Proposta do desenvolvimento de um jogo utilizando HTML5, CSS e JavaScript

Vinícius Flores Maier[1], Gustavo Stangherlin Cantarelli[1]

[1]Centro Universitário Franciscano
Santa Maria – RS – Brasil
maiervf@gmail.com, gus.cant@gmail.com

Abstract. *This article presents a proposal for a study of linguem HTML5 and technologies used in web development using the production of a play as a source of study. Conduct a study to develop a complete game while verifying the capabilities and limitations of HTML5 Canvas and mainly, and compare their performance and behavior on all major browsers currently available.*

Resumo. *Este artigo apresenta a proposta de um estudo da linguem HTML5 e das tecnologias utilizadas no desenvolvimento web, utilizando a produção de um jogo como fonte de estudo. Realizar um estudo para desenvolver um jogo completo e ao mesmo tempo verificar as capacidades e limitações do HTML5 e principalmente do Canvas, e comparar seu desempenho e comportamento nos principais navegadores disponíveis atualmente.*

1. Introdução

O desenvolvimento de aplicações web vem popularizando-se diariamente, com uma variedade enorme de temas abordados, desde o desenvolvimento de sistemas completos de gestão até aplicações simples com objetivos específicos como uma lista de compras. Essa popularização cresce na mesma proporção em que os meios de desenvolver essas aplicações são cada vez mais facilitados (JUNIOR, VIDAL, 2005). O desenvolvimento de jogos é um dos temas que apresenta bons resultados, onde é possível desenvolver projetos sem a necessidade um grande conhecimento prévio.

Analisando o mercado de jogos eletrônicos atuais, conclui-se que os primeiros surgiram a mais de 50 anos, em uma época em que a informática começa a dar seus primeiros passos. Foi em 1958 que Willy Higinbotham desenvolveu o jogo que batizou de *Tennis Programming,* jogo que foi utilizado para promover um laboratório de Física em Nova Iorque (KALNING, 2008). Infelizmente, o jogo nunca chegou a ser patenteado ou comercializado. Após alguns anos, em 1962, surgiu o primeiro jogo comercializado e patenteado do mundo, o "Spacewar!", desenvolvido na linguagem Assembly por uma equipe do *Massachusetts Institute of Technology* (MIT). Da mesma maneira que o jogo desenvolvido por Willy Higinbotham, o jogo tinha por objetivo principal atrair a atenção dos visitantes de instituto das capacidades computacionais existentes (A História dos Videogames, 2014).

Desde a sua primeira geração até hoje, os jogos evoluíram paralelamente com os recursos computacionais e tecnológicos, passaram a movimentar um mercado de bilhões de dólares por ano, sendo encontrados nas mais diversas plataformas e nos mais variados gêneros (LOBO, VERDI, ELIAS, 2012).

1.1. Justificativa

Desenvolver um jogo utilizando as tecnologias de desenvolvimento atuais é uma maneira atrativa de realizar um estudo aprofundado sobre as limitações e desempenho dessas tecnologias como um todo. Sendo esse desenvolvimento um tema atrativo, utilizá-lo dessa forma pode facilitar essa obtenção de resultados.

Como o desenvolvimento de um jogo é uma tarefa que pode envolver diversas tecnologias web em um mesmo projeto, ele pode facilitar o entendimento dessas tecnologias como um todo e como elas se interligam, visando ao final do projeto melhorar o conhecimento das variáveis do desenvolvimento web como um todo.

1.2. Objetivo Geral

O objetivo desse projeto é melhorar o conhecimento no desenvolvimento de sistemas web baseados em HTML5 através da programação de jogos.

1.3. Objetivos Específicos

- Realizar um estudo do Canvas e seu desempenho, buscando encontrar maneiras da aplicação do mesmo em novos projetos.
- Pesquisar maneiras de renderizar imagens no Canvas de maneira a consumir o mínimo de recursos computacionais disponíveis, facilitando sua portabilidade para diversos dispositivos.
- Desenvolver o roteiro do jogo, bem como obter todas as mídias necessárias para desenvolvê-lo, como sons e imagens.
- Avaliar o comportamento, compatibilidade, desempenho do jogo em diferentes navegadores.
- Adquirir alguns indicadores referentes ao desenvolvimento, como tempo de desenvolvimento, tamanho do arquivo e curva de aprendizado na API.

Esse trabalho está organizado por seções, apresentando o referencial teórico do mesmo na seção dois, com uma abordagem das tecnologias estudadas e metodologia de desenvolvimento. Na seção três serão apresentados os trabalhos relacionados ao tema do trabalho. Nas seções seguintes será apresentado, respectivamente, o jogo, com suas peculiaridades e sistemática, e o projeto do mesmo, com seus diagramas e funcionalidades.

2. Referencial Teórico

Nessa seção serão apresentados os conceitos básicos para o entendimento do trabalho, listando as principais tecnologias envolvidas em aplicações web, focando no referencial do HTML5 e do Canvas.

2.1. HTML5

Segundo a W3C (*World Wide Web Consortium*) a web baseia-se em três pilares: Um esquema de nomes para localização de informações (URL ou *Uniform Resource Locator*), um protocolo de acesso a esse esquema de nomes (HTTP ou *Hypertext Transfer Protocol*) e uma linguagem de hipertexto que interprete todas essas fontes de informação (HTML ou *HyperText Markup Language*). Dentro desse contexto o HMTL pode ser visto como linguagem para publicação de conteúdo na web (W3C, 2012).

Entre os anos de 1993 e 1995 a linguagem HTML dava seus primeiros passos. Nesse período surgiram as três primeiras versões (1, 2 e 3), que infelizmente não possuíam um padrão bem definido. Apenas no ano de 1997 a W3C começava a regular e padronizar a linguagem. A W3C, após a aceitação mundial do HTML e popularização da internet, começou a trabalhar em um projeto conhecido como XHTML (*eXtensible Hypertext Markup Language*), que deveria ser uma reformulação do HTML, mesclando aspectos já conhecidos da linguagem com os padrões de XML (*eXtensible Markup Language*) (FERREIRA, EIS, 2013).

Em contrapartida, o grupo conhecido como WHATWG (*Web Hypertext Application Technology Working Group*), cujos participantes eram programadores em empresas de grande porte, tais como Apple, Microsoft e Mozilla, não estavam satisfeitos com o caminho para qual a W3C estava levando a HTML e começaram a trabalhar em um projeto que hoje é conhecido como HTML5. Desde 2006 ambos os grupos trabalham em conjunto, desenvolvendo os padrões do HTML até os dias de hoje (FERREIRA, EIS, 2013).

Em termos de mudança, o HTML5 acrescentou, além de melhorias nos elementos DOM (*Document Object Model*), três grandes novidades: As *tags* semânticas, as *tags* Multimídia e a API Canvas (MOZILLA, 2013).

Elementos DOM são basicamente a interface dá página, é um conjunto de elementos dispostos em uma estrutura de árvore que podem ser manipulados via CSS e Script para que se comporte de uma maneira que atende as necessidades dos desenvolvedores.

Uma *tag* é a marcação do HTML que é interpretada pelos navegadores. Em grande maioria são apresentadas em duplas, sendo que uma *tag* representa a abertura do contexto e outra o fechamento. Por exemplo, quando for necessário começar um novo parágrafo no HTML, utiliza-se a estrutura `<p>Texto do parágrafo.</p>`; onde e são *tags* que representam respectivamente o inicio e o fim do parágrafo (MUSCIANO, KENNEDY, 2006).

Tags semânticas nada mais são que tags que aplicam um contexto que anteriormente não existia no HTML, como por exemplo, *header*, que simboliza o cabeçalho da página, e *section***,** que representa uma sessão genérica, como notícias ou introdução (MARTINS, MARCHI, 2013). Segundo Hey (2010), esses *tags* possuem o objetivo de facilitar a estruturação de documentos HTML e facilitar a renderização dos mesmos nos mais diversos dispositivos, sejam computadores, celulares ou *tablets*.

Tags multimídia são responsáveis por reproduzir áudio e vídeo sem a utilização de nenhum *plugin* de terceiros (MARTINS, MARCHI, 2013), como era necessário nas versões antigas. Apesar de reproduzidas de maneira extremamente simples e consumindo poucos recursos do navegador, as *tags* infelizmente não possuem 100% de compatibilidade de em todos os navegadores, portanto ainda não são amplamente utilizadas (IRISH, COOLEY, 2011). A Figura 1 exemplifica como devem ser declaradas as *tags* de áudio e vídeo.

```
<audio controls="controls">
    <source src="audio.ogg" type="audio/ogg" />
    <source src="audio.mp3" type="audio/mp3" />
    Sem suporte a áudio em HTML5
</audio>

<video width="600" height="400" controls="controls">
    <source src="video.mp4" type="video/mp4" />
    <source src="video.ogg" type="video/ogg" />
    Sem suporte a vídeo em HTML5
</video>
```

Figura 1 - Exemplo de sintaxe das tags áudio e vídeo do HTML5

Segundo Firmino (2010) o HTML5 tem como principal objetivo facilitar a estruturação e apresentação de dados no navegador sem o uso de *plugins* de terceiros. Dessa maneira, é possível obter um melhor desempenho com um menor consumo dos recursos computacionais disponíveis. O Canvas reserva uma área na pagina para a exibição de animações e ainda conta com uma API (*Application Programming Interface*)

que inclui operações para possibilitar a construção de formas básicas de desenho (W3C, 2012).

Essas foram apenas duas mudanças apresentadas nessa versão do HTML. Houve uma melhoria em muitos fatores da linguagem, como por exemplo, formulários e seus validadores, *Local Storage*, que permite salvar dados diretamente no navegador para futuras consultas, aplicações que podem ser acessadas *offline* e até mesmo o *drag-and-drop,* ou arrastar-e-soltar, que antes era praticamente inaplicável, pois exigia uma carga enorme de scripts, além de possuir vários bugs conhecidos (FERREIRA, EIS, 2013).

2.1.1. API Canvas

A Canvas API permite a você desenhar na tela do navegador utilizando Javascript. O elemento Canvas pode ser considerado um container, um quadro para apresentar os desenhos e animações, todo o resto fica a cargo do script (FULTON, FULTON, 2013), normalmente Javascript. Na figura 2, a declaração do elemento Canvas na página HTML.

```
<canvas id="myCanvas" width="640" height="480">Sem suporte ao HTML5</canvas>
```

Figura 2 - Declarando um elemento Canvas

É uma boa prática declarar um atributo ‘id’ para o Canvas, para facilitar obter o contexto do mesmo com Script. Um ‘id’ é utilizado como identificador único, facilitando a busca desse elemento tanto pelo Javascript como pelo CSS da página. Além disso, qualquer texto apresentado entre as tags <canvas></canvas> será apresentado caso o navegador não tenha suporte ao mesmo, então é comum definir uma mensagem de erro (MARTINS, MARCHI, 2013). O elemento aceita estilos CSS (Cascading Style Sheets), como, por exemplo, *background-col*or, *border* e *float*.

Após a declaração do Canvas nada mais é necessário no HTML, basta utilizar o Javascript para manipular o mesmo e realizar os desenhos. Existem vários métodos prontos que permitem desenhar de maneira fácil no Canvas, desde formas geométricas até gradientes e animações. Tal elemento nada mais é que um quadro que possui uma altura e uma largura definida. Os métodos de desenho utilizam sempre pontos nesse quadro, formando coordenadas (x, y) para se localizar no Canvas e a partir daí desenhar o que é proposto (FULTON, FULTON, 2013). Para realizar as animações e desenhos, é preciso obter o contexto do Canvas, como apresentado na Figura 3.

```
<body>
    <canvas id="myCanvas" width="640" height="480">Sem suporte ao HTML5</canvas>
</body>
<script>
    var canvas = document.getElementById('myCanvas');
    var context = canvas.getContext('2d');
    // Sua Lógica vem aqui
</script>
```

Figura 3 - Adquirindo o contexto do Canvas para manipulação via Javascript

Após adquirir o contexto, podem-se utilizar os métodos para desenhos que a API nos fornece. Os métodos mais comuns empregados para desenho são geométricos, como círculos, retângulos ou quadrados e apresentação de imagens.

2.2. CSS

Cascading Style Sheets (Folha de estilo em cascata) ou simplesmente CSS é aquilo que formata o HTML. Com ele, podemos alterar características como cores, tamanho e estilo de fonte posição de objetos (W3C, 2013). A grande utilidade do CSS é a de separar a marcação HTML de seu estilo. Basicamente, o HTML deverá ser utilizado para gerar o conteúdo do site e suas marcações, como tabelas, cabeçalhos e rodapés, enquanto a cor e plano de fundo das tabelas, fontes dos rodapés e manipulação de alguns efeitos ficam por conta do CSS (Silva, 2014).

Ainda segundo Silva, a ideia de "Cascata" refere-se à utilização de um arquivo CSS para diversos arquivos HTML, podendo ser reutilizado em várias páginas sem a necessidade de replicação de código. Essa técnica economiza um trabalho considerável ao desenvolver aplicações web, pois caso seja necessário utilizar o mesmo tipo de fonte e a mesma cor em 50 páginas distintas, é possível utilizar apenas um arquivo CSS que será herdado por todas as páginas em que o mesmo for referenciado. Na Figura 4 pode-se observar a estrutura de arquivos do CSS e o seu estilo cascata.

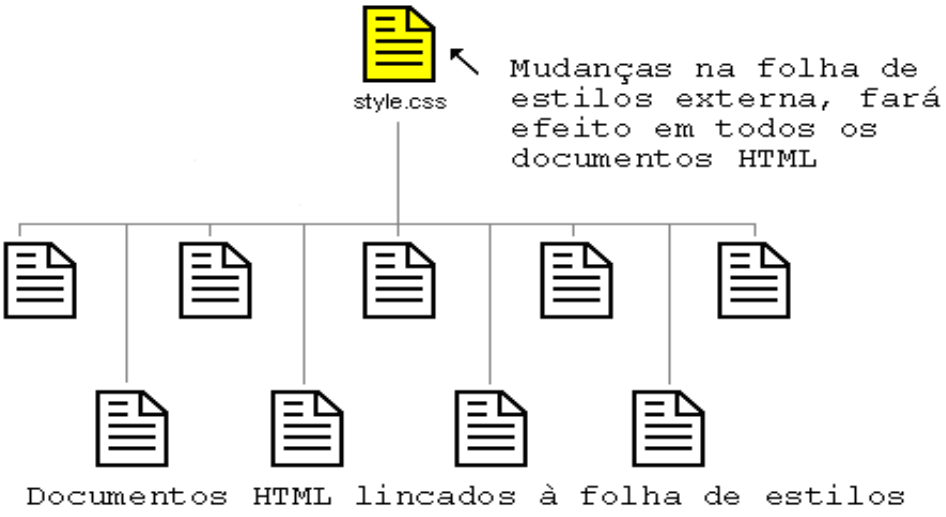

Figura 4. Exemplo de utilização da folha de estilos

2.3. Javascript

JavaScript é uma linguagem de programação baseada em scripts e padronizada pela ECMA *International* (*European Computer Manufacturers Association*). É interpretada do lado do cliente (conhecido como *client-side*) e todos os navegadores atuais possuem interpretadores de *script* nativos (MOZILLA, 2013). Dentre suas características principais podemos citar a tipagem fraca e estruturação voltada a prototipagem, que representa uma estrutura semelhante a orientação à objetos, de uma maneira um pouco diferente. O Javascript trabalha com o conceito de objetos, e não classes, e esses objetos podem ser modificados dinamicamente através da função padrão *prototype.*

O Javascript é responsável por grande parte dos eventos que ocorrem nas páginas e aplicações web, como por exemplo, a apresentação de mensagens ou alertas, carregamento dinâmico de conteúdo, *post* de formulários, dentre vários outros. Seu ponto fraco é justamente ser uma linguagem *client-side*, podendo ser manipulado, ou até mesmo desabilitado, diretamente no navegador do cliente, deixando-o muito vulnerável a manipulação do código por parte de usuários capacitados.

No contexto desse trabalho, é o Javascript que realiza todas as alterações realizadas no Canvas, desde a movimentação de personagens do jogo, até a atribuição de eventos a determinados elementos.

3. Trabalhos Relacionados

Nessa seção serão apresentados os trabalhos que foram utilizados como base para o desenvolvimento, que se enquadram no mesmo escopo.

3.1. O navegador como uma plataforma para jogos: Uma experiência extracurricular para o desenvolvimento de software

O trabalho realizado por Henrique et al. (2011) utilizou o desenvolvimento de jogos utilizando Canvas como uma maneira de praticar o desenvolvimento de softwares por alunos do curso de Sistemas de Informação, pois apesar do mesmo abordar uma grande carga de teoria acerca de desenvolvimento, existe uma falta de experimentação pratica por parte dos alunos.

O artigo apresenta uma definição das primitivas utilizadas no Canvas para realizar o desenho de formas geométricas como polígonos, quadrados e círculos, bem como técnicas de detecção de colisão e aplicação de imagens ao Canvas. Além disso, relata que o envolvimento e empenho dos alunos envolvidos, bem como o aprendizado adquirido pela equipe de alunos foi satisfatório.

Como consideração final, os autores ressaltaram que o desenvolvimento utilizando o Canvas possui uma curva pequena de aprendizado, sendo facilmente entendida por desenvolvedores sem um prévio conhecimento da linguagem Javascript e que o mesmo pode ser desenvolvido sem a necessidade de um ambiente especial de desenvolvimento.

Foi importante a análise desse trabalho para entender as dificuldades enfrentadas pelos estudantes ao desenvolver o primeiro jogo, pois os mesmo também eram alunos do curso de Sistemas de Informação. A principal diferença entre os trabalhos é a quantidade de membros envolvidos no grupo de desenvolvimento, que foi fundamental para o bom desempenho dos estudantes no projeto.

3.2. Desenvolvimento de Jogos em HTML5

Nesse artigo, o autor descreve técnicas mais avançadas para a manipulação do Canvas, utilizando algoritmos e recursos que visam melhorar o desempenho do jogo e a experiência do usuário final. Esses contextos foram utilizados no desenvolvimento de um jogo semelhante ao famoso “Snake”, popular nos primeiros celulares de alguns anos atrás, e em um jogo da velha que utiliza inteligência artificial.

Os desenvolvedores do trabalho optaram por manipular o Canvas utilizando Javascript, e as teorias apresentadas por eles no trabalho serão de grande valia para o desenvolvimento do projeto, visto que foi realizada uma pequena experimentação para descobrir qual a melhor forma de empregas os métodos da API Canvas. Utilizando desse ferramental eles obteram bons resultados no desenvolvimento do jogo mesmo com a inexperiência da equipe, o que foi considerado um ponto positivo para o projeto que será desenvolvido.

4. Projeto

O jogo desenvolvido será chamado de “Aventura Épica”, mas será referenciado a partir de agora somente como AE. O objetivo principal do mesmo é apresentar ao usuário (jogador) uma experiência de jogo em que o mesmo precise traçar estratégias dentro do jogo para atingir o objetivo final. Mesmo se tratando de um jogo online, não será necessário realizar nenhum login para jogá-lo, e todas as informações referentes ao jogador, como nome e resultados das partidas anteriores utilizarão algumas funcionalidades do HTML5 para armazenamento local, abstraindo-se assim questões referentes a banco de dados.

Existirão dois modos de jogo, selecionados previamente pelo jogador ao começar o jogo, o modo campanha e o modo desafio, cada um deles com suas particularidades. No modo campanha, haverá uma sequência de fases que o jogador deve concluir para acabar o jogo e vencer.

Para um completo entendimento da modelagem do projeto, é necessário primeiro conhecer os aspectos e roteiros do jogo a ser desenvolvido. Optou-se por desenvolver um jogo que se enquadre nos estilos de um *Turn-based strategy* (TBS), ou jogo de estratégia baseado em turnos. O exemplo mais clássico de um TBS é o xadrez, onde cada jogador possui um tempo definido (turno) para realizar uma ação, e a mesma não pode acontecer simultaneamente entre os dois jogadores.

Antes de explicar o funcionamento do jogo, é necessário apresentar algumas terminologias que serão utilizadas para descrevê-lo. São elas:

- Jogador: O usuário que irá jogar o jogo.
- IA: A inteligência artificial da aplicação, que tentará ganhar o jogo do personagem.
- Mapa: Cada mapa é um estágio do jogo, onde aconteceram as partidas.
- Partida: Segundo BYL (1990), uma partida é um confronto entre dois times adversários, sendo um o jogador e o outro a IA, onde cada um possui um turno para posicionar o seu exército, cuja vitória ocorrerá quando o jogador ou a IA não possuírem mais Heróis no mapa, nem ouro necessário para recrutar outro Herói.
- Turno: Unidade de tempo de até 30 segundos, onde o jogador deverá realizar suas ações no jogo.
- sqm (*Square metre*): Será a unidade de medida para movimentação dos Heróis e para geração do mapa. Cada sqm será um quadrado de *n* x *n pixel*, e cada herói possui um número limite de sqm's disponíveis para se locomover por turno, de acordo com suas características.
- Herói: É a unidade que o jogador (e a IA) controlam.
- Atributos: São as características de cada herói, que buscam individualizar cada um. Os valores das características são sempre números inteiros individuais para cara herói. São elas:
    1. Ataque: O quanto de dano aquele herói pode causar.
    2. AtaqueSqm: A distância, calculada em sqm's, que o ataque pode chegar.
    3. Defesa: Valor subtraído de cada ataque antes de reduzir da vida do alvo atacado.
    4. Vida: Define o quão resistente é o herói. Quando a vida de um herói chega a zero, indicando que o mesmo está derrotado e é removido do mapa.
    5. Deslocamento: Define quantos sqm's o herói pode mover-se no seu turno.
    6. Custo: A quantidade de ouro necessário para recrutar esse herói e colocá-lo no mapa, durante a partida.
- Exército: É o conjunto de heróis do jogador e da IA.
- Ouro: É a moeda existente no jogo. Em cada partida, jogador e IA possuem um valor inicial de ouro, e podem obter mais derrotando heróis inimigos ou encontrando em pontos do mapa.

- Recrutar: Ação de utilizar ouro disponível para comprar um herói (baseado no atributo custo do mesmo), podendo utilizá-lo durante a partida atual.
- Pontos: Valor obtido de acordo com o desempenho do jogador ao vencer (ou perder) a partida. Será apresentado em outro tópico as as formas de se obter pontos durante a partida.
- Zona Inicial: Região do mapa, representada por alguns sqm's, onde é possível posicionar os heróis após recrutá-los.
- Ações: São os atos possíveis de se realizar durante o turno. Cada ação ocorre apenas uma vez por turno para cada herói. As ações disponíveis são:
    1. Mover: Desloca o herói por até *n* sqm's, onde *n* é o valor do atributo deslocamento do herói.
    2. Atacar: Ataca o herói inimigo, desde que o mesmo esteja a uma distância menor ou igual a *n* sqm's, onde *n* é o valor do atributo ataqueSqm do herói atacante. Caso esteja no alcance, o ataque resultada em um dano, que é calculado com base no ataque do atacante subtraído da defesa do atacado. Caso o resultado seja zero ou menor, o dano é igual a 1, caso seja maior que zero, o dano é o resultado da subtração. Esse dano é subtraído da vida do atacado, e caso a mesma seja zero ou menor, o atacado é removido do mapa, bonificando o jogador atacante com um valor em ouro.
    3. Passar: Caso opte por não mover nem atacar, é possível simplesmente passar o turno sem executar nenhuma ação.

Todo o roteiro do AE é baseado nos turnos, onde o personagem define as ações que deseja executar. Não existe um limite de turnos definido por partida. Como o jogo é sequencial, pode-se entender o processo do mesmo da seguinte forma:

1. Escolha do modo de jogo, dentre os dois já apresentados.
2. Escolha do herói inicial: Em ambos os modos de jogo, o jogador começará cada partida com um herói que não será necessário recrutar.
3. Inicio da partida: Após o carregamento do mapa e inicio da partida, o jogador possuirá uma quantia inicial de ouro para recrutar heróis. Baseando-se no atributo custo de cada herói, o valor do mesmo é descontado do ouro que o personagem possui.
4. Turnos: Realiza uma das ações possíveis para cada herói disponível no mapa. O turno é alternante, então após o jogador encerrar seu turno, é a vez da IA realizar o seu.
5. Fim da partida**:** Caso perca todos seus heróis, o jogador é derrotado e retorna para a tela inicial de menu, caso o jogador vença, avança para a próxima partida (na campanha) ou retorna para o menu (no desafio).

5. Modelagem e Implementação

Esse trabalho tem como objetivo desenvolver um jogo utilizando os recursos presentes no HTML5 onde o usuário seja capaz de optar por um dos dois modos de jogo disponíveis, bem como pesquisar seus resultados anteriores. Nas seções seguintes serão abordadas a metodologia de desenvolvimento e as especificações do projeto.

5.1. *Feature Driven Development* (FDD)

FDD ou Desenvolvimento Guiado por Funcionalidades é uma metodologia ágil baseada em cinco processos básicos orientados a funcionalidade. Ela se encontra em um ponto

médio entre as abordagens mais perspectivas, como cascata, e as mais ágeis, como XP, sendo amplamente escolhida por empresas um pouco mais conservadoras, que não concordam com as abordagens tão radicais das metodologias encontradas no mercado (PALMER; FELSING, 2012).

Seus cinco processos básicos são utilizados como um roteiro de desenvolvimento, orientando a equipe do projeto de uma maneira simples.

- Desenvolvimento do modelo: Segundo Barbosa (2008) é o momento de entender o problema a ser resolvido, e estudar uma maneira de solucioná-lo. Esse processo pode envolver levantamento de requisitos, modelagem lógica e outras formas de documentação que serão utilizadas como guia ao longo do projeto (BARBOSA, 2008).
- Listar Funcionalidades: Momento do projeto em que cada funcionalidade deve ser listada e especificada, em ordem hierárquica. Todas as funcionalidades listadas devem ser entregues no produto final (PALMER; FELSING, 2012). Esse processo gera o *product backlog*, também chamada de lista de espera, muito comum nas metodologias ágeis (BARBOSA, 2008).
- Planejar Funcionalidades: Nesse processo cada uma das funcionalidades listada no processo anterior deve ser estimada com relação a prazos de desenvolvimento e custo, de maneira a gerar uma estimativa do projeto (PALMER; FELSING, 2012).
- Detalhar Funcionalidades: Primeiro processo, que já está inserido na fase de desenvolvimento, tem como principal resultado esperado um modelo de domínio mais detalhado e os primeiros trechos de código planejados (BARBOSA, 2008).
- Construir Funcionalidades: Utilizando o material gerado nos processos anteriores, desenvolve-se cada uma das funcionalidades individualmente. No final do processo, todas as funcionalidades são integradas para gerar o produto final (PALMER; FELSING, 2012).

5.2. Projeto

Partindo do objetivo específico do trabalho, é possível iniciar a modelagem do sistema, partindo do levantamento dos requisitos funcionais e não funcionais do projeto. Um requisito funcional define uma função de um software ou parte dele. Ele é o conjunto de entradas, seu comportamento e sua saída, ou seja, envolve cálculos, lógicas de trabalho, manipulação e processamento de dados, entre outros (Makesys, 2013). Os requisitos que foram encontrados para esse trabalho são os seguintes:

1. Permitir a que o usuário selecione o seu nome no jogo, e utilizar o mesmo em partes do jogo que envolva diálogos com o usuário.
2. Possibilitar a escolha de um entre dois dos módulos de jogo planejado (campanha ou desafio rápido).
3. Permitir a visualização da pontuação obtida nas últimas partidas em cada um dos módulos.
4. Permitir ao usuário a escolha de um dos personagens disponíveis para começar o jogo, sendo inicialmente disponíveis dois.

Após a definição dos requisitos funcionais, o próximo objetivo é definir os requisitos não funcionais, que segundo Hazrati (2011) são frequentemente associados com o estado do sistema e não com as funcionalidades oferecidas; incluem características do sistema, como escalabilidade, interoperabilidade, facilidade de

manutenção, desempenho, portabilidade e segurança. Foram encontrados os seguintes requisitos não funcionais para esse projeto:

1. Conexão com a internet, por se tratar de um jogo online, é necessário exisitir conexão para gerar a comunicação entre o usuário e o servidor onde o jogo estará hospedado.
2. Navegador compatível com HTML5, pois grandes partes dos recursos utilizados no desenvolvimento do jogo necessitam ser executado nos navegadores mais modernos, como o Google Chrome, Mozilla Firefox ou Opera.

Definido então os requisitos funcionais e não funcionais, e com o projeto do jogo já descrito, é possível construir um modelo geral de domínio, que segundo Palmer e Pelsing (2002), é um modelo que engloba diversos objetos em um domínio de problema e as relações entre eles. Dessa forma, chegou-se ao seguinte diagrama:

Após a construção do modelo inicial de domínio, o próximo passo exigido pela metodologia utilizada é o desenvolvimento de uma lista de funcionalidades que estarão presentes no sistema. Baseado no roteiro descrito foram levantadas as seguintes funcionalidades:

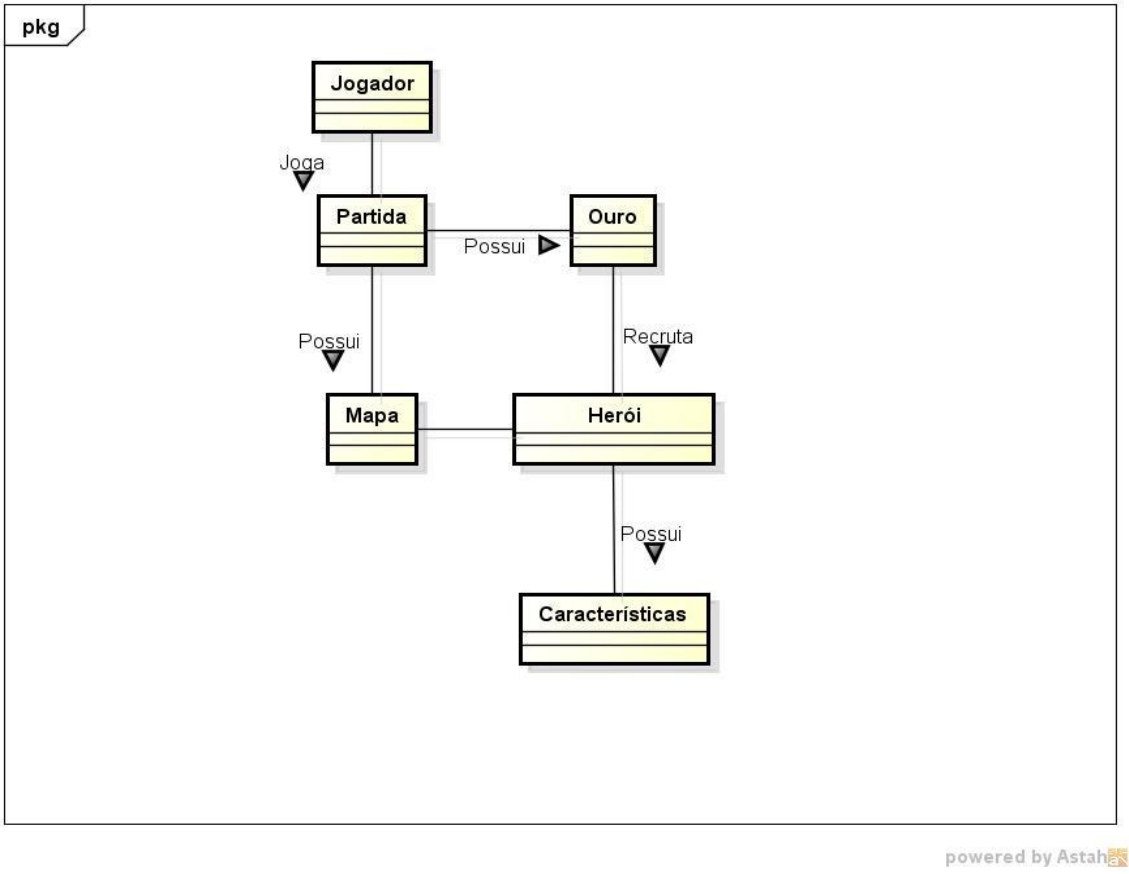

Figura 5 - Diagrama de Domínio do Sistema

- Iniciar jogo
- Selecionar nome
- Selecionar modo de jogo
- Iniciar partida
- Recrutar Herói
- Realizar ataque

- Passar turno
- Movimentar Herói

Ainda de acordo com a metodologia utilizada, o próximo passo para o desenvolvimento da modelagem é o planejamento por funcionalidade. Para mapear esse planejamento, foi desenvolvido um diagrama dos casos de uso utilizando a listagem de funcionalidades descritas, como visto na figura 6.

Após todas as definições, já é possível planejar por funcionalidade, gerando o descritivo dos casos de uso, que segundo Bressan (2012) constitui-se na elaboração de um esquema descritivo para se organizar o caso de uso, de uma maneira clara e documental. Os descritivos podem ser encontrados no apêndice desse trabalho. Além disso, foi desenvolvido o diagrama de atividade, que em conjunto com os casos de uso, que servirá de referência para o desenvolvimento do projeto, como visualizado na Figura 7.

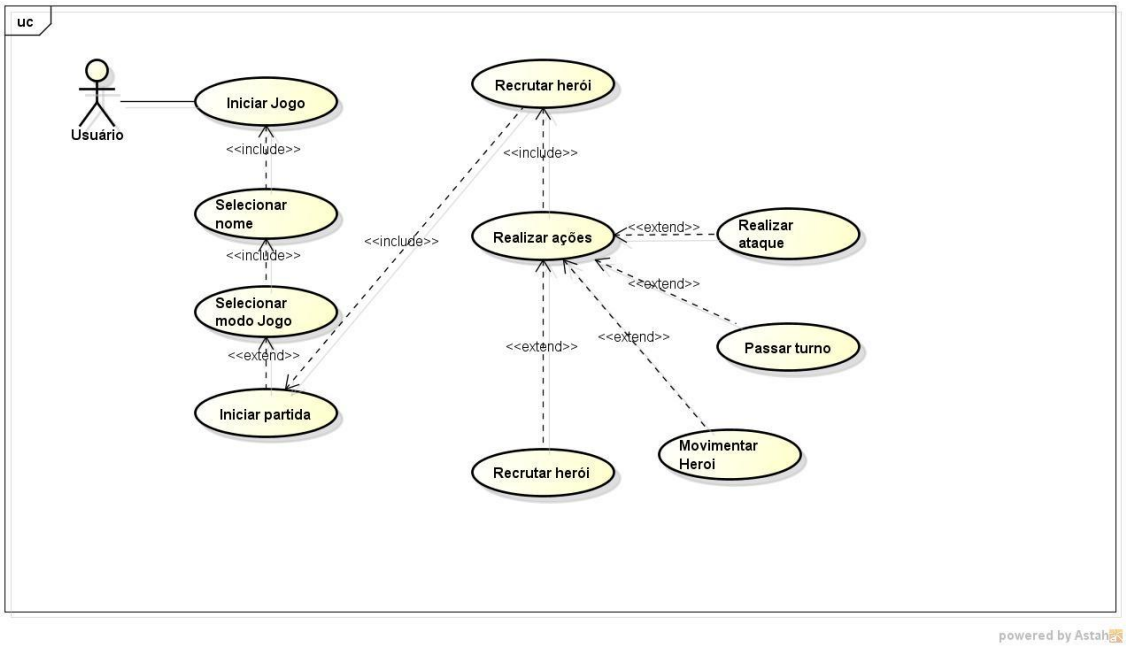

Figura 6 – Diagrama de Casos de Uso

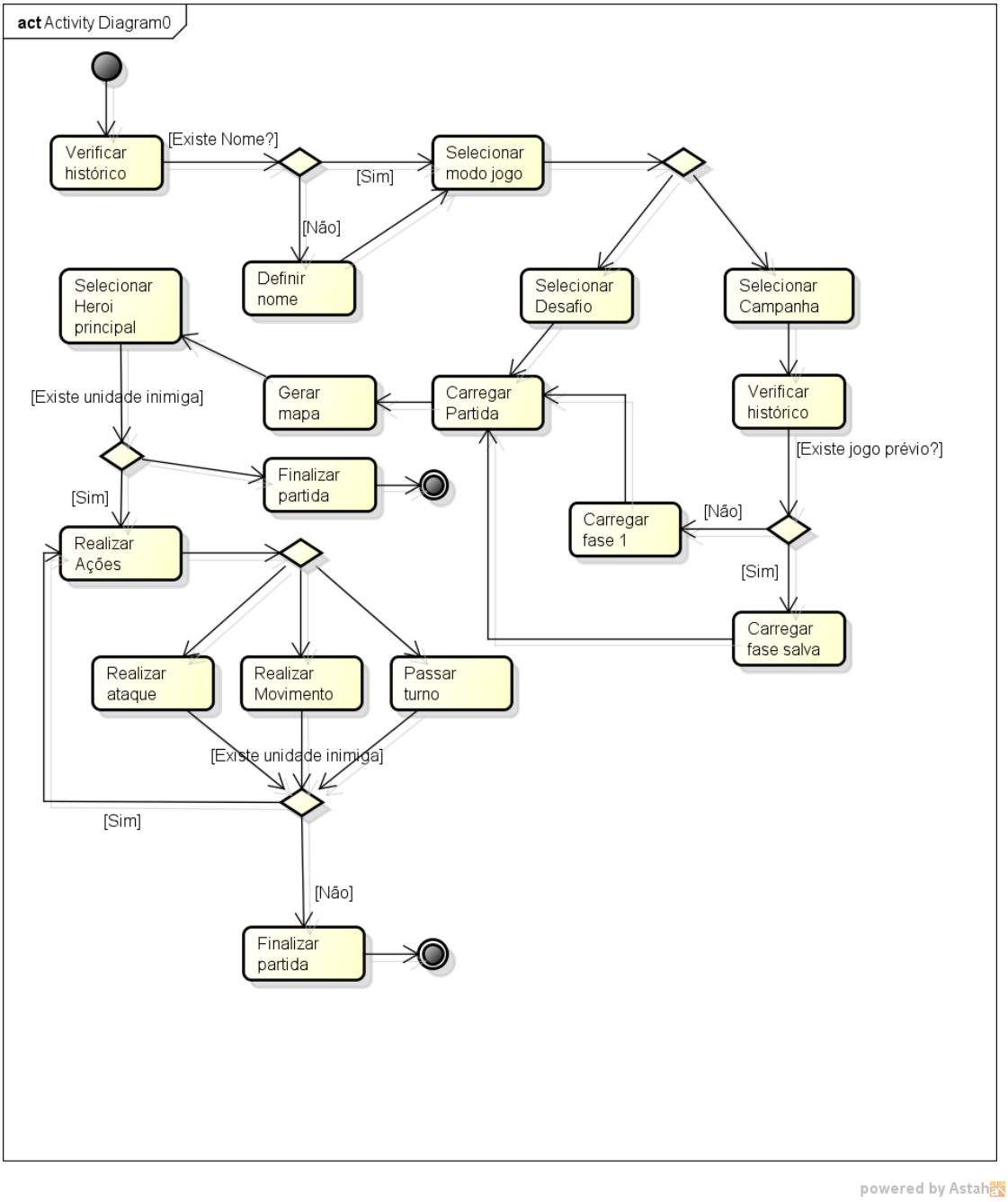

Figura 7 – Diagrama de Atividades

Finalmente, o ultimo passo que a metodologia descreve é o desenvolvimento por funcionalidade, de forma que todos os pontos previamente definidos sejam implementados pelos desenvolvedores. Essa fase do projeto será desenvolvida durante o Trabalho Final de Graduação 2.

6. Considerações Finais

Após realizar os estudos sobre as características do HTML5 e suas peculiaridades, foi constatado que a linguagem evoluiu consideravelmente com o tempo e que o ponto em que a mesma se encontra atualmente é repleto de possibilidades. A curva de aprendizado do desenvolvimento de jogos é curta, existindo uma grande variedade de guias e tutorias espelhados pela internet, e a comunidade em torno disso é grande.

O estudo realizada sobre a linguagem permitiu o desenvolvimento de um primeiro protótipo funcional para testes utilizando movimentação de imagens no Canvas, que foi bastante útil para entender os desafios que estarão presentes no desenvolvimento real do jogo. A simplicidade da API é um ponto favorável para iniciantes em desenvolvimento web, podendo ser facilmente entendível com um pouco de esforço.

Além disso, por ser um conteúdo atual e em constante desenvolvimento, existe uma grande quantidade de literatura de qualidade disponível, facilitando ainda mais o aprendizado e o desenvolvimento utilizando as tecnologias estudadas.

Por fim, chegou-se a conclusão que o desenvolvimento de um jogo utilizando HTML5 é plenamente possível, e com os resultados já obtidos com prototipagem pode-se esperar por um resultado final de qualidade, com um bom jogo sendo desenvolvido na segunda parte desse trabalho.

7. Referências Bibliográficas

Jr, J. M. da S. e Firmino and E. C. M. (2010) "Desenvolvimento de Jogos em HTML5", Universidade do Estado do Amazonas.

Martins, G. M. and Marchi, K. R. da C. (2013) "Análise Comparativa dos Recursos e Diferenças das Tecnologias de Programação HTML5 e HTML4", Universidade paranaense (Unipar).

Henriques, A., Vargas, M., Auad, T. and Knop, I. (2011) "O navegador como uma plataforma de jogos: Uma experiência extracurricular para desenvolvimento de software", Centro de Ensino Superior de Juiz de Fora.

Morais, S. N. and Borges, M. A. F. (2012) "Jogo educativo sobre Ecotoxicologia em HTML5", Universidade Estadual de Campinas (UNICAMP).

Hey, D. F. (2010) "Estudo da Viabilidade do HTML5 para desenvolvimento Web", Universidade Estadual de Maringá.

Ferreira, E. and Eis, D. (2013) "HTML5 – Curso W3C Escritório Brasil".

Musciano, C. and Kennedy, B. (2006) "HTML & XHTML: The Definitive Guide", sexta edição, editor O'Reilly Media Inc.

Ikeno, S. K. and Marchi, R. da C. (2013) "Análise da Nova Linguagem HTML5 para o Desenvolvimento Web", Universidade Paranaense (Unipar).

Junior, L. A. Z. and Vidal, A. G. da R. (2005) "Construção de sistemas de informação baseados na tecnologia Web".

A História dos Videogames, disponível em http://outerspace.terra.com.br/retrospace/materias/consoles/historiadosconsoles1.htm. Acessado em: Abr/2014.

Kalning, K. (2008) "The anatomy of the first vídeo game", disponível em http://www.nbcnews.com/id/27328345/ns/technology_and_science-games/t/anatomy-first-video-game/#.U4evefldW_g. Acessado em: Mar/2014.

Paula, R. (2013) "HTML5", disponível em https://developer.mozilla.org/pt-BR/docs/HTML/HTML5. Acessado em: Mai/2014.

"Padrões de código & melhores práticas em Front-end", disponível em http://andrecomws.com/lab/code-standards/. Acessado em: Mar/2014.

Case, Thiago "Javascript", disponível em https://developer.mozilla.org/pt-BR/docs/JavaScript. Acessado em: Mar /2014.

Fulton, S. and Fulton, J. (2013) "HTML5 Canvas", disponível em http://books.google.com.br/books?hl=pt-

BR&lr=&id=zLUyKvtdCQwC&oi=fnd&pg=PR2&dq=html5+canvas&ots=Haqj2XRQ1o&sig=UhXEkZKhHReZkTvRtQNnUHdw4h0#v=onepage&q&f=false
Acessado em: Abr/2014.

8. Apêndice – Descritivo dos Casos de uso

Identificador:	UC01
Caso de Uso	Iniciar Partida
Atores	Usuário
Pré Condições	
Pós Condições	Sistema verifica se existe nome cadastrado
Fluxo Principal	Acessar site do jogo
Exceções	Site fora do Ar
Regras de Negócio	O Usuário deve utilizar um dos navegadores compatíveis com HTML5

Identificador:	UC02
Caso de Uso	Selecionar nome
Atores	Usuário
Pré Condições	Ter acessado o jogo
Pós Condições	Sistema apresenta os modos de jogo disponíveis
Fluxo Principal	Sistema verifica se existe um nome cadastrado. Se nome existir, vai para UC03 Se nome não existir, sistema solicita ao usuário que informe.
Exceções	Caso usuário não insira o nome, é impossível prosseguir. Caso o nome exista, mas por algum motivo não possa ser resgatado, solicita um novo nome.
Regras de Negócio	É impossível selecionar um dos modos de jogo sem possuir o nome.

Identificador:	UC03
Caso de Uso	Selecionar modo de jogo
Atores	Usuário
Pré Condições	Possuir um nome associado.
Pós Condições	Sistema carrega os mapas correspondentes.
Fluxo Principal	Sistema apresenta os modos de jogo disponíveis. Usuário seleciona um deles. Caso modo selecionado seja campanha, sistema verifica se já existem jogos anteriores. Caso existam jogos anteriores, carrega a ultima fase não vencida. Caso não existam, carrega a primeira fase.
Exceções	Existe um jogo salvo, mas é impossível carregar o mesmo, então carrega a primeira fase.
Regras de Negócio	

Identificador:	UC04
Caso de Uso	Iniciar Partida
Atores	Usuário

Pré Condições	Sistema carregar mapa
Pós Condições	Sistema carregar heróis oponentes
Fluxo Principal	Usuário seleciona herói inicial Usuário posiciona herói inicial
Exceções	Se usuário não selecionar nenhum herói, o jogo não irá iniciar.
Regras de Negócio	Sistema deve selecionar um herói com características correspondentes ao herói do usuário. Sistema deve posicionar o herói adversário o mais longe possível do herói pertencente ao usuário.

Identificador:	UC05
Caso de Uso	Recrutar Herói
Atores	Usuário
Pré Condições	Possuir um herói principal Possuir ouro suficiente
Pós Condições	Sistema deve apresentar o herói escolhido pelo usuário no local escolhido por ele.
Fluxo Principal	Usuário seleciona herói. Sistema verifica se usuário possui ouro suficiente. Se o ouro for suficiente, usuário seleciona um local disponível para posicionar herói. Se ouro for insuficiente, sistema gera um alerta.
Exceções	Caso selecione um herói, possua ouro disponível mas não o posicione no mapa, o sistema irá posicionar automaticamente o herói no primeiro local disponível no mapa
Regras de Negócio	Sistema deve garantir a remoção do ouro utilizado do usuário. Sistema deve garantir que herói recrutado entre em jogo apenas uma vez, sem duplicar. Sistema deve garantir que usuário posicione herói apenas em locais disponíveis no mapa

Identificador:	UC06
Caso de Uso	Realizar Ações
Atores	Usuário
Pré Condições	Possuir herói. Ser o jogador do turno.
Pós Condições	Sistema deve realizar a ação escolhida.
Fluxo Principal	Usuário seleciona herói para realizar ação. Usuário seleciona ação. Sistema verifica se o herói selecionado já realizou aquela ação esse turno. Caso não tenha realizado, sistema libera ação. Caso tenha realizado, sistema gera um alerta.
Exceções	Usuário seleciona herói, mas ele já realizou todas as ações disponíveis. Nesse caso, sistema gera alerta.
Regras de Negócio	Sistema deve garantir a confiabilidade do usuário no jogo, não bloqueando ou permitindo ações de maneira equivocada.

Identificador:	UC07
Caso de Uso	Movimentar Herói
Atores	Usuário
Pré Condições	Não ter realizado a ação de movimentar herói nesse turno.
Pós Condições	Sistema irá realocar o herói na posição selecionada, caso a mesma seja valida.
Fluxo Principal	Após o UC06, usuário seleciona ação “Movimentar herói”. Sistema apresenta no mapa o alcance máximo que o herói pode alcançar. Usuário seleciona um dos sqm’s disponíveis. Sistema gera a animação de movimentação.
Exceções	Usuário seleciona um sqm inválido, nesse caso sistema gera um alerta.
Regras de Negócio	

Identificador:	UC08
Caso de Uso	Passar Turno
Atores	Usuário
Pré Condições	
Pós Condições	Sistema encerra o turno do usuário e começa o seu.
Fluxo Principal	Usuário seleciona ação “Passar turno”. Sistema encerra o turno do mesmo.
Exceções	
Regras de Negócio	Cada turno deve ser alternado usuário x sistema, e o sistema deve garantir esse fluxo contínuo.

Identificador:	UC09
Caso de Uso	Realizar Ataque
Atores	Usuário
Pré Condições	Herói não ter realizado a ação de realizar ataque nesse turno.
Pós Condições	Sistema deve realizar os cálculos de dano sofrido pelo atacado, e caso herói atacado seja derrotado, removê-lo do mapa e somar ao ouro do usuário atacante a quantia equivalente.
Fluxo Principal	Usuário seleciona ação de realizar ataque. Sistema apresenta os alvos dentro do alcance do ataque. Usuário confirma o alvo escolhido. Sistema verifica dano e derrotas.
Exceções	Se não existir nenhum alvo ao alcance, é impossível atacar.
Regras de Negócio	Sistema deve garantir que somente alvos válidos sejam atacados, ou seja, somente alvos ao alcance do ataque ou ao alcance de ataque somado ao movimento do herói atacante.