as an instructive explanation.

This instructive explanation contains the information about:

* + Rules and main purpose of the game.
  + Player controls.
  + Obstacle types.
  + Mines and health explanations

This help document mainly provides Player a chance to learn rules and game controls. When Player learned and used all these information during game, he will have much more fun.

**3.2 Non-Functional requirements**

**3.2.1 Game Performance**

We plan to make Tank War work with high performance. Since there will be sound and some displays such as exploding brick, mine and tank, we want these features not to effect speed of the game. Also, we will try to keep system requirements as low as we can since it is an old fashioned game and it should work in every standard computer in these days.

**3.2.2 User-Friendly Interface**

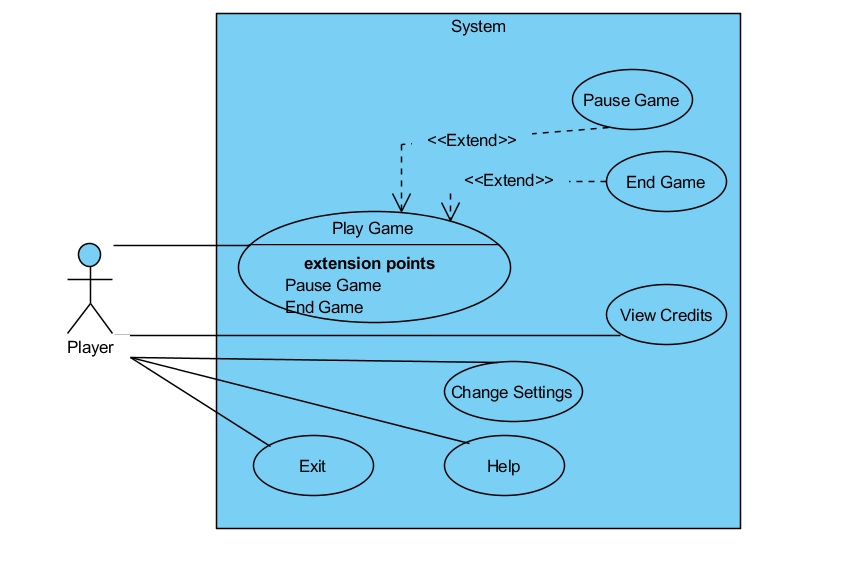
The game will consist of a user-friendly interface which Player will be able to play the game easily. This feature of the game is important to make Player feel comfortable while playing the game. In order to provide easy access to players, our game will have only necessary buttons and option. In addition, we will use special colors which do not make tension in the eyes.

**3.2.3 Extendibility**

In software engineering, reusability and extendibility are two importance concepts. We plan to make Tank War suitable to be extended and re-used for future works.

**4. System Model**

**4.1 Use Case Model**

**

**4.1.1** Play Game

**Participating actors:** Player

**Entry condition:** Player has already opened the game and is on main menu

**Exit condition:**

* One player has killed the other player, OR
* One player has chosen to exit the game using exit in the pause menu.

**Flow of Events:**

1. Player starts the game by pushing start game button.
2. The system constructs the map of the game.
3. One of the players kills the other.
4. The system displays the name of the winner and then the credits of the game.
5. The system goes back to the main menu of the game.
6. Player returns to the main menu.

**4.1.2** View Help

**Participating actors:** Player

**Entry condition:** After Player has opened the program the game is on the start menu.

**Exit condition:** Player returns to the start menu.

**Flow of Events:**

1. Player presses help button.
2. The system displays brief description of the gameplay and the controls in the game.
3. Player pushes back button and returns to the start menu.

**4.1.3** View Credits

**Participating actors:** Player

**Entry condition:** After Player has opened the program the game is on the start menu.

**Exit condition:** Player returns to the start menu.

**Flow of Events:**

1. Player presses view credits.
2. The system displays the credits of the game.
3. Player pushes back button and returns to the start menu.

**4.1.4** Change Settings

**Participating actors:** Player

**Entry condition:** Player has already opened the game and is on main menu

**Exit condition:** Player returns to main menu.

**Main Flow of Events:**

1. Player presses settings button.
2. The system displays the sound setting of the game.
3. Player configures the settings.
4. Player returns to main menu.

**Alternative Flow of Event:**

1. Player does not change any settings. (go to step 4)

**4.1.4** Exit

**Participating actors:** Player

**Entry condition:** Player has already opened the game and is on the start menu.

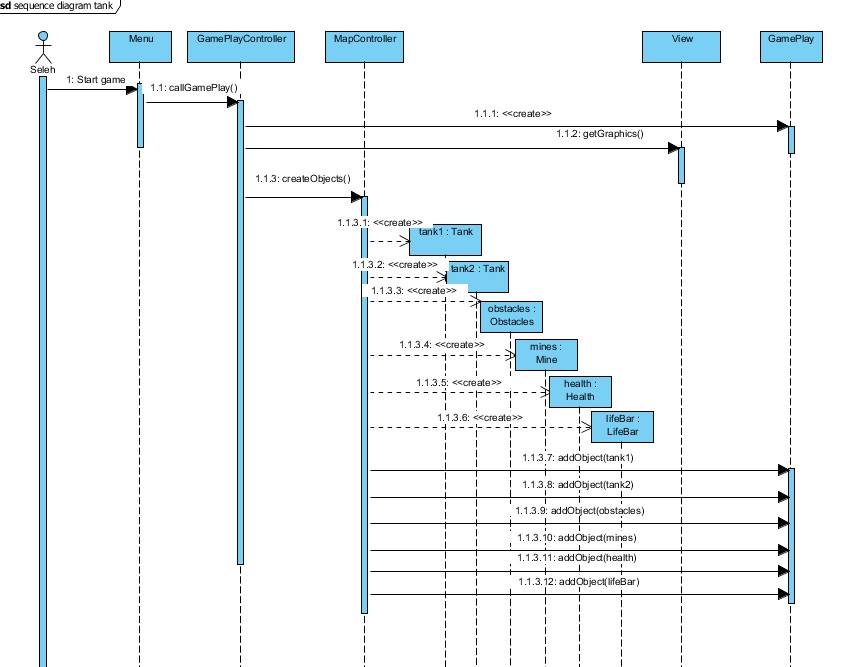
**Exit condition:** Player opens up the game again.

**Main Flow of Events:**

1. Player presses exit button.
2. Program exits

**4.2 Sequence Diagrams**

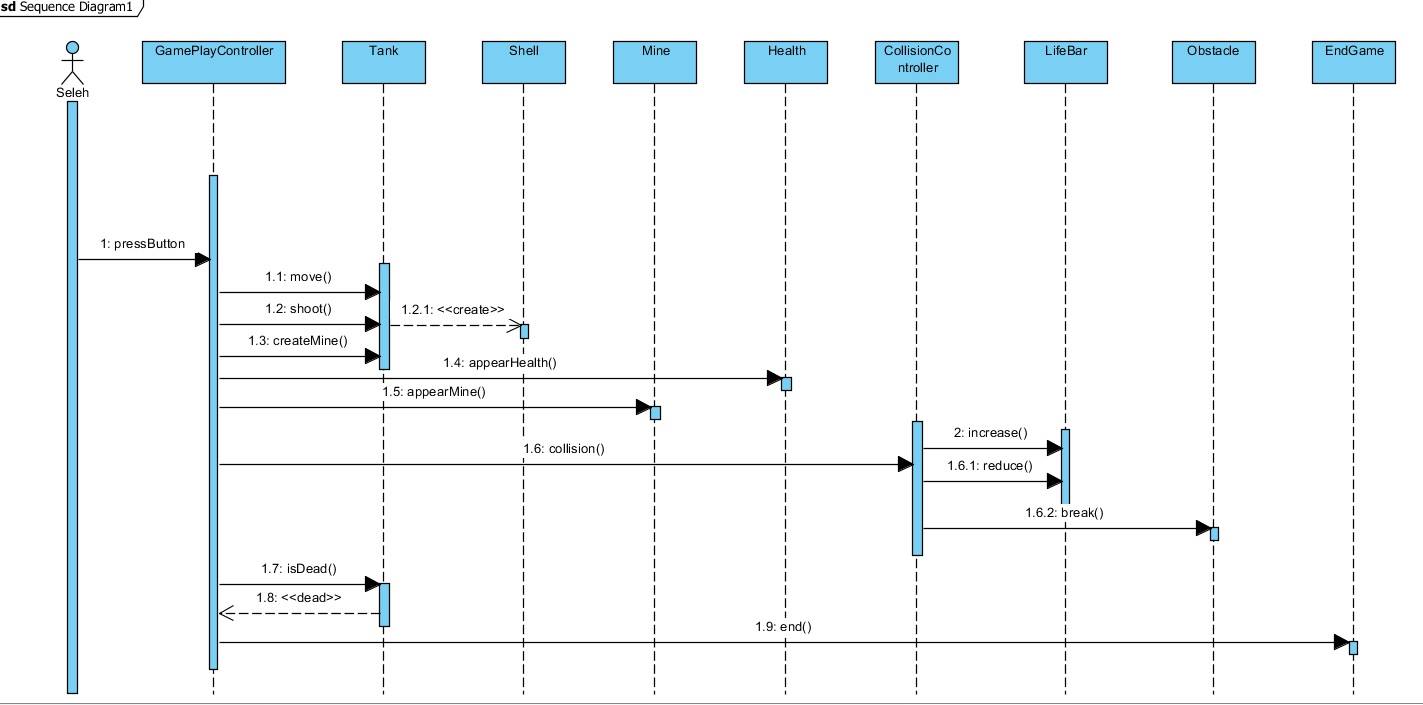
**4.2.1 Game Setup**

****

**Scenario 1:** Seleh launches the program from the desktop and waits for it to open. The menu screen opens and Seleh clicks on the ‘Start Game’ button. The map opens up showing the initial position of the tanks, obstacles, and the landmines.

**Description:** Within the menu, pressing the start button leads to a call of the method callGamePlay () of the GamePlayController. This leads to the following chain of events: GamePlay is created. A message is sent to View to render the graphics. MapController instantiates the objects tanks1, tank2, obstacles, mines, health, and lifebar and adds hem to GamePlay.

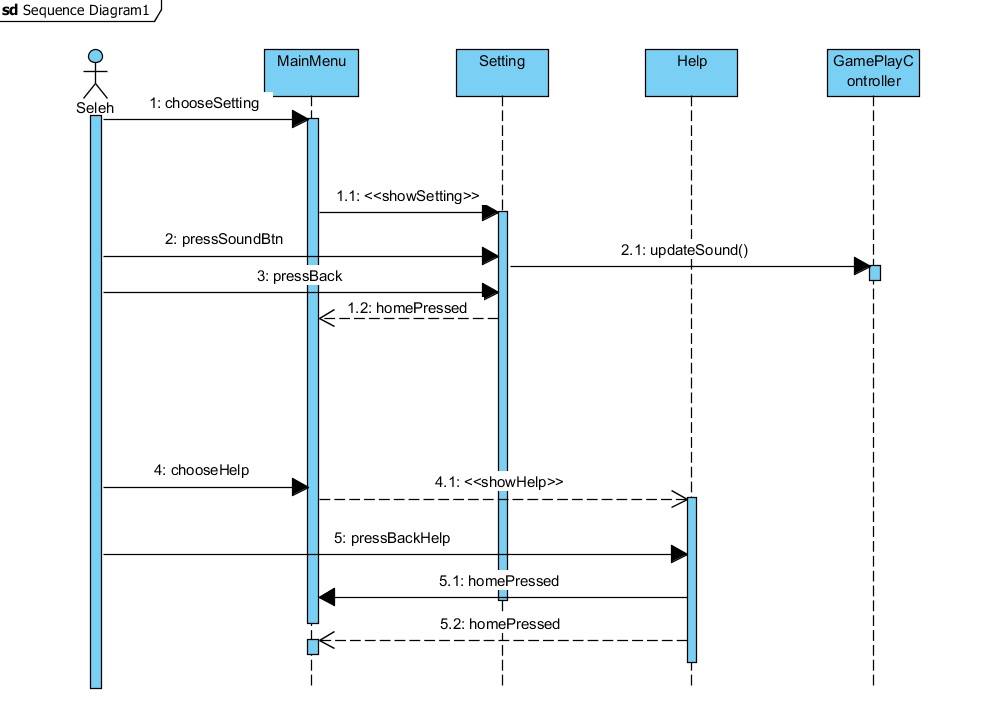
**4.2.2 Game**

****

**Scenario 2: After** Seleh starts the game, he proceeds to play it with another user. He does different activities, such as moving the tank, shooting shells, breaking obstacles etc.

**Description:** The GamePLayController controls the button presses and determines when the game ends. When a button is pressed, the GamePlayController determines which one it was and what action to take based on this. For instance, if direction button is pressed, the GamePlayController finds out with which tank the button is associated and in which direction to move it. Similarly, the CollisionController checks if there is a collision between a shell and a tank or a shell and an obstacle and makes a decision about it.

**4.2.3 Main Menu Options**

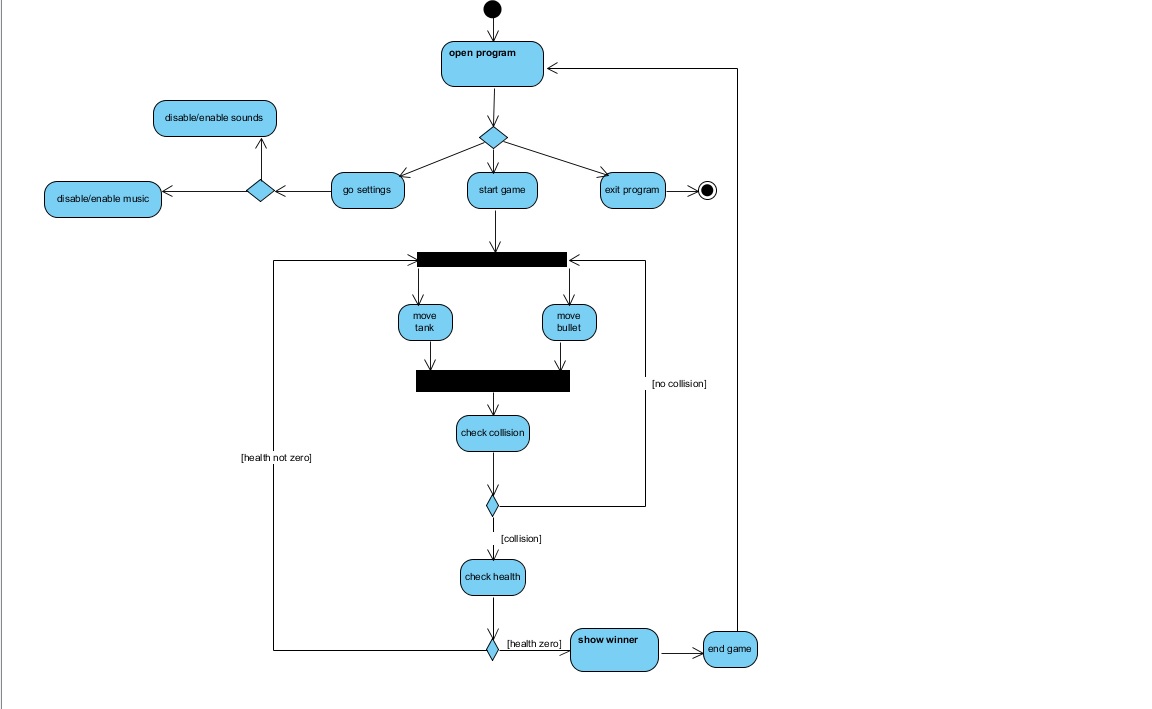
****

**Scenario 3:** Seleh is in the main menu and he wants to change the settings. He clicks on the settings button and a new screen pops up, he toggles the sound button and the sound turns off. Then Seleh presses the Home button and enters the main menu. He presses the Help button and enters a new screen. After reading the help section he again presses Home button and goes back to the main menu.

**Description:** Pressing Settings makes the call showSetting where toggling the Sound button call the updateSound(), this changes sound status from the current to the other one. Pressing Home brings back the main menu. Pressing Help calls showHelp which renders a screen with some text.

**4.2.2 Activity Diagram**

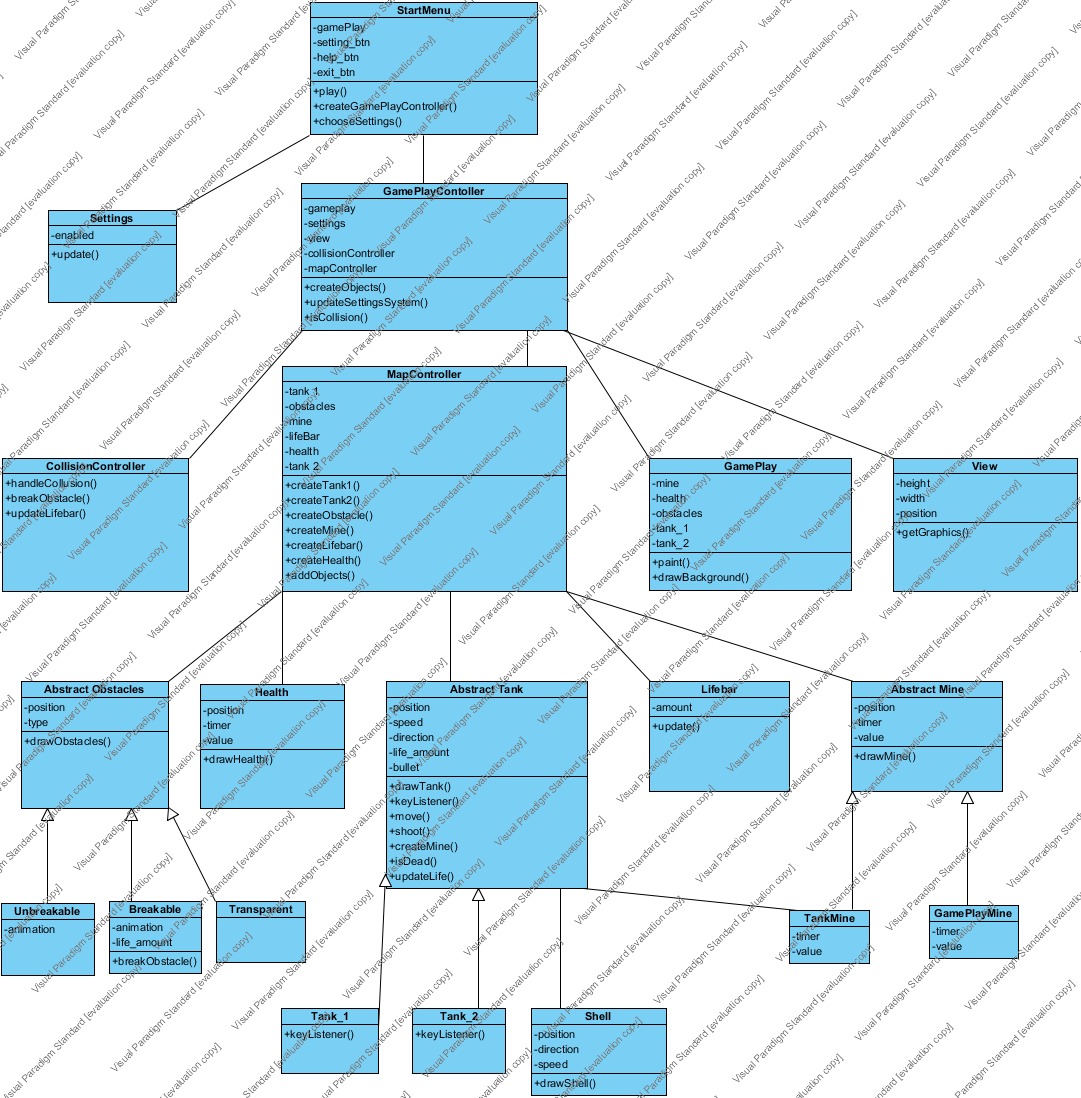
The diagram below displays how system runs the game.

****

When the user selects Start button system sets up the game by instantiating game objects and creating game map. Once The game is ready players can interact with system by pressing specific button for Tank objects movement or shooting shells. When shells are shot system checks if a collision has happened or not. If there is a collision, then system examines each player’s life and decides whether to end the game or continue based on whether any of the two players are dead.

When the user selects setting in the main menu the system goes to settings and based on the user’s option selection disables or enables sounds and music in the game. Also, if in the main menu the user chooses exit option the system terminates the program.

**4.3 Object and Class Model**

****

The object model of the Tank War game is illustrated above. Class Diagram of Tank War consists of 20 classes.

* **Start Menu:**

This class basically displays the Menu for the game. It creates Game Play Controller and Setting objects.

* **Settings:**

Settings have “update” method which updates the music’s and sound’s condition by setting it to ON or OFF.

* **Game Play Controller:**

This class is one of the main classes. It creates all control objects and Game Play playground object. It has methods such as isCollusion which detects the collisions.

* **Map Controller:**

It creates all objects which interact with player such as tank, mines, health and obstacles etc. It also makes all calculations for the movement of the objects.

* **Collision Controller:**

It handles the collusion of tanks, obstacles and mines. It just checks whether one of these objects has a collision.

* **Game Play:**

This class has objects such as tank, mine and obstacles. It uses the View class’s methods such as paint to draw objects.

* **View:**

It is a library for the graphics. It handles the animation of tanks, shells and mines. It has getGraphics method.

* **Life bar:**

This class shows the health of each tank, and it is automatically updated when tank loses his health.

* **Health**

It is an object that GamePlay creates in random periods of time in the game. Tanks controlled by players can pick the power-ups (health objects in the form of a waffle) in order to increase their amount of life.

* **AbstractObstacle**

It is an abstract form of an obstacle which has their mutual attributes and operations. Obstacles have various purposes and there are 3 types of them:

1) Breakable

Tanks can break these obstacles by shooting at them; however the first shell to the obstacle shot will be blocked.

2) Unbreakable

These obstacles cannot be broken and they always block the shells of tanks.

3) Transparent

These obstacles are needed to camouflage the tanks. They go under them and they cannot be broken.

* **AbstractTank**

It is an abstract form of a Tank which has their mutual attributes and operations. It is abstract because the first and the second tanks have different implementation for the same method, keyListener(). Tanks controlled by players and their amount of life is updated when they get hit by a shell or when they pick up a power-up:

1)Tank1

Controlled by w,a,s,d keys to move

2)Tank2

Controlled by arrows to move

* **AbstractMine**

This object can be either created by a GamePlay at random time and a random position or can be set by a tank object. After some time the mine will explode and the object located near them will get their lives updated in other words reduced:

1)TankMine

A Mine set by a Tank

2)GamePlayMine

A mine set by a GamePlay

* **Shell**

An Object created by a Tank once the relevant keys are pressed. When the shell reaches the map limits it disappears, when it encounters the transparent obstacle, it goes through it, for any other object it either destroys it or reduces the amount of life.

**5 User Interface**

**6. Glossary**

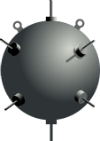
**Tank**

Each Tank Object corresponds to a player. Users can control their tanks’ movement and shoot shells from it. They can also drop limited number of mines. Each Tank object has a life which is decreased if it gets shot. The icon used to represent Tank object is below.

C:\Users\shahrzad\Desktop\my files\backup\Uni\3rd year\summer\whaaat\MyGame\src\tank_up.png

**Mines**

Mines in the game are explosives that if a Tank object hits one of them or if its lifecycle finishes, it would cause damage to Tank objects in a certain area around it. These Objects could be either implanted by Tanks or could appear randomly in the game. They have a limited and short lifecycle. Mine objects are displayed with the icon below in the game.



**Shells**

Shells are objects in the game that each Tank can shoot and if they get hit by a Shell their life would decrease. Shells move in the direction that the tank is facing at the time of shooting. One shot they would move until they hit another object like a Tank or an Obstacle.

C:\Users\shahrzad\Desktop\my files\backup\Uni\3rd year\summer\whaaat\MyGame\src\shell_right.png C:\Users\shahrzad\Desktop\my files\backup\Uni\3rd year\summer\whaaat\MyGame\src\shell_right.png C:\Users\shahrzad\Desktop\my files\backup\Uni\3rd year\summer\whaaat\MyGame\src\shell_right.png C:\Users\shahrzad\Desktop\my files\backup\Uni\3rd year\summer\whaaat\MyGame\src\shell_right.png C:\Users\shahrzad\Desktop\my files\backup\Uni\3rd year\summer\whaaat\MyGame\src\shell_right.png C:\Users\shahrzad\Desktop\my files\backup\Uni\3rd year\summer\whaaat\MyGame\src\shell_right.png

**LifeBar**

LifeBars are objects that keep track of each player’s life. They are represented like the picture below in the game.

C:\Users\shahrzad\Desktop\my files\backup\Uni\3rd year\summer\whaaat\lifeBar.png

**Obstacles**

Obstacles are walls in the game that prevent players from passing through. There are two types of Obstacles, breakable and non-breakable. Breakable Obstacles break when they are shot with a shell. Non-breakable Obstacles are not affected by anything. The two types are shown below.

 breakable

 non-breakable

**Health**

The only power up in the game is the Health object which increases the life of a player if the player eats it. These objects appear at random times during the game and have a limited and short lifecycle.

