Poke Battle

Game Functional Requirements

Name: Anders Dahl

Email: dahl.a@husky.neu.edu

**Poke Battle Game:**

Poke Battle 1.0 is a 64-bit Windows 8 based console game. The user, a Pokémon trainer, will battle other computer trainers in a 6v6 Pokémon match. The user will pick his or her six Pokémon at the very beginning if time permits (otherwise it will be set for you). After defeating a trainer, a new challenger will enter the battle field and the old one will leave. Eventually, Ash, the final contestant will be faced. If defeated, the game is over and you have won. There are three types of Pokémon – grass, water, and fire. Each type is weak and super effective against another; thus, choosing what Pokémon to battle with is important. The battle will also consists of three weather patterns that give Pokémon of the right type a 10% increase in damage output. Rain Dance boosts water types, Sunny Day boosts fire types, and Sandstorm boosts grass types. If all of a player’s Pokémon have fainted he or she has lost.

The rules for battling are similar to Pokémon on the Gameboy.

1. The user chooses between four options – moves, items, switch, and run (lose).
2. There are four moves for each Pokémon. The first two are normal attacks. The third is an attack specific to its type; therefore, it can be super effective, normal, or weak depending on the type of the other Pokémon. The fourth move changes the weather to the type of that Pokémon.
3. Items consists of potions. There are two different types. Potions restore 20 health points and super potions restore 50. Time permitting, there might be some potion spawns in the battle room that the player can pick up.
4. Switch switches Pokémon. It counts towards your turn, and it prevents you from choosing a move for that round.
5. If all of the player’s Pokémon have been defeated then he or she has lost.

The user will be able to walk around in the battle room. Once ready to battle, he or she will simply walk up to the trainer and press enter. At least 5 challangers will be faced including Ash.

**Contents:**

1. Requirements Analysis

1.1 Player Controls

1.2 Art/Animations

1.3 Game Scoring/Rounds

2. Physical Architecture Analysis

2.1 Hardware Requirements

2.2 System Requirements

2.2 Software Dependencies

3. Programming Analysis

4. Component Analysis

4.1 Pokémon Class

4.2 Player Class

4.3 Game Class

4.4 Item Class

5. Evaluation

6. Milestones

# **Requirements Analysis**

The user audience will mostly consist of Pokémon fans. The game will have 2D graphics similar to the ones seen in the original Pokémon series for the Gameboy. Even though Poke Battle is a Windows 8 console game, the goal is to tailor to some of these original fans.

## **Player Controls**

Outside battle –

The player will move up, down, left, and right using the arrow keys. He or she will pick up an item or challenge trainers using enter. The sprite of the user will have to be close enough to the item or trainer in order to perform the action.

In battle –

The player will use keys 1-4 to pick between moves, switch, items, and run. If moves are selected 1-4 will be used to choose a move, and if switch is selected 1-6 will be used to pick a Pokémon.

## **Art/Animations**

Not in battle –

* The battle room is walkable by the user
* Animation of old trainer leaving and a new trainer entering

In battle –

* Pokémon attacking animation
* Pokémon defending animation
* Pokémon coming into battle animation

## **Game Scoring/Rounds**

* Keep track of the amount of trainers defeated
* Keep track of the health of the Pokémon

# **Physical Architecture Analysis**

## **Hardware Requirements**

The game will run on any relatively modern computer

## **System Requirements**

The game will be developed on 64-bit Windows 8. A mouse, monitor, and keyboard are required to play the game.

## **Software Dependencies**

The game will come packaged with SMFL 2.1.

http://www.sfml-dev.org/

# **Programming Analysis**

Anders Dahl will be the sole developer and tester of this game.

# **Component Analysis**

## **Pokémon**

4.1.1 Description and Priority

Pokémon are the fictional characters in the game. Implementing the Pokémon is of high priority.

4.1.2 Stimulus/Response Sequences

The Pokémon will be able to attack in battles. Pokémon are weak, neutral, or super effective towards others. In general a super effective attack will deal 2x damage, and a weak attack will deal 1/2x damage. The weather also has an effect on how much damage is dealt when a Pokémon attacks another. If it is a rain dance water types will deal 10% more damage, if it is a sandstorm fire types will deal 10% more damage, and if it is a sunny day then grass types will deal 10% more damage. Items can be used on Pokémon to restore their health. Each Pokémon will have a choice of four moves one which will be used to change the weather to its benefit. They start off with a health of 100.

4.1.3 Functional Requirements

Exceptions will be used to handle errors, and each error will have its own specific exception message so that the developer can easily find out what is going wrong. If the user enters invalid inputs then nothing will happen.

REQ- Attack:

Pokémon will be able to attack using one of four different moves chosen by the player through keyboard keys 1, 2, 3, and 4.

REQ-2- Attack Animation:

Pokémon will need to move in the direction of the other Pokémon when attacking. The other Pokémon will also have to move back to characterize that it is defending itself.

## **Player**

4.2.1 Description and Priority

A player represents a trainer in the game. Implementing the Player is of high priority.

4.2.2 Stimulus/Response Sequences

The player will be able to choose the move of the Pokémon that is currently out. It will also be able to use items on the Pokémon to restore health, and he or she can switch the Pokémon that is currently out. If the trainer runs from a battle it will result in a loss. The amount of wins will be kept track of for the user.

4.2.3 Functional Requirements

Exceptions will be used to handle errors, and each error will have its own specific exception message so that the developer can easily find out what is going wrong. If the user enters invalid inputs then nothing will happen.

REQ-1- User input:

The trainers will be able to choose move, items, switch, and run. If switch is called, keys 1-6 will be used to choose the user input accordingly, while move will only use keys 1-4.

REQ-2- Walking and Challenging:

The trainer will be able to walk around the battlefield and challenge the trainer currently out.

## **Game**

4.3.1 Description and Priority

Implementing the Game is of high priority.

4.3.2 Stimulus/Response Sequences

The Game class will be responsible for handling the playing of the game in general. It will have the ability to start a battle between two trainers, and it will draw everything to the window.

4.3.3 Functional Requirements

Exceptions will be used to handle errors, and each error will have its own specific exception message so that the developer can easily find out what is going wrong.

REQ- 1-Play Trainers

The Game class is responsible for battling two players and drawing all of the components to the screen that goes with it.

REQ-2- Battlefield

The game is responsible for drawing the user walking around on the battle field, and if the user decides to challenge the trainer then it should take care of handling that.

## **Item**

4.4.1 Description and Priority

Implementing the Item class is of moderate priority.

4.1.2 Stimulus/Response Sequences

Items are responsible for healing Pokémon.

4.1.3 Functional Requirements

REQ- 1-Heal Amount

It needs to keep track of how much it heals.

REQ-2- Potion Name

The items class needs to keep track of the two different types of items – potions and super potions.

# **Evaluation**

* Testing will consists of test playing the game in order to find bugs. The user will try strange things such as walking into a wall repeatedly to make sure one cannot leave the screen, for example.
* Exceptions and print statements will be used for error handling.
* If time permits, some sort of unit testing will be implemented. Time would have to be spent to become familiar with the process, however.

# **Milestones**

November 16th

Become familiar with SFML 2.1. Player, Pokémon, and Games class set up to a limited amount. Player should be able to walk around on the battlefield. Challengers should leave if pressed enter on, and a new challenger should enter. Ash should eventually come, and if pressed enter on the game should be over.

November 30th

The challengers should be setup with their 6 Pokémon of 3 types. Many of them will be repeats. For example, I might just have Charmander, Bulbasaur, and Squirtle represent the three types. The Player, Pokémon, and Games class should be completely set up.

December 7th

The minimum goal is to have a working game with at least 5 challengers. The battling part should be working. More ideas might be implemented if time permits.