UNIVERSIDADE FEDERAL DE UBERLÂNDIA

ARIANE SANTOS BORGES
KAREN CATIGUÁ JUNQUEIRA
MATHEUS PRADO PRANDINI
NICOLAS GONÇALVES PEIXOTO
RAPHAEL LUCA DE CASTELLA E BERTHOLUCCI

JOGO MARCOS PRIME – “O LEGADO”

UBERLÂNDIA
2016

1. DESCRIÇÃO GERAL DO PROJETO

O objetivo deste relatório é descrever sobre a implementação do Jogo Marcos Prime desenvolvido para a disciplina de Programação Orientada a Objetos 2. O jogo foi desenvolvido na linguagem Java utilizando o ambiente virtual NETBEANS e na parte gráfica, desenvolvido em Pixel Art.

Além disso, será discutido sobre os padrões de projeto utilizados para a implementação do jogo, sobre o desenvolvimento em camadas usando o padrão arquitetural MVC (Model View Control) e sobre a interface gráfica.

Por fim, os diagramas de classes detalhando os padrões usados serão anexados e um tópico sobre o modo de jogar do Jogo Marcos Prime.

2. DESCRIÇÃO GERAL DO JOGO

O objetivo principal do jogo é o personagem Marcos que evolui ao concluir cada fase. Pelo qual, assim que o personagem conclui as fases sem morrer, este enfrentará o chefão específico de determinada fase. A vida do personagem e dos inimigos serão representadas por uma barra de vida (em termos de código, por um inteiro). Cada vez que o personagem ou o inimigo é atacado, ele perde vidas e essa barra de vida diminui. Neste projeto, não foi implementado um método de defesa, pois o objetivo é atacar e desviar.

Além do Modo Arcade, foram adicionados mais dois modos no jogo: Modo Survival e Modo Multiplayer, que serão discutidos em breve.
Apresentamos os seguintes requisitos:

Funcionais

- Movimentos dos personagens e inimigos, incluindo desviar de ataques.
- Ataques dos personagens e inimigos.
- Barra de vida para o personagem e inimigos.
- Quatro fases compostas por vários obstáculos e inimigos, além de um chefão final.

- Fases:
    - Bar
    - Cidade
    - Lua
    - Espaço Sideral

- Chefões:
    - Bar -> Saad
    - Cidade -> Monstro
    - Lua -> Batman
    - Espaço Sideral -> Todos anteriores

- Personagens:
    - Bar -> Marquin
    - Cidade -> Marcos Prime
    - Lua -> Superman
    - Espaço Sideral -> Zordon

- Dificuldades:
    - Fácil
    - Médio
    - Difícil

Cada fase é composta por três níveis de dificuldade (a escolha do jogador), mudando a quantidade de inimigos, a vida dos mesmos e a força do chefão.

- Obstáculos:
    - Bar: Mesas e Cadeiras;
    - Cidade: Carros e Árvores;
    - Lua: Buracos;
    - Espaço Sideral: Espaçonaves e Planetas.

- Ataques (Armas):
    - Marquin: Garrafa de Cerveja;
    - Marcos Prime: Celular;
    - Superman: Raio Laser;
    - Zordon: Raio Laser.

- Ataque Inimigos:
    - Soco.

- Poderes Inimigos (Ataque Inimigos Decorado):
    - Inimigos menores: Somente socos;
    - Saad: Tele transporte;
    - Monstro: SuperForça;

- Batman: Boomerang.

Não funcionais:

- Plataforma (2D);
- Linguagem: Protótipo em Java;
- Ambiente Virtual: Netbeans (Java).

3. DESENVOLVIMENTO DAS FASES

Cada fase pode ser jogada em três níveis de dificuldade: Fácil, Médio e Difícil. As únicas mudanças de uma dificuldade para outra são quanto ao número de inimigos que o personagem irá enfrentar antes do chefão, a quantidade de vida e o poder do ataque dos chefões, que custarão mais vida para o personagem. Segue abaixo a descrição de cada fase:

- Fase 1:
  - Ambiente -> Bar;
  - Personagem -> Marquin;
  - Chefão -> Saad;
  - Obstáculos -> Mesas e cadeiras;
  - Ataque do personagem -> Atirar Garrafas de cerveja;
  - Ataque dos inimigos menores -> Socos;
  - Ataque do chefão -> Tele transporte.

- Fase 2:
  - Ambiente -> Cidade;
  - Personagem -> Marcos Prime;
  - Chefão -> Monstro;
  - Obstáculos -> Carros e Árvores;
  - Ataque do personagem -> Atirar celulares;
  - Ataque dos inimigos menores -> Socos;
  - Ataque do chefão -> SuperForça;

- Fase 3:
  - Ambiente -> Lua;
  - Personagem -> Superman;
  - Chefão -> Batman;
  - Obstáculos -> Buracos;
  - Ataque dos inimigos menores -> Socos;
  - Ataque do chefão -> Boomerangue.

- Fase 4:
  - Ambiente -> Espaço Sideral;
  - Personagem -> Zordon;

- Chefão -> Todos os chefões das fases anteriores;
- Obstáculos -> Espaçonaves e Planetas;

4. DESENVOLVIMENTO DOS PADRÕES DE PROJETO

Visto que um dos objetivos dos Padrões de Projeto é garantir a extensibilidade e a reusabilidade do sistema desenvolvido, o Jogo Marcos Prime pôde usufruir dessas qualidades. Ao implementar todas as classes e desenvolver o jogo, com os Padrões de Projeto, foi perceptível o quanto melhorou a transparência das classes na qualidade de Jogo e o quanto contribuiu para manutenção do mesmo. Portanto, como previsto, para a realização do Jogo Marcos Prime foi indispensável o auxílio dos Padrões de Projeto vistos em sala de aula.

Os Padrões de Projeto utilizados foram:

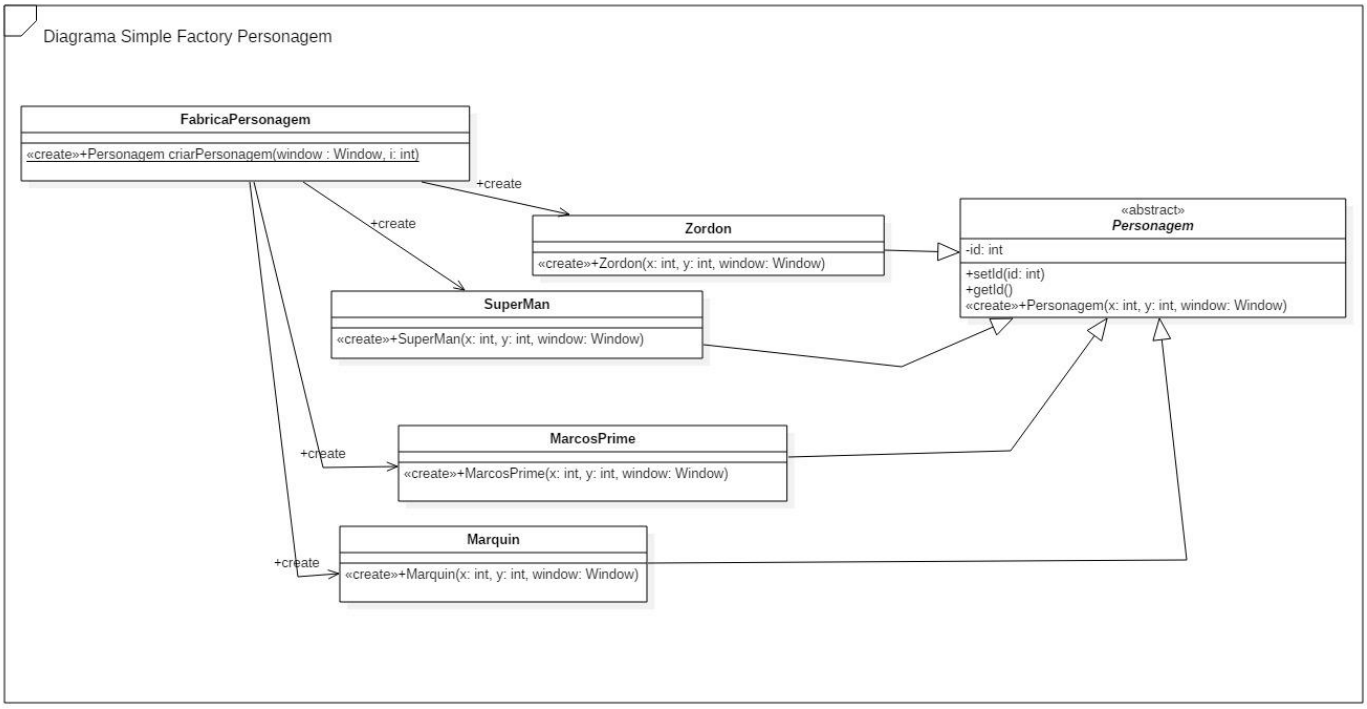
- Padrão Simple Factory (Personagem, Inimigo e Fase);
- Padrão Observer;
- Padrão Decorator;
- Padrão Abstract Factory e Singleton;
- Padrão State.

Para cada um dos Padrões de Projeto implementado no Jogo, foram desenvolvidos seus respectivos diagramas de classes, que serão vistas abaixo.

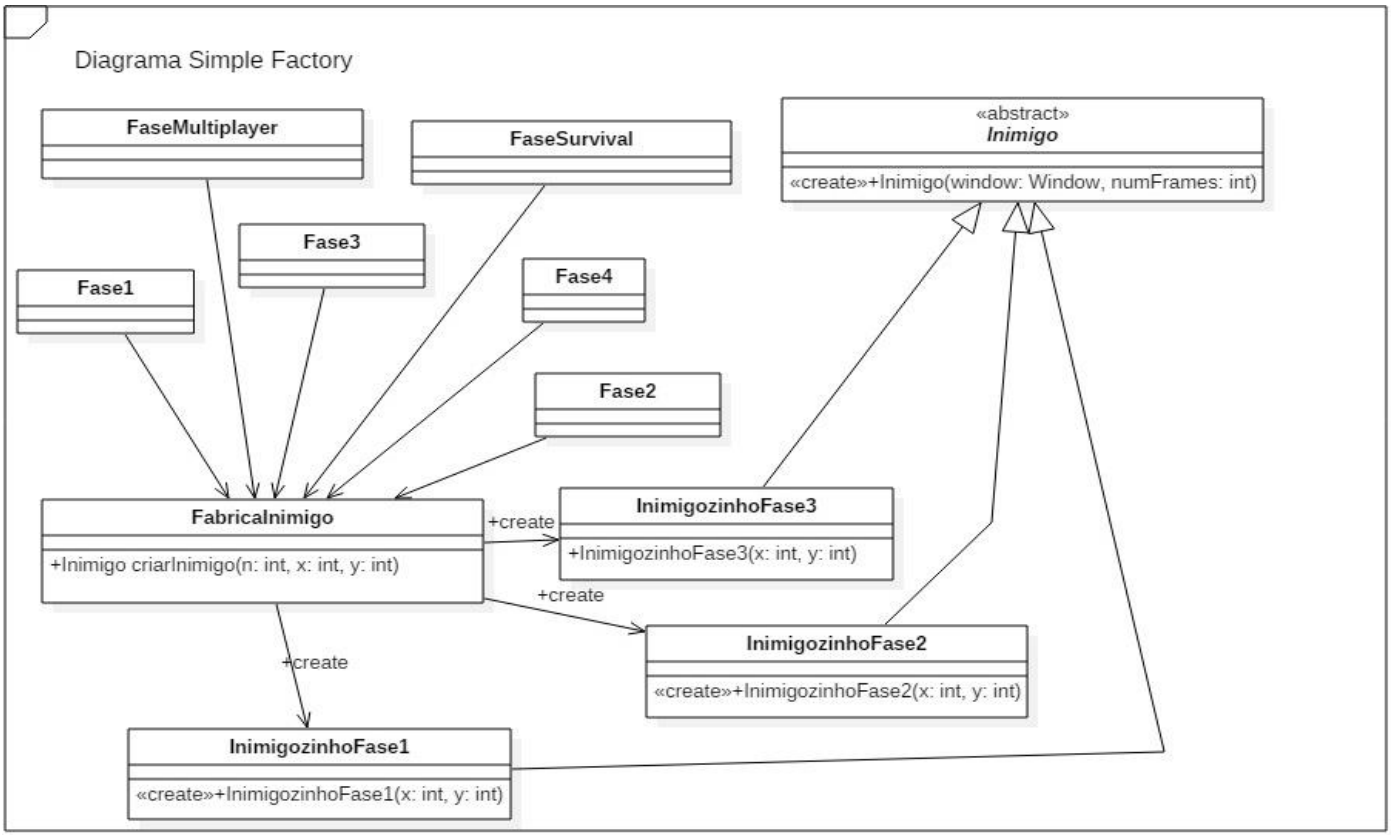
- Padrão Simple Factory:

Para a implementação do Jogo Marcos Prime, foi utilizado o Padrão Simple Factory para o Personagem, para o Inimigo e para a Fase. Esse padrão possibilita que a instanciação de novos objetos fique separada do seu uso. Portanto, no Jogo, esse padrão permite que a Classe FabricaPersonagem crie novos personagens que terão métodos diferentes entre si. Dessa forma, esses novos personagens serão requisitados pela Classe Personagem para serem usados no jogo. Já a Classe FabricaInimigo, cria novos inimigos para serem requisitados pela Classe Inimigo e serem usados no jogo. Além disso, as fases terão como Classe FabricaInimigo um meio de utilizarem os novos inimigos em cada uma. Por fim, a Classe FabricaFase possui quatro métodos: gerarFases, gerarFasesChefao, gerarFaseSurvival e gerarFaseMultiplayer. O primeiro citado é responsável por gerar as fases Fase1, Fase2 e Fase3 que estão no modo história. O segundo é responsável por gerar as fases FaseChefao1, FaseChefao2 e FaseChefao3, também usadas no modo história. Estes dois métodos são utilizados pelas fábricas de dificuldade. O terceiro é responsável por gerar uma fase aleatória para o Modo Survival. E o último é responsável por gerar uma fase aleatória para o Modo Multiplayer. Estes dois métodos são utilizados na classe Jogo.

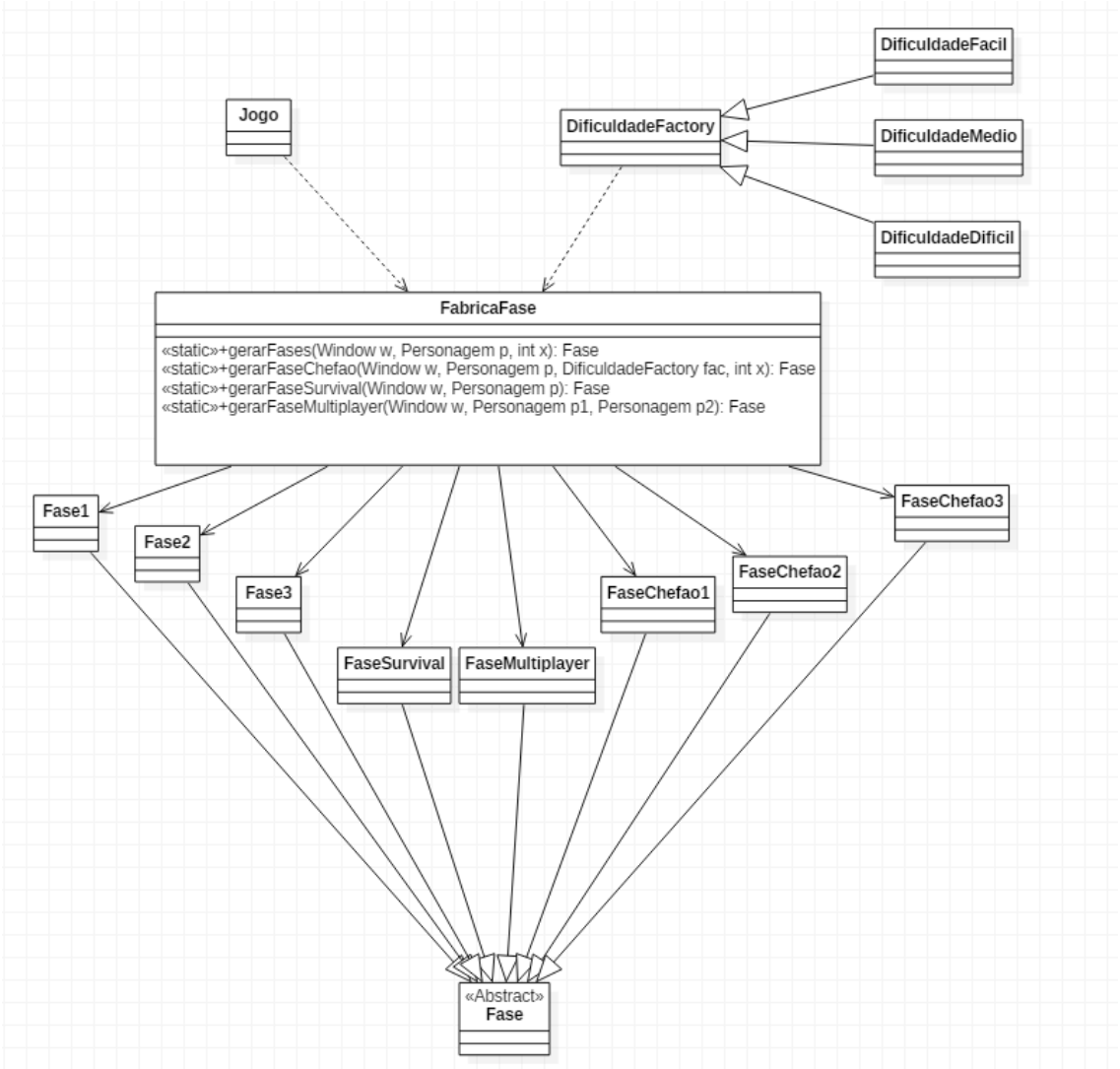
- Simple Factory Personagem:



- Simple Factory Inimigo:

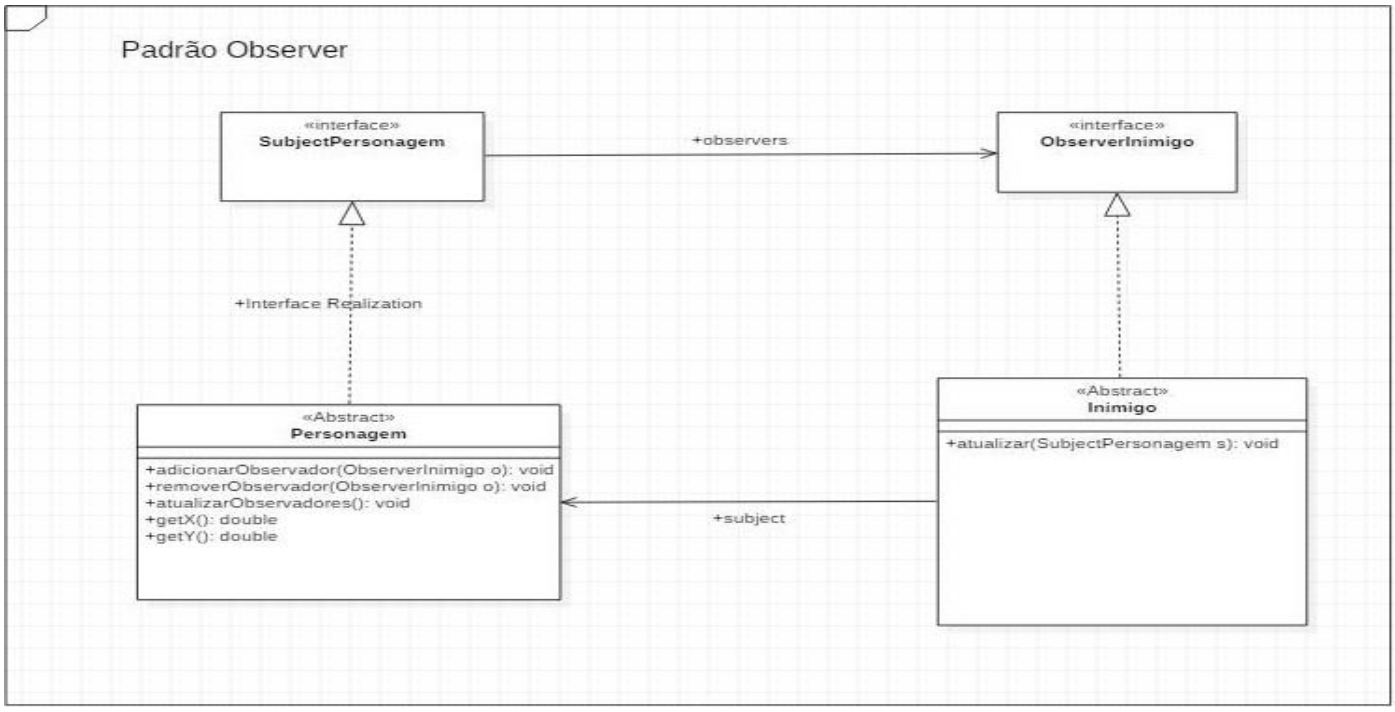


- Padrão Simple Factory FabricaFase:



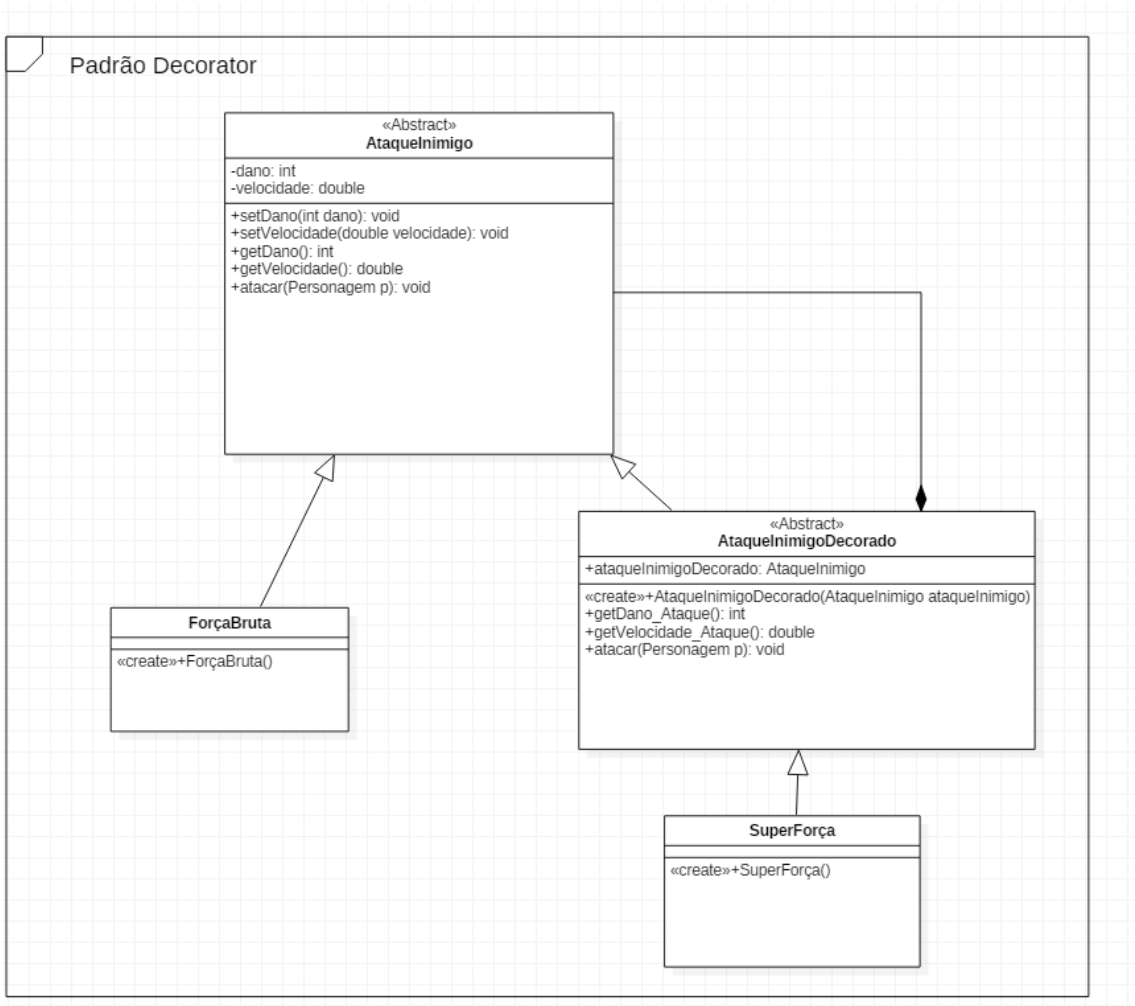
- Padrão Observer:

Para a implementação do Jogo Marcos Prime, foi utilizado o Padrão Observer. Esse padrão possibilita que quando um objeto muda seu estado, todos seus dependentes serão notificados e atualizados para um novo estado. Portanto, no Jogo, esse padrão permite que a Classe SubjectPersonagem envie notificações para a Classe ObserverInimigo, em outras palavras, permite que quando o Personagem se mover com coordenadas x e y, os Inimigos se movimentam para perto dele, então a cada mudança de caminho do Personagem, o Inimigo é notificado dessa mudança.



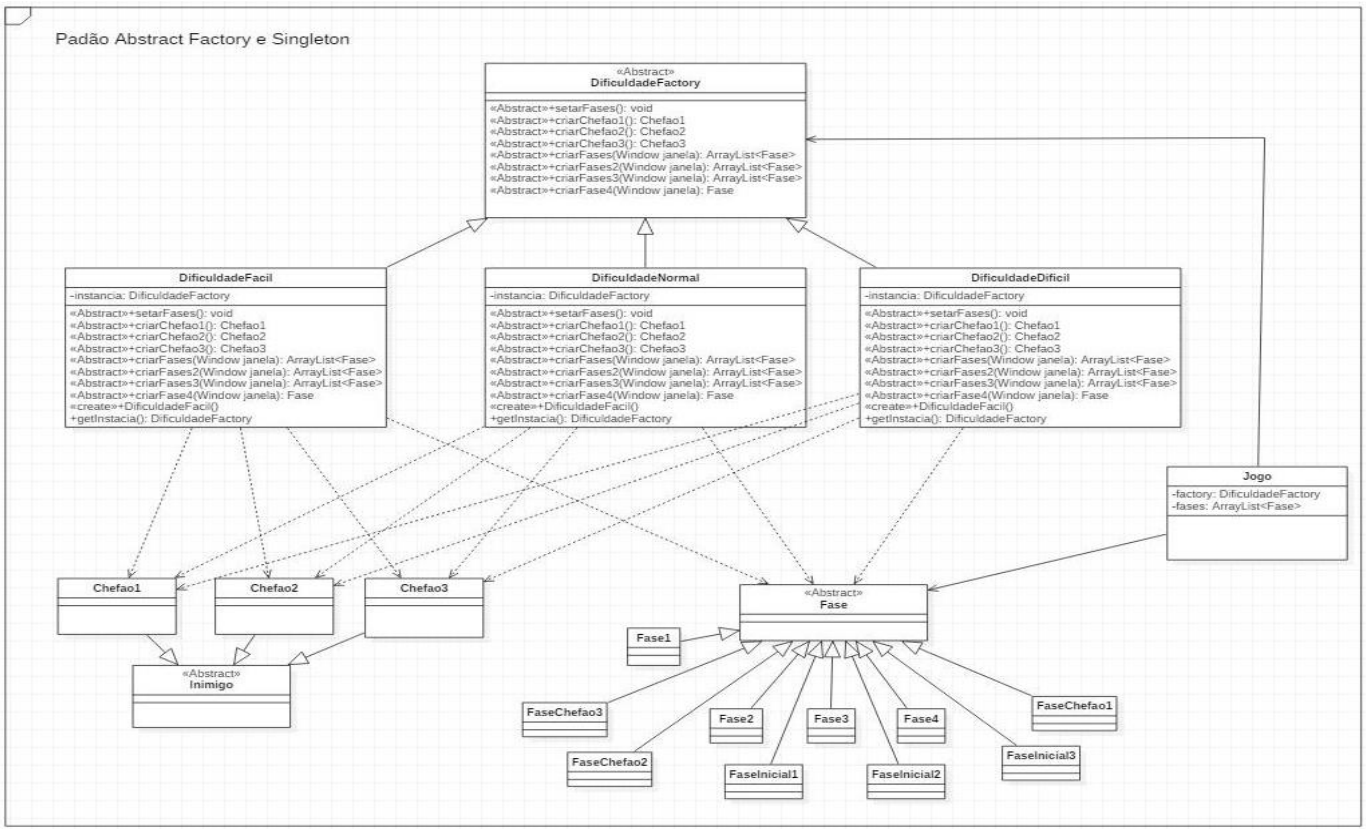
- Padrão Decorator:

Para a implementação do Jogo Marcos Prime, foi utilizado o Padrão de Projeto Decorator. Esse padrão possibilita que em tempo de execução seja permitido adicionar funcionalidades ao objeto em específico. Portanto, no Jogo, esse padrão permite que a Classe AtaqueInimigo seja decorada com além da ForçaBruta, a SuperForça. O chefão Monstro possui esse poder.



- Padrão Abstract Factory e Singleton:

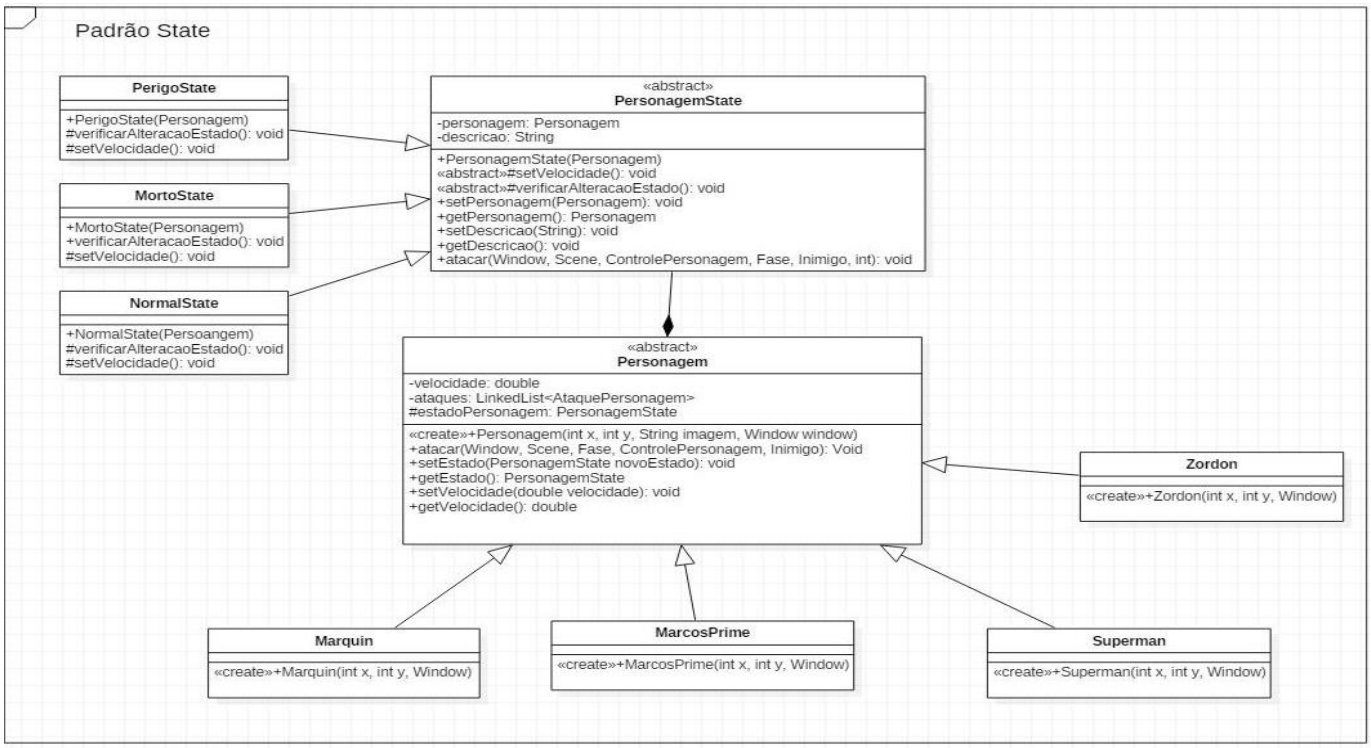
Para a implementação do Jogo Marcos Prime, foi utilizado o Padrão de Projeto Abstract Factory. Esse padrão possibilita sejam criadas famílias de produtos concretos diferentes relacionados ou dependentes por meio de uma interface. Portanto, no Jogo, esse padrão permite que a Classe DificuldadeFactory crie as dificuldades que o jogo precisa disponíveis no modo história, que são as dificuldades: fácil, médio e difícil. Em cada uma dessas Classes são implementados diferentes métodos que serão utilizados para criação das fases do jogo. Foi utilizado também, o padrão Singleton que garante que apenas uma classe seja responsável por sua única instância. Dessa forma, as Classes DificuldadeFacil, DificuldadeMedio e DificuldadeDificil implementam esse padrão oferecendo às suas classes dependentes uma forma mais flexível e única de se instanciar.



- Padrão State:

Para a implementação do Jogo Marcos Prime, foi utilizado o Padrão State. Esse padrão possibilita que seja alterado o comportamento do objeto somente quando o estado do mesmo muda. Portanto, no Jogo, esse padrão permite que a Classe PersonagemState armazene um objeto da Classe Personagem e altera o comportamento de cada um dos Personagens (MarcosPrime, Marquin, Superman e Zordon) quando algo acontecer com eles em um dado momento no jogo. No caso é alterado somente o atributo velocidade do jogador, sendo que cada um dos personagens citados possui sua própria velocidade.

O personagem quando é criado, é instanciado com o NormalState e sua velocidade não muda. Conforme for jogando e for atacado pelos inimigos, sua vida diminui. Quando ela cai para 500 ou menos, o estado do Personagem muda para PerigoState e sua velocidade aumenta para um valor fixo, assim, facilitando sua fuga dos inimigos. E quando sua vida diminui para 0, significa que o jogador morreu o jogo se encerra.



5. PADRÃO ARQUITETURAL MVC (MODEL-VIEW-CONTROL)

Model-View-Control é um padrão arquitetural de software que divide as informações e regras de negócio da interação com o usuário. Essa arquitetura é separada em modelo, visão e controle. Dentro da estrutura Modelo, consistem as classes que dominam a solução do problema, em outras palavras, é o núcleo do Jogo Marcos Prime, onde estão os padrões de projeto e onde estão a lógica e as funções do jogo. Já na estrutura Visão, está representada a interface gráfica que fará interação diretamente com o usuário, apresentando imagens e dados do jogo. Por fim, a estrutura Controle faz intermédio da comunicação entre o Modelo e a Visão, isto é, é essa estrutura que controla e orienta a aplicação como um todo.

O Jogo Marcos Prime optou pelo padrão arquitetural MVC, pois foi priorizada a organização do projeto colocando cada camada com sua responsabilidade, sem haver mistura de Modelo, Visão e Controle. Desta forma, quaisquer alterações feitas em alguma delas, não afetarão a manipulação dos dados.

Usamos a camada Visão para tratar a interface gráfica, como a tela de início, e também para mostrar as informações durante o jogo. Assim, temos quatro classes principais: TelaInicial, correspondente à tela de início e escolha do modo de jogo, TelaDificuldade, correspondente à tela de seleção de dificuldade do jogo, TelaPersonagem, correspondente à seleção do personagem por parte do jogador, e FaseInfo, que mostra informações da Fase, variando para cada uma delas. Algumas dessas informações são a vida do personagem, quantidade de inimigos, vida do chefão, entre outros.

Usamos a camada Controle para tratar eventos relacionados ao teclado, assim conseguir diminuir o acoplamento de eventos com as classes do domínio do problema. Assim, temos duas classes principais: ControleTela, que corresponde aos eventos na Interface Gráfica, como a escolha do modo de jogo, de dificuldade, de personagem, e saída do programa com o

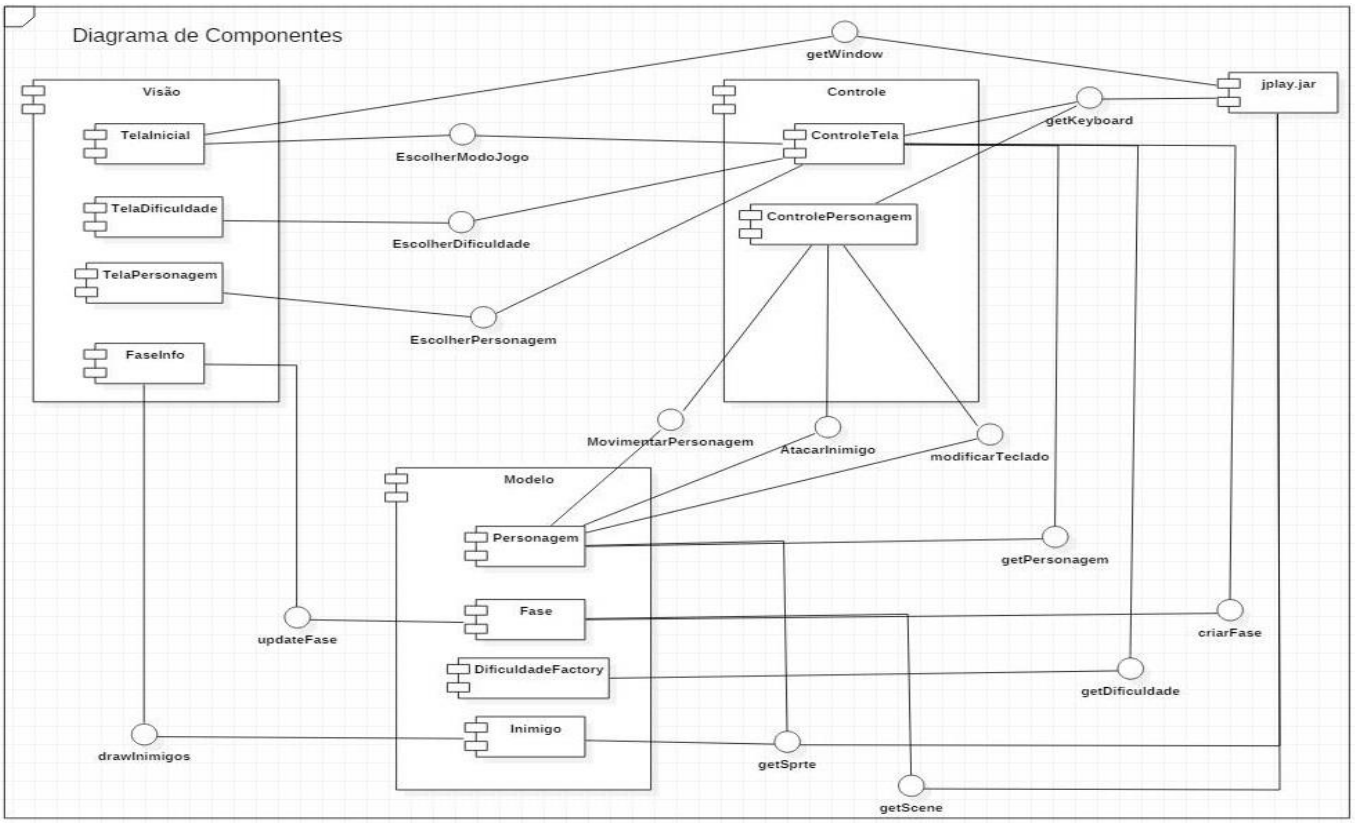
botão ESC.
Trecho de sua utilização do código:

```
//Verifica a escolha do modo do jogador.
public void escolherModo(Window window) {

    if((telaIni.getOpcao() == 0)&&(keyboard.keyDown(Keyboard.ENTER_KEY)) )
    {
        telaIni.modoArcade();
    }

    if((telaIni.getOpcao() == 1)&&(keyboard.keyDown(Keyboard.ENTER_KEY)) )
    {
        telaIni.modoMultiplayer();
    }

    if((telaIni.getOpcao() == 2)&&(keyboard.keyDown(Keyboard.ENTER_KEY)) )
    {
        telaIni.modoSurvival();
    }

    //Se apertar a tecla ESC, sai da tela inicial.
    if (keyboard.keyDown(Keyboard.ESCAPE_KEY))
        window.exit();

}
```

E também temos a classe ControlePersonagem, que trata da movimentação do personagem e também do seu ataque.
Trechos de sua utilização no código:

```
//Trata evento de ataque do player 1 para todos os modos.
public boolean verificarAtaque() {

    if(teclado.keyDown(Keyboard.SPACE_KEY)) {

        return true;

    }
    else
        return false;

}
```

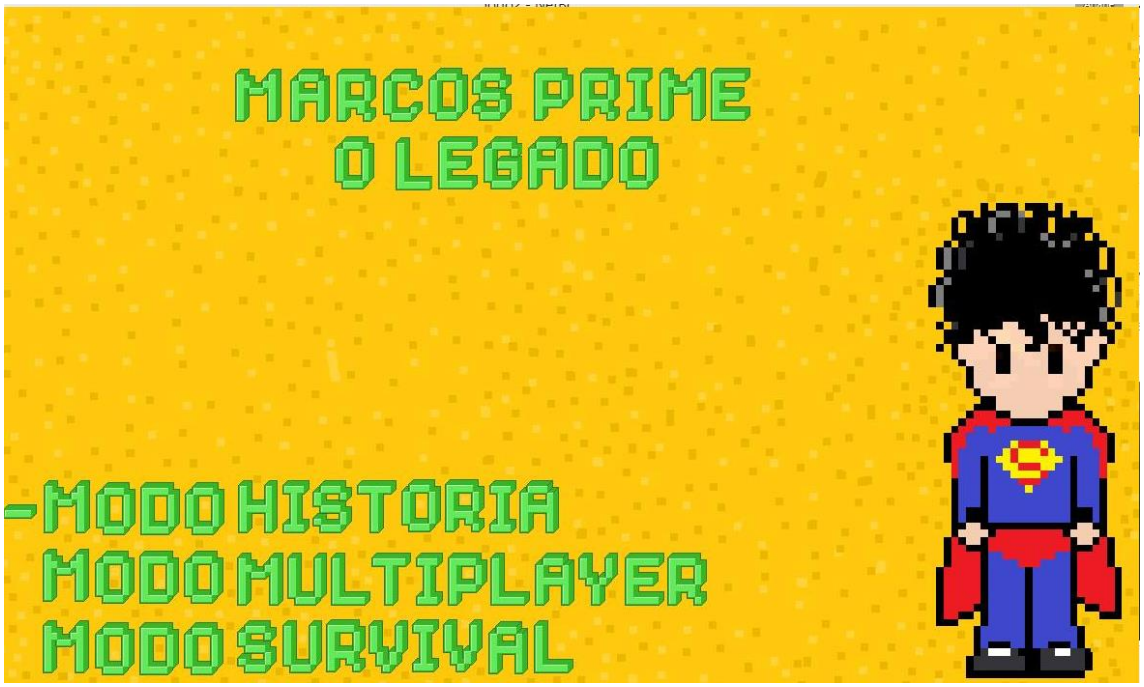
Para esse projeto, foi construído um Diagrama de Componentes que apresenta em detalhes em como essa divisão está sendo representada:



6. DIAGRAMA DE CLASSE

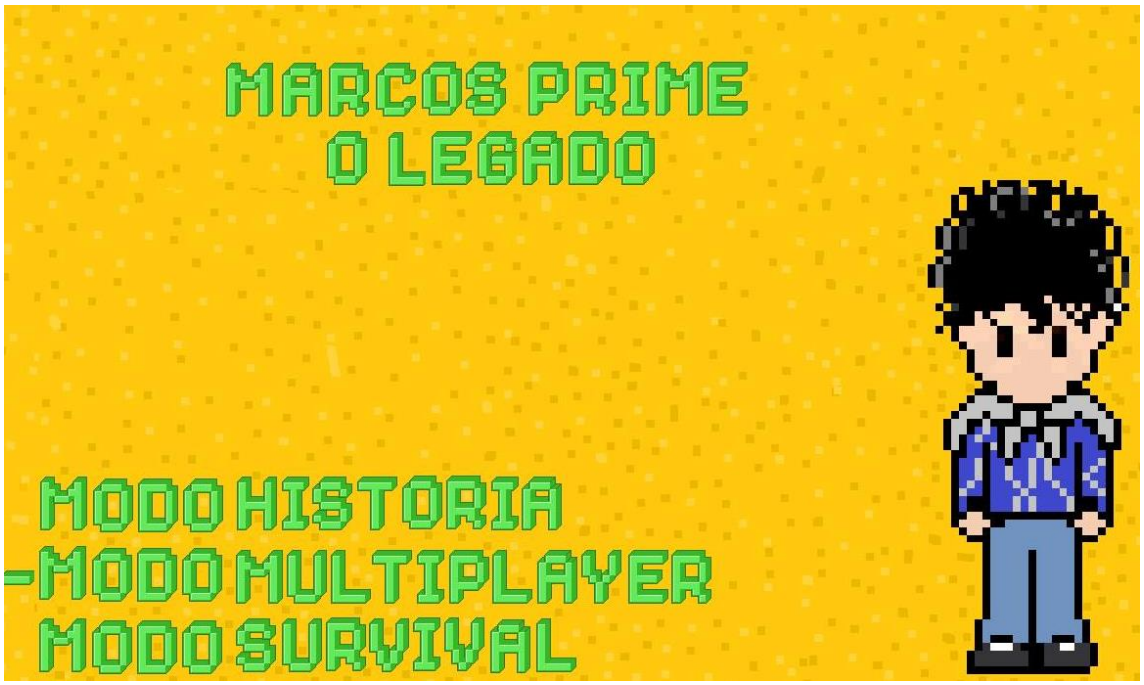
7. INSTRUÇÕES DE JOGO

Como dito no início do relatório, no Jogo Marcos Prime foi utilizado uma interface gráfica baseada em Pixels tanto nas fases quanto na iteração com o usuário. Segue abaixo a imagem de tela de escolha do Modo História:



Como pode ser visto, ao selecionar o Modo História, o jogador pode escolher em qual fase (fácil, médio ou difícil) deseja jogar usando apenas as setas para baixo e cima e apertando ENTER. Ao iniciar o jogo, são usadas as setas para movimentação do personagem e a tecla espaço para atirar nos inimigos.

Ao escolher o Modo Multiplayer, para o primeiro jogador serão usadas as teclas de setas para a movimentação do jogo e a tecla ENTER para atirar nos inimigos. Enquanto para o segundo jogador, serão usadas as teclas “w”, “d”, “a”, ”s” para movimentar o personagem nas direções cima, direita, esquerda e baixo, respectivamente e a tecla espaço para atirar nos inimigos. Segue abaixo a tela de escolha do Modo Multiplayer:



Ao escolher o Modo Survival, o jogador poderá escolher o personagem (Marquin, MarcosPrime, Superman ou Zordon) desejado e ao clicar poderá jogar em fases aleatórias. O modo de jogar é o mesmo do Modo História. Segue abaixo a tela de escolha do Modo Survival: