Projeto Battle Element

Especificação de Requisitos de Software

Primeira especificação de requisitos

03/04/2023

Edivaldo Ferreira de Souza Júnior

Gabriel Avila

Guilherme Cavalieri

# 1. Introdução

## 1.1 Objetivo

Desenvolvimento de um programa distribuído que suporte a criação de partidas e jogatina entre dois jogadores em computadores distintos.

## 1.2 Referencias

Olhar o anexo

## 1.3 Explicação do jogo

## Battle Element é um jogo de estratégia baseado em turnos, no qual dois jogadores se enfrentam utilizando equipes de três personagens com diferentes elementos e raças. Cada jogador cria sua própria equipe, definindo a raça e o elemento de cada personagem.

# A combinação de raça e elemento tem um impacto direto nos status de dano e defesa de cada personagem. Esses status determinam a eficácia dos ataques dos personagens contra os adversários, bem como a resistência aos ataques do adversário.

# Durante a batalha, cada jogador tem a sua vez de jogar e pode escolher entre diferentes movimentos, como ataque normal, ataque elemental e aumento de defesa. O ataque normal causa dano padrão, enquanto o ataque elemental tem um multiplicador de dano baseado no elemento do personagem que está usando e no elemento do adversário. A habilidade de aumentar a defesa aumenta temporariamente a resistência do personagem aos ataques adversários.

# O objetivo do jogo é derrotar os três personagens do adversário antes que seus personagens sejam derrotados. Para vencer, o jogador deve utilizar suas habilidades estratégicas para escolher os melhores movimentos e personagens para cada situação.

Ver as regras e definições no apêndice.

# 2. Visão geral

**2.1 Arquitetura do programa**

Cliente-servidor distribuído.

## 2.2 Premissas de desenvolvimento

* O programa deve ser implementado em Python
* O programa deve usar DOG como suporte para execução distribuída
* A interface gráfica deve ser implementada usando a biblioteca Tkinter
* O programa deve ser modelado com base nas especificações da linguagem UML 2
* Para a modelagem, será utilizado o software Visual Paradigm

# 3. Requisitos de Software

## 3.1 Requisitos Funcionais

**RF01 – Inicialização do Jogo:** O jogo deve inicializar corretamente, preparando todos os componentes e gerenciadores necessários, incluindo interface do jogador, comunicação, gerenciador de servidor, gerenciador de janelas e gerenciador de gráficos. Além disso, o jogo deve solicitar o nome do jogador e tentar estabelecer conexão com o servidor até que seja bem-sucedida. Após a conexão ser estabelecida, o jogo deve abrir o menu principal.

**RF02 - Iniciar Partida:** O jogo deve permitir que o jogador inicie uma partida ao solicitar o início da mesma. Em seguida, o jogo deve procurar por um jogador disponível através do servidor e avaliar a resposta do servidor até obter uma resposta positiva. Após encontrar um jogador disponível, o jogo deve abrir a janela de criação de equipe.

**RF03 - Receber Início de Partida:** O jogo deve ser capaz de receber uma determinação para iniciar uma partida originada por outro jogador através do servidor. Após receber essa determinação, o jogo deve abrir a janela de criação de equipe para que os jogadores possam montar suas equipes antes do início da partida.

**RF04 - Montar e Enviar Equipe:** O jogo deve permitir que o jogador monte sua equipe configurando as características do time, selecionando e preparando os heróis e visualizando os atributos do time na tela. Após a confirmação da seleção, a equipe deve ser enviada ao servidor e o jogo deve preparar a batalha caso já tenha recebido a equipe adversária.

**RF05 - Receber Equipe Adversária:** O jogo deve ser capaz de receber a equipe adversária do servidor, criar e armazenar essa equipe. Caso o jogador já tenha criado sua própria equipe, o jogo deve preparar a batalha.

**RF06 - Preparar Batalha:** O jogo deve criar a lista de turnos, distribuir os heróis na tela e mostrar os status dos heróis antes de preparar o primeiro turno.

**RF07 - Preparar Turno:** O jogo deve atualizar a lista de turnos, definir o papel do jogador no turno (atacante ou defensor), sinalizar esse papel na tela, atualizar o herói de destaque na tela e atualizar o tempo de recarga das habilidades. Se o jogador for o atacante, o jogo deve definir um ataque e alvo padrão e desbloquear as opções de jogador para que ele possa selecionar um ataque e alvo diferente, caso deseje.

**RF08 - Enviar Ataque:** O jogo deve permitir que o jogador selecione um alvo e ataque, processar o ataque e avaliar se há uma revanche em caso de vitória. Em seguida, o ataque deve ser enviado ao oponente através do servidor e o jogo deve preparar o próximo turno caso a partida não tenha terminado.

**RF09 - Selecionar Alvo e Ataque:** O jogador deve ser capaz de escolher um alvo entre os heróis inimigos vivos e escolher um ataque entre os que podem ser usados de acordo com as regras do jogo. A seleção deve ser confirmada pelo jogador.

**RF10 - Processar Ataque:** O jogo deve aplicar o dano ao alvo selecionado, atualizar os status dos heróis envolvidos na tela e, caso o alvo tenha sido eliminado pelo ataque, removê-lo da lista de turnos e da tela.

**RF11 - Avaliar Revanche:** Após uma vitória, o jogo deve abrir a tela de vitória e perguntar ao jogador vencedor sobre sua intenção de realizar uma revanche. Caso o jogador negue a intenção de revanche, o jogo deve retornar ao menu principal.

**RF12 - Receber Ataque:** O jogo deve ser capaz de receber informações do ataque do oponente através do servidor, processar o ataque e, caso o jogador não tenha perdido a partida, preparar o próximo turno. Se o jogador tiver perdido, o jogo deve abrir a tela de derrota e perguntar ao jogador sobre a intenção de revanche caso o oponente tenha solicitado uma. Se o jogador recusar o pedido de revanche, o jogo deve retornar ao menu principal, caso contrário, o jogo deve enviar a resposta da proposta de revanche e, se aceita, abrir a janela de criação de equipe. Se a revanche for recusada, o jogo deve retornar ao menu principal.

**RF13 - Receber Confirmação de Revanche:** Após a aceitação da proposta de revanche, o jogo deve limpar os dados das equipes e abrir a tela de criação de heróis para que os jogadores possam montar suas equipes novamente.

**RF14 - Receber Notificação de Desistência:** O jogo deve ser capaz de receber uma mensagem do servidor sobre a desistência do oponente, mostrar um aviso na tela informando o abandono e, em seguida, abrir o menu principal. Além disso, o jogo deve realizar a limpeza dos dados pós-partida.

## 3.2 Requisitos Não-funcionais

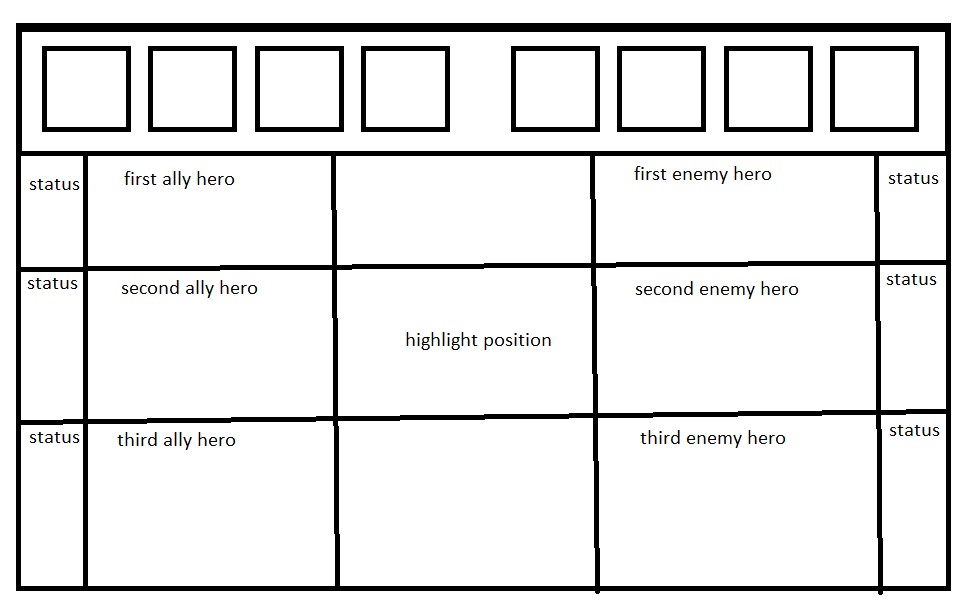
**RNF01 -** **Utilização da linguagem de programação Python:** A linguagem de programação utilizada no projeto deve ser Python.

**RNF02 -** **Implementação da interface gráfica com a biblioteca Tkinter do Python:** A interface gráfica deve ser implementada usando a biblioteca Tkinter do Python

**RNF03 -** **Utilização da tecnologia DOG para execução distribuída:** A tecnologia DOG deve ser usada para permitir o suporte à execução distribuída

**RNF04 -** **Compartilhamento da interface gráfica entre os jogadores durante todo o uso do programa:** A interface gráfica deve ser compartilhada entre os dois jogadores com as mesmas informações durante todo o uso do programa.

**RNF5 –** **Interface do programa**

****

## 3.3 Regras de negócio

**RN01 - Criação de Heróis:** Ao criar um herói, o jogador deve escolher entre cinco raças diferentes, cada uma com atributos base distintos (conforme especificado no anexo de status). Além disso, o jogador deve escolher entre cinco classes (ofícios), cada uma com multiplicadores de atributos e uma habilidade distinta (conforme especificado no anexo de status e habilidades). Por fim, o jogador deve escolher entre cinco elementos, cada um com multiplicadores de atributos e uma habilidade distinta (conforme especificado no anexo de status e habilidades).

**RN2 - Atributos de Personagens:** Os personagens possuem quatro atributos principais: pontos de vida, pontos de mana, dano de ataque e agilidade.

**RN3 - Eliminação de Herói:** Um herói é eliminado quando seus pontos de vida atingem 0 ou menos. Heróis eliminados não podem atacar nem ser atacados.

**RN4 - Uso de Habilidades:** O uso de habilidades pode ter um custo em mana e um tempo de recarga, ou seja, a habilidade precisa de um determinado número de turnos (x) para poder ser usada novamente.

**RN5 - Ordem de Turnos:** A ordem dos turnos é definida pela agilidade dos heróis, de modo que heróis com maior agilidade atacam mais rápido. O jogador controla o herói da vez em seu turno, caso seja do seu time.

**RN6 - Atributos de Habilidades:** As habilidades possuem um custo de mana e tempos de recarga pré-definidos, conforme especificado no anexo de habilidades. Além disso, o dano de ataque das habilidades é calculado como um valor pré-definido multiplicado pelo dano de ataque do herói. As habilidades herdadas pela escolha do elemento do herói têm um multiplicador extra, que varia dependendo da combinação do elemento do herói com o elemento do alvo, conforme especificado no anexo de ataque elemental.

**RN7 - Condição de vitória:** eliminar os três heróis adversários.

# 4. Apêndice

Regras do jogo:

1 O jogo consiste em partidas individuais entre dois jogadores.

2 Cada jogador monta um time com três heróis que atuam independentemente.

3 Cada herói é montado escolhendo uma raça (que possui atributos base distintos), uma classe e um elemento que multiplicam os atributos.

As raças e seus atributos base são:

\* Humano: vida: 3, agilidade: 3, dano: 3, mana: 3;

\* Elfo: vida: 2, agilidade: 4, dano: 4, mana: 4;

\* Anão: vida: 5, agilidade: 1, dano: 2, mana: 2;

\* Orc: vida: 4, agilidade: 2, dano: 5, mana: 1;

\* Gnomo: vida: 1, agilidade: 5, dano: 1, mana: 5.

As classes e seus multiplicadores são:

\* Guerreiro: vida: 5x, agilidade: 2x, dano: 1x, mana: 2x;

\* Assassino: vida: 3x, agilidade: 3x, dano: 4x, mana: 4x;

\* Lutador: vida: 4x, agilidade: 4x, dano: 3x, mana: 1x;

\* Arcanista: vida: 1x, agilidade: 1x, dano: 5x, mana: 5x;

\* Atirador: vida: 2x, agilidade: 5x, dano: 2x, mana: 3x;

Os Elementos e seus multiplicadores:

\* Fogo: vida: 2x, agilidade: 3x, dano: 5x, mana: 2x;

\* Água: vida: 3x, agilidade: 4x, dano: 2x, mana: 4x;

\* Terra: vida: 5x, agilidade: 1x, dano: 3x, mana: 3x;

\* Raio: vida: 1x, agilidade: 5x, dano: 4x, mana: 1x;

\* Planta: vida: 4x, agilidade: 2x, dano: 1x, mana: 5x;

4 Cada herói possui atributos, incluindo vida, agilidade, dano, mana e elemento.

5 Cada partida é realizada em turnos, a ordem de jogada dos heróis é determinada pela agilidade de cada um.

6 Durante o turno de um herói, o jogador responsável por controlá-lo deve escolher um alvo do time adversário e uma habilidade para atacar.

7 Os heróis possuem três tipos de habilidades: básica, especial e elemental.

8 As habilidades podem causar dano ou bloquear dano, além de poderem gastar mana e terem um tempo de recarga.

9 A habilidade elemental causa dano multiplicado de acordo com o elemento da dela em relação ao elemento do herói alvo, sendo:

\* fogo: 2x na planta, 0.5x na água, 1.4x na terra, 0.7x no raio;

\* planta: 2x na terra, 0.5x no fogo, 1.4x no raio, 0.7x na agua;

\* terra: 2x no raio, 0.5x na planta, 1.4x no agua, 0.7x na fogo;

\* raio: 2x na agua, 0.5x na terra, 1.4x na fogo, 0.7x no planta;

\* agua: 2x no fogo, 0.5x no raio, 1.4x na planta, 0.7x na terra.

10 O vencedor da partida é o jogador que conseguir reduzir a vida dos três heróis adversários a zero.

# Anexo

Links de jogos de referências:

<https://pt.wikipedia.org/wiki/Pok%C3%A9mon_FireRed_e_LeafGreen>

<https://pt.wikipedia.org/wiki/Final_Fantasy_(jogo_eletr%C3%B4nico)>

<https://mutants.kobojo.com/>