UNIVERSIDADE DO SUL DE SANTA CATARINA

JHONATTAN AMORIM DE OLIVEIRA

ROGÉRIO RIVERA TORRES FILHO

PROCESSO DE DESENVOLVIMENTO DE UM JOGO DIGITAL PARA DISPOSITIVOS MÓVEIS

Palhoça

2014

JHONATTAN AMORIM DE OLIVEIRA
ROGÉRIO RIVERA TORRES FILHO

PROCESSO DE DESENVOLVIMENTO DE UM JOGO DIGITAL PARA DISPOSITIVOS MÓVEIS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade do Sul de Santa Catarina, como requisito parcial à obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Saulo Popov Zambiasi.

Palhoça
2014

JHONATTAN AMORIM DE OLIVEIRA
ROGÉRIO RIVERA TORRES FILHO

PROCESSO DE DESENVOLVIMENTO DE UM JOGO DIGITAL PARA DISPOSITIVOS MÓVEIS

Este Trabalho de Conclusão de Curso foi julgado adequado à obtenção do título de Bacharel em Ciência da Computação e aprovado em sua forma final pelo Curso de Graduação em Ciência da Computação da Universidade do Sul de Santa Catarina.

Palhoça, 18 de junho de 2014.

Prof. Dr. Saulo Popov Zambiasi
Universidade do Sul de Santa Catarina

Prof. Ms. Patricia Leandra Barrufi Pinheiro
Universidade do Sul de Santa Catarina

Prof. Vinicius Faria Culmant Ramos
Universidade do Sul de Santa Catarina

AGRADECIMENTOS

Jhonattan agradece a:

A toda a minha família, meu pai Nilton Carlos de Oliveira, à minha mãe Laurete Amorim de Oliveira e a meu irmão Jefferson Amorim de Oliveira pelo apoio e carinho durante todo a vida.

Ao professor Saulo Popov Zambiasi, nosso orientador, pela ajuda, contribuição, e orientação ao longo de todo o TCC que foram essenciais.

A todos os professores que também fizeram parte de nossa trajetória na universidade e que sem eles não estaríamos aqui.

Aos nossos colegas de turma, André Meurer, Eduardo Jordano, Pedro Padro, Silmara Horstmann e diversos outros que estiveram com a gente ao longo da universidade.

E a todos que de certa forma contribuíram em nossas vidas e na realização deste trabalho.

Rogério agradece a:

Gostaria de agradecer primeiramente ao professor orientador Saulo Popov Zambiasi pelas orientações e os conhecimentos passados, que foram o principal fator para o sucesso do presente trabalho.

Aos mestres que estiveram comigo durante esta caminhada deixando muito mais do que conhecimentos técnicos, deixando filosofias e lições de vida.

A minha família por todo o investimento e apoio na minha carreira.

RESUMO

Os avanços tecnológicos da indústria de jogos digitais tem impulsionado um crescente número de novos desenvolvedores e uma ampla gama de novos jogos surgindo a cada dia, tanto a nível industrial, quanto a nível dos *Indiegames* (Desenvolvedores Independentes). O processo para o desenvolvimento desses jogos varia de empresa para empresa. Entretanto, para um desenvolvedor que necessita iniciar nesse mercado, há a questão de saber por onde começar ou quais as diferenças no desenvolvimento de um jogo digital e de um software tradicional. Em vista disso, a proposta desse trabalho foi apresentar o processo de desenvolvimento de um jogo com o enfoque em sistemas operacionais para dispositivos móveis, mais especificamente para a plataforma Android. Para tal, um jogo foi desenvolvido desde o processo de criação do *Game Design*, gerando o *Game Design Document* (GDD), modelagem UML, testes e verificação do resultado. O jogo desenvolvido possui uma interface 2D e segue o estilo de *Role-Playing Games* (RPG). Foi utilizada a linguagem de programação Java com a biblioteca LibGDX e ambiente de desenvolvimento *Android Development Tool* (ADT). O resultado do jogo, versão beta, passou por um período de testes ao público geral, disponibilizado para download através de um blog. Pela avaliação geral, o jogo se mostrou muito bem nos resultados em sua totalidade, recebendo sua aprovação positiva pelo público.

Palavras-chave: Jogo, Dispositivos Móveis, Android ,Processo de Desenvolvimento, LibGDX, Java, *Game Design Document*.

ABSTRACT

The technological advances of the digital games industry has driven a growing number of new developers and a broad range of new games coming up every day , both industrial as the level of IndieGames (Independent Developers) . The process for developing these games varies from company to company . However, for a developer who needs to start in this market , there is the question of where to start or what the differences in the development of a digital game and a traditional software . In view of this , the purpose of this study was to present the development process of a game with the focus on operating systems for mobile devices , more specifically for the Android platform . For such a game was developed from the process of creation of Game Design , Game Design Document generating (GDD) , UML modeling, testing and verification of income. The game has developed a 2D interface and follows the style of Role- Playing Games (RPG) . The Java programming language with LibGDX library and development environment Android Development Tool (ADT) was used . The outcome of the game , beta, went through a period of testing to the general public, available for download through a blog . For the overall rating , the game turned out very well in the results at all, getting your positive approval by the public.

Keywords: Game, Mobile Devices, Android, Development Process, LibGDX, Java, Game Design Document.

.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

Com o avanço tecnológico cada vez mais rápido, ligado ao crescente mercado consumidor de dispositivos móveis e da indústria de jogos para os mesmos estar em alta, a procura por profissionais especializados cresce diariamente. Segundo a ABRAGAMES (2008) (Associação Brasileira de Desenvolvedores de Jogos Digitais) cerca de 560 profissionais em 42 empresas trabalhavam na área de jogos em 2008 com rápido crescimento previsto e procura por profissionais principalmente na área de artistas gráficos e programadores.

Neste contexto, este trabalho tem como foco compreender a estrutura e processo de desenvolvimentos de jogos para dispositivos móveis baseados na plataforma Android, o qual possui desenvolvimento baseado na linguagem Java e que de acordo com Sampaio (2008). Tais jogos têm grande aceitação no mercado se tratando de jogos casuais, pois "só em 2007, esse mercado rendeu mais de US$ 2,25 bilhões e pelo menos 200 milhões de pessoas jogam títulos casuais através da internet".

No Brasil de acordo com dados da ANATEL – Agência Nacional de Telecomunicações (ANATEL, 2013) há acima de 262 milhões de acessos móveis, de acordo com dados de janeiro de 2013. Logo há um grande número de usuários de dispositivos móveis no Brasil, e assim sendo o desenvolvimento de aplicativos, incluindo-se os jogos, para tais dispositivos se torna um grande mercado, para profissionais que desejem trabalhar nesta área.

Desta forma esse trabalho trata então das etapas da criação de um jogo, desde a concepção da ideia, passando por suas etapas de criação dos personagens, contexto, cenário, história até sua fase de animação, sonorização e, por fim, sua parte de codificação assim como uma breve introdução sobre a história dos jogos eletrônicos desde o seu princípio até a atual geração e a dita "próxima geração", sobre o sistema operacional Android, jogos específicos para dispositivos móveis assim como os perfis de jogadores que procuram jogos para estes tipos de aparelhos, sejam eles *smartfones* ou *tablets*.

Seja para a criação de aplicativos a jogos ou projetos de hardware, o mercado dos dispositivos móveis hoje é um dos que mais evoluem e ganham espaço na área de Tecnologia da Informação.

Logo os autores pretendem assim com esta monografia mostrar o caminho para o desenvolvimento de jogos digitais para dispositivos móveis, como base para se começar nesta área de desenvolvimento, pois assim como Morpheus no filme Matrix disse “Eu só posso lhe mostrar a porta. Você tem que atravessá-la”.

1.1 PROBLEMA

Nos últimos anos a indústria da tecnologia móvel vem se tornando cada vez maior, segundo PERREIRA e SILVA (2009, p. 1) “o celular é o produto de consumo mais utilizado no mundo, sendo a quantidade existente correspondente à metade da população mundial(3,3 bilhões – 2007) e acredita-se que até o final de 2013 este número chegará a 5,6 bilhões”, assim dispositivos com cada vez mais recursos entram no mercado a cada dia, com melhores softwares e hardwares, o que possibilita a desenvolvedores criarem seus aplicativos com melhor qualidade.

O grande crescimento dessa indústria de dispositivos móveis ocorre pelo fato de que com a evolução da tecnologia presente neles suas características agradam cada vez mais as pessoas, que segundo SAMPAIO e RODRIGUES (2012, p. 17) fazem de aplicações móveis um grande sucesso, são “intuitividade, interatividade, conectividade, integração, visual sofisticado, sons, vídeos e velocidade”.

Dentre os diversos aplicativos desenvolvidos para estes dispositivos móveis estão os jogos que se beneficiam ainda mais de todo o avanço tecnológico, com isso essa área de desenvolvimento vem crescendo no mesmo ritmo dos dispositivos, de acordo com SOLLITTO (2012) “Em 2010, o mercado mundial de games movimentou US$ 56 bilhões de dólares”, e saber como desenvolver estes jogos se torna cada vez mais importante para profissionais que queiram entrar neste mercado.

Com isso, ter conhecimento da estrutura, dos processos e das etapas que compreendem o desenvolvimento de jogos para dispositivos móveis, se mostra de grande

importância para aqueles que desejam estar nesta fatia do mercado de desenvolvimento, por toda complexidade por trás desses aplicativos. Logo saber como tais estágios de desenvolvimento funcionam desde sua estrutura inicial de apenas uma ideia, passando pelas suas etapas de elaboração, a escolha da linguagem, da plataforma, da biblioteca gráfica, são decisivos para o sucesso do aplicativo.

Dentre os dispositivos móveis no mercado a possibilidade de desenvolvimento em Java com a plataforma Android, acaba se tornando a escolha mais adequada as características desta monografia, pelo tempo, recursos disponíveis para programação e com boa abrangência do mercado.

Utilizando-se de uma biblioteca gráfica para auxiliar no desenvolvimento, permite-se ao desenvolvedor focar nos outros aspectos relacionados ao jogo.

1.2 OBJETIVOS

Os objetivos desta monografia são divididos em:

- Objetivos Gerais
- Objetivos Específicos

1.2.1 Objetivos Gerais

O objetivo deste trabalho é o desenvolvimento um jogo digital para dispositivos móveis, contextualizando e apresentando o processo de desenvolvimento deste, desde seu Game Design Documento (GDD) até sua implementação.

1.2.2 Objetivos Específicos

Como objetivos específicos para nortear este trabalho, tem-se:

1. Obter referências bibliográficas para suportar os conteúdos envolvidos no desenvolvimento de jogos digitais;
2. Concepção de um *Game Design Document*(GDD);
3. Criação dos recursos audiovisuais definidos no GDD;
4. Modelagem da implementação do sistema em *Universal Modeling Language* (UML);
5. Implementação de um protótipo do jogo;
6. Testes e avaliação;

1.3 JUSTIFICATIVA

Atualmente com o grande crescimento da área de desenvolvimento para dispositivos móveis, ligado ao atual mercado de jogos, cujo segundo ABRAGAMES (2008) o produto nacional no setor de jogos é de R$ 87.5 milhões em 2008. E com participação de cerca de 15% vindo de celulares, faz com que profissionais desejem trabalhar nesta área, seja pela possibilidade de trabalhar com algo que gostem, pelo dinheiro na área, ou pelo desafio de se trabalhar com algo diferente.

Segundo Novak (2010), metade do público norte-americano consome games eletrônicos. Em 2006 a *Entertainment Software Association* realizou uma pesquisa e constatou que 69% dos chefes de família norte-americanos jogam games eletrônicos regularmente e em média 6,5 horas por semana, jogando pelo menos três gêneros de games.

Além disso o jogo eletrônico traz como um de seus principais benefícios para a pessoa a interação social, que possibilita o jogador de poder conversar e interagir com os

outros jogadores. Novak (2010, p. 43) ainda relata que, “quando há mais de uma pessoa participando de um game, jogadores podem se sentir motivados a interagir socialmente com seus adversários ou colegas de equipe.”

Logo a ideia, deste trabalho em se se desenvolver um jogo para apresentar o processo de desenvolvimento de um jogo para dispositivos móveis Android, trazendo os conceitos e estruturas básicos para começar a desenvolver jogos.

Apresentando todo o processo desde a concepção do conceito, passando por todo processo de desenvolvimento do GDD, desenvolvimento de sua modelagem, ate sua inteira implementação. Assim como para Mendes (2002, p. 6), “o processo de desenvolvimento de um sistema de software vai desde a concepção do sistema, quando os requisitos são elicitados e analisados, até a sua concreta implementação”, acredita-se que todo processo é de igual importância para o desenvolvimento de um jogo.

A escolha do sistema Android vem de sua facilidade de desenvolvimento em linguagem Java por parte dos autores e a utilização de uma biblioteca gráfica para auxiliar no desenvolvimento e agilizar o processo de criação do jogo, retirando a necessidade de se criar algo do zero, assim como a definição de um jogo com gráficos 2d por se tratar de algo mais simples em se tratando da criação das *spritesheets*[1], voltando-se a outras funcionalidades mais importantes do mesmo.

1.4 ESTRUTURA DA MONOGRAFIA

Capítulo 1: Introdução – O presente capítulo apresenta a introdução dividida entre justificativa, objetivos gerais e específicos, a problemática e a definição da estrutura do trabalho.

Capítulo 2: Revisão bibliográfica – Apresenta um estudo sobre os conceitos necessários para o desenvolvimento de um jogo específico para dispositivos móveis, como Game Design, motores gráficos, apresentando um conteúdo relevante existente, apresentando

[1] Spritesheet: são o conjunto de sprites, que são objetos gráficos, neste caso objetos do game em diversas posições que ao serem colocados em sequência na tela durante o jogo fazem a animação dos mesmos.

seus pontos positivos e negativos e assim justificando as escolhas técnicas que os autores utilizaram.

Capítulo 3: Método – Neste capítulo é apresentado a metodologia a ser utilizada no projeto, assim como classificações do projeto quanto a natureza da pesquisa, quanto a abordagem da pesquisa, quanto aos objetivos da pesquisa e quanto aos procedimentos abordados.

Capítulo 4: GDD do Projeto – É apresentado o GDD referente ao projeto do game a ser desenvolvido durante a monografia.

Capítulo 5: Modelagem do Sistema – Apresenta a modelagem do projeto do game a ser desenvolvido durante a monografia.

Capítulo 6: Implementação / Testes / Avaliação – Neste capítulo é abordado a implementação, os testes e a avaliação do game.

Capítulo 7: Considerações Finais – É apresentado as considerações finais sobre o projeto do game.

Capítulo 8: Conclusões e trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

A revisão bibliográfica aborda os temas fundamentais do desenvolvimento desta monografia, temas estes relacionados ao processo de desenvolvimento de um jogo digital para dispositivos móveis. Inicialmente será apresentado uma breve história sobre a evolução dos jogos, em sequência é apresentado a parte conceitual do planejamento do processo das fases do desenvolvimento de um jogo digital que começa pelo seu conceito ate a fase de pós-produção. Após isto são apresentados tópicos relacionados aos conceitos de UML, linguagem de programação, estilo do jogo e plataforma de destino do jogo.

2.1 HISTÓRIA DOS VIDEOGAMES

Para se chegar no patamar aonde os jogos estão hoje em dia muito fatos ocorreram na história dos videogames, é apresentado alguns deles a seguir, com uma breve história sobre os videogames e jogos que deram origem ao universo dos games que conhecemos hoje.

2.1.1 O Fliperama

O primeiro contato do público com os games eletrônicos não se deu com os consoles de games domésticos ou com os computadores pessoais, mas nas casas de games ou de fliperama.(NOVAK, 2010).

Porém, segundo Novak (2010, p.4), os primeiros games eletrônicos não foram jogados em casa ou mesmo nas casas de fliperama.[…] os primeiros passos foram dados em departamentos de pesquisa de universidades e instalações militares.

2.1.1.1 Pré-Fliperama

A partir da década de 1950, dois segmentos do setor de games se desenvolveram em paralelo.

Um desses segmentos começou em 1951, quando Marty Bromley, que cuidava das salas de games em bases militares no Havaí […] lançou o SEGA[…]. (NOVAK, 2010).

"Marty estava na base de Pearl Harbor quando a mesma foi bombardeada pelos japoneses.[…] porém ele nunca guardou ressentimentos para com os japoneses. Em 1952,os caça-níqueis foram proibidos nos Estados Unidos, então Marty Bromley viu no mercado Japonês uma oportunidade, ele comprou máquinas caça-níquel do governo e criou uma empresa, denominada *Service Games*(SEGA),que importava-as para o Japão." (IGN, 2010, tradução nossa).

Em seu livro, Novak (2010) diz que o outro segmento do setor de games eletrônicos começou com os games para computadores de grande porte desenvolvidos por professores e alunos em universidade, seja para aprimorar suas habilidades de programação seja como forma de entretenimento[…].

2.1.1.2 As casas de fliperama

As casas de fliperama eram espaços localizados geralmente próximo as escolas e parques de diversões, atraindo crianças e adolescentes que se reuniam para jogar os jogos que la haviam.

> Vários games são considerados marcos fundamentais dessa época. Embora limitados pela tecnologia disponível, esses games eram inovadores, inspirando novas tendências de conteúdo, gêneros e jogabilidade, bem como técnicas de desenvolvimento que jamais haviam sido consideraras.[…] Eles já apontavam para um futuro em que os games eletrônicos seriam um meio de entretenimento em massa. (Novak, 2010, p.6).

A seguir, uma introdução a alguns games da era dos fliperamas:

1. *Computer Space* (Figura 1): Criado em 1971 por Nolan Bushnell, que mais tarde fundaria a Atari. É uma versão do jogo *Spacewar*!, criado em 1961 por Steve Russel, estudante do MIT(Instituto de Tecnologia de Massachusetts).

Figura 1. Propaganda do *Computer Space*

Fonte: http://www.cedmagic.com

2. Pong: O primeiro game eletrônico memorável surgiu em 1958, quando Willy Higinbotham, dos Laboratórios Nacionais Brookhaven em Nova York, mostrou seu game simulando uma partida de tênis de mesa (*Tennis for Two*), na Figura 2 em um computador analógico.(NOVAK, 2010 p.8). Quase uma década depois, a Atari e a Magnavox brigavam judicialmente pela licença do jogo.

Figura 2. Esquerda *Tennis for Two*, direita PONG

Fontes: www.evilmadscientist.com e http://pongmuseum.com

3. Pac-Man: Em 1980, a Namco lançou o Pac-Man (Figura 3), que atraiu um mercado muito mais diversificado, […] o game era controlado apenas por um *joystick* multidirecional. Em vez de destruir espaçonaves, o Pac-Man comia pílulas de energia que lhe permitiam engolir seus inimigos por um tempo. (NOVAK,2010 p.10). Segundo NOVAK (2010), Com mais de 300 mil unidades vendidas em todo o mundo, tornou-se o game para máquinas eletrônicas mais popular de todos os tempos. Quando conseguiam concluir uma fase, os jogadores passavam para a seguinte, que continha o mesmo labirinto, mas com um nível de dificuldade maior.

Figura 3. Primeira versão de Pac-Man

Fonte: http://www.boom973.com

4. Donkey Kong (Figura 4): Em 1977, Shigeru Miyamoto[…] queria criar algo diferente. O resultado foi o Donkey Kong.[…] Em Donkey Kong, um gorila rapta a namorada de seu tratador e foge. O Jogador assume o papel do tratador, chamado *JumpMan* (que hoje é conhecido como Mario), que se transforma no herói da história.[…] A Nintando entrou de maneira espetacular no mercado norte-americano com o Donkey Kong. (NOVAK,2010 p.12).

Figura 4. Primeiro Donkey Kong

Fonte: http://www.donkey-kong.info

2.1.2 Os consoles, do fliperama para o Lar.

Os fliperamas sempre fizeram um grande sucesso, porém o custo de uma máquina de fliperama era demasiadamente caro, sendo não viável aos consumidores e sim apenas aos donos das casas de fliperama e aos mais ricos. Logo então viu-se que a portabilidade dessas máquinas para as casas e a diminuição do seu tamanho poderiam gerar uma grande alta no mercado de jogos.

"Logo se tornou evidente que a venda direta ao consumidor poderia expandir tremendamente o setor, o que fez com que os videogames comerciais migrassem para as residências na forma de consoles de videogame acessíveis." (NOVAK,2010 p.14).

> A migração das casas de games eletrônicos para os lares foi o momento mais significativo da história do desenvolvimento dos games. Os consoles e PCs permitiram que os games se integrassem plenamente ao nosso consumo de produtos de mídia, facilitando o uso cotidiano. Acredito que isso incentivou o aumento da diversidade nos tipos de jogos desenvolvidos. Drew Davidson (Diretor do centro de tecnologia de entretenimento da universidade Carnegie Mellon).(Novak, 2010, p.14).

A seguir uma breve introdução aos consoles mais relevantes do início da história dos videogames:

1. Atari VCS/2600 (Figura 5): Lançado nos Estados Unidos em 1977, o atari não foi o primeiro console lançado, porém era mais acessível e ganhou o mercado. Segundo Novak (2010), A Atari manteve o preço do hardware baixo, garantindo a maior parte de sua receita com os games que desenvolvia para o console.

Figura 5. Atari VCS 2600

Fonte: atariage.com

2. Mattel Intellivision (Figura 6): Lançado pela Mattel em 1980, era mais caro que o Atari, custando em média U$300,00. A Atari começou a enfrentar alguma concorrência dois anos após o lançamento do VCS 2600, quando a Mattel lançou um sistema de console alegadamente superior(e mais caro) conhecido como Intellivision. Em vez de um *joystick*, o Intellivision era equipado com um controlador "inteligente" que consistia em um teclado numérico e um disco de movimento[…].(NOVAK,2010 p.16).

Figura 6. Mattel Intellivision

Fonte: www.scottdecker.com

3. ColecoVision (Figura 7): Lançado em 1982, representando a entrada da Coleco no mercado de consoles de jogos. O Donkey Kong, jogo altamente popular da Nintendo para fliperama, era incluído em cada ColecoVision.(NOVAK,2010 p.17).

Figura 7. ColecoVision

Fonte: www.old-computers.com

4. NES: O nintendo Entertainment System (Figura 8) foi lançado em 1985 pela Nintendo. Segundo Novak (2010), a entrada da Nintendo no setor de consoles[…] deu nova vida ao mercado de games domésticos, mas também ajudou a apressar a extinção do setor de fliperamas.

> O sucesso do NES no mercado foi tão grande que a Atari, até então líder do setor, criou o Tengen, uma subsidiária voltada exclusivamente para o desenvolvimento de games para o NES. Logo depois, a Tengen descobriu uma maneira de contornar o "chip´de travamento" da Nintendo. (Novak,2010 p19).

Figura 8. NES

Fonte: compauta.com.br

5. Master System, Genesis, Saturn e Dreamcast (Figura 9): Após a introdução bem-sucedida do NES pela Nintendo, a Sega começou a lançar uma série de sistemas[...], que incluíram o Master System, Genesis, Saturn e Dreamcast.(NOVAK,2010 p21).

Figura 9. Master System

Fonte: gamehall.uol.com.br

2.1.3 Sony x Microsoft x Nintendo

Nas últimas três gerações de consoles, as empresas Sony, Microsoft e a Nintendo foram definitivamente as que dominaram o mercado, porém a "guerra" entre as três empresas começou mais precisamente no ano 2000.

> A disputa do mercado de consoles pelas "três grandes" começou em 2000 com o lançamento, pela Sony, do PlayStation 2 (PS2), que logo se tornou o console mais vendido da história com mais de 100 milhões de unidades comercializadas até 2006. O GameCube da Nintendo e o Xbox da Microsoft entraram nessa nova guerra dos consoles em 2001 e 2002. O GameCube voltava-se para um mercado mais jovem[…] enquanto a ênfase do Xbox em desempenho e recursos atraía jogadores mais velhos e exigentes.(NOVAK, 2010 p.24).

Essa "guerra" pela preferência dos jogadores, no mercado de consoles, entre as empresas continua forte, em sua atual geração de consoles, de acordo com Novak(2010), a Sony com o Playstation 3, a Microsoft com Xbox 360 e a Nintendo com o Wii, agregam novas funcionalidades, como reprodução em formato Blue-Ray, acesso à internet possibilitando jogar on-line, baixar jogos, músicas, filmes, etc. Com isso, as empresas continuam inovando e tornando cada vez mais competitivo esse mercado.

2.1.4 O início dos jogos para PCs

Segundo Novak (2010), EM meados da década de 1970, outro segmento da indústria de games começou a entrar em uma nova era. Os computadores pessoais trouxeram para dentro da casa da população, tecnologias que eram de exclusividade dos mais ricos, pesquisadores ou programadores, disponibilizando para todos os games para computadores, que antes eram apenas jogados e desenvolvidos por estudantes nas universidades.

> Games que eram desenvolvidos anteriormente como passatempo por estudantes universitários agora eram adaptados para computadores pessoais, e, com isso, o grande público podia participar da diversão. Já havia games de fliperama sendo transportados para sistemas de videogames domésticas e roubando parte do mercado das casas de games eletrônicos. (NOVAK, 2010 p.24).

A expansão do uso de computadores domésticos também contribuiu para o consequente declínio do negócio de casas de fliperama, além de constituir uma ameaça para o segmento de consoles de videogame. (NOVAK, 2010 p.24).

Dos primeiros games para computadores pessoais, os que mais faziam sucesso eram baseados apenas em textos, com nenhuma ou muito pouca utilização de elementos gráficos.

2.2 FASES DE DESENVOLVIMENTO DE UM JOGO

O desenvolvimento de um jogo digital é um processo longo, demorado e que requer muita atenção aos detalhes, ele se apoia em documentos gerados nas fases iniciais do design do jogo, esses assemelhados segundo Schuytema (2008 p.10) “a uma planta baixa – podemos ter a matéria-prima e os técnicos habilidosos para construir uma casa, mas, sem um plano, o trabalho não pode continuar sem uma direção real”, ele possui certos aspectos e etapas que devem ser respeitadas.

Na maioria das vezes segundo Rabin (2012) “existem cinco fases no processo de desenvolvimento de um jogo: conceito, pré-produção, produção, pós-produção e pós-venda” porém, este trabalho citará apenas nas quatro primeiras.

2.2.1 Conceito

Uma fase de grande importância para o desenvolvimento do jogo, esta fase é responsável pela criação do conceito, da ideia geral do game, essa fase inicia quando uma ideia de game é criada ate o começo da fase de pré-produção. Para Novak(2010) “o objetivo do desenvolvimento do conceito é decidir em que consiste o game e transmitir essa ideia a outras pessoas por escrito”. É ao final desta fase que é criado o documento de conceito e que ainda segundo Novak os objetivos deste documento são: “identificar um público-alvo, avaliar os recursos da empresa e identificar um conceito que seja atraente aos desenvolvedores e tenha um mercado potencial”.

> Os conceitos de jogo normalmente não são frutos da imaginação de um designer. Em geral são decisões empresariais lógicas ou obvias baseadas em sucessos anteriores ou negócios. Por exemplo, com frequência acontece de má editora de jogos já possuir uma franquia de jogos de sucesso e desejar fazer uma sequência; ou ela negociou com um estúdio de cinema para fazer um novo jogo baseado num filme; ou detêm tecnologia, o motor de jogos, e deseja desenvolver um novo jogo usando o potencial da tecnologia existente. Ocasionalmente, novos conceitos originais vem junto, sejam projetados pelo pessoal de criação ou apresentados por um desenvolvedor externo (Rabin, 2012, p.804)

É nesta fase em que tudo começa, com o desenvolvimento do conceito do jogo e da criação do documento de conceito, os outros documentos pré implementação podem ser criados. Documentos que serviram para que quando o jogo comece a ser desenvolvido, todos saibam como deverá ser o resultado final do game.

2.2.1.1 Documento de Conceito

O documento de conceito é o produto final da fase de conceito no processo de desenvolvimento de um jogo digital, segundo Novak (2010 p.366),

> o documento de conceito ou documento de venda tem como finalidade de transmitir o objetivo e a finalidade do game proposto […] ajuda a avaliar se a ideia do game é viável, oportuna e factível. A finalidade desse documento é vender a ideia às fontes de financiamento, editoras ou outras instâncias decisórias na empresa.

Este documento deve possuir ainda de acordo com Novak(2010), os seguintes componentes:

- Premissa: ou conceito geral, é a ideia básica do game, endereçada ao jogador, descrevendo a atmosfera do game e seu diferencial, pode-se pensar como algo que possa ser colocado na parte da frente da embalagem do game, em cartazes ou próximo ao título.

- Motivação do jogador: basicamente é descrito quais as condições de vitória e derrota do game, como o jogador vence, o que motivara ele a chegar ao fim do jogo.

- Diferencial: a descrição do que torna seu game único, o que fara com que o público jogue o seu jogo em vez de outro da concorrência, aquilo que destaca seu jogo dos demais, o que fara ele ser desenvolvido, o que faz ele especial, recursos como estilo gráfico ou tecnologia avançada da *engine*.

- Público-alvo: aqui é descrito para que público ou que tipo de jogadores que terão mais probabilidade de joga o game, informações relacionando o público ao gênero e a faixa etária.

- Gênero: descrição do gênero do jogo, qual o gênero do jogo, isso relacionado ao estilo e modo de jogar o game, se será um gênero já consolidado, um híbrido ou algo totalmente novo.

- Classificação etária: descreva qual a faixa etária do jogo, relacionando ao porque da escolha.

- Plataforma de destino e requisitos de hardware: uma importante etapa, a escolha da plataforma para seu game, devera descrever qual será a plataforma dentre as várias do mercado, fliperama, consoles, dispositivos portáteis, computadores, em alguns casos será necessário um relacionamento com o fabricante do hardware da plataforma por serem plataformas proprietárias, indique aqui também se tem pretensão de adaptar o jogo a outra(s) plataformas, é interessante ressaltar requisitos mínimos e recomendados da plataforma escolhida para rodar o jogo, esta escolha é diretamente ligado a seu público-alvo.

- Licença: aqui indique caso seu jogo seja adaptado de uma propriedade sujeita a uma licença, se será necessário, ou fara algum tipo de exclusividade com licenciador, inclua a popularidade do mesmo e seu apelo no mercado, jogos de esporte geralmente devem incluir as licenças de logotipos, nomes, elementos de times.

- Analise competitiva: selecione títulos bem-sucedidos disponíveis atualmente no mercado e descreva como o seu poderá competir com cada um deles, escolha entre três a cinco e explique como competira, como ele se diferenciara e no que ele será melhor que os outros.

- Objetivos: descreva quais são as expectativas quanto a game, ao usuário, quais as sensações que deseja produzir em quem joga, vá além da ideia de diversão, descreva que tipo de emoção pretende fornecer a quem joga, se o jogar ira criar suas próprias histórias e personagens e descreva como pretende alcançar tais objetivos.

2.2.2 Pré-produção

É nesta fase do processo de desenvolvimento de um jogo digital que o jogo em si começa tomar forma, como visto em Schuytema (2008 p.12) "[…]Esse é um momento para discussões, *brainstormings*[2] e avaliação dos games concorrentes.[…] Nesse período os designers escrevem diferentes documentos[…]" e que ainda segundo Schuytema serviram como a planta baixa do projeto,é o coração e a alma do projeto.

Diferentes são os modelos de documentos e documentos gerados nesta fase, dependendo de cada autor e empresa. Porem todos apresentam como objeto final desta fase o *Game Design Document* (GDD) ou documento de design do game (DDG).

Segundo Novak (2010) é nesta fase que é desenvolvido o documento de proposta do game e se entra na fase de planejamento do desenvolvimento, nela é criado o guia de estilo da arte, o plano de produção, o GDD, sendo o GDD a soma do que se é obtido através do plano de proposta mais alguns outros elementos.

2.2.2.1 Documento de proposta

O documento de proposta do jogo digital é basicamente uma adição ao documento já criado de conceito, que segundo Novak (2010 p.370),

> a proposta do game é um complemento do documento de conceito, descrevendo mais detalhadamente todos os componentes do documento anterior. A finalidade da proposta é apresentar os detalhes do game a uma empresa ou possível parceiro que já esteja interessado na ideia[…] também pode ser usado para explicar detalhadamente o game a possíveis membros da equipe de arte antes que comecem a planeja seu desenvolvimento.

[2] Brainstorming: é uma técnica usada para gerar novas ideias de forma coletiva. (QUINN et al., 2011)

Ou seja, ele vem pra auxiliar na elaboração da ideia geral do jogo e ampliar o conhecimento do mesmo antes das próximas etapas e documentos. Novak (2010) ainda relata que produtores e diretores das equipes de arte, programação e design participam de seu desenvolvimento, e que um tratamento narrativo é dado abordando a premissa, elementos da história, história anterior, sinopse e descrição personagens, e que todas os tópicos abordados no documento de conceito devem constar neste novo documento além das seções descritas por Novak (2010, p. 370):

- Gancho: no gancho descreve-se o que atrairá o jogador para seu game e o que manterá nele, o que fara com que os jogadores o comprem, cite de três a cinco melhores características baseados em aspectos do game como modo de jogar, história, visual, áudio, etc.

- Modo de Jogar: neste tópico os elementos do modo de jogar, ou seja, a experiência que o jogador ira ter dentro do game, os desafios, o combate, os caminhos que poderá escolher durante a história, outras atividades como exploração, construção, solução de enigmas, problemas, ou a cooperação com outros jogadores em caso de *multiplayer*[3].

- Recursos On-line: é a parte, caso o game possua elementos on-line ou *multiplayers* on-line, onde se descreve como funcionaram em relação ao trabalho cooperativo, serviços on-line, serviços de busca de jogadores, nível de compatibilidade, modos de combate online jogador contra jogador(*PvP*, *Player vs. Player*).

- Tecnologia: descreve-se a utilização de tecnologias específicas, de hardware ou software, se ira utilizar tecnologias de terceiros licenciadas, como motores gráficos, reconhecimento de voz/movimento, etc.

- Características da Arte e do Áudio: é descrito as características de arte e áudio, com relação a licenciamento de músicas de terceiro para uso no game ou se criaram toda a

[3] Multiplayer: jogos multiplayer são jogos que permitem com que vários jogadores joguem juntos e ao mesmo tempo. (NOVAK, 2010)

parte de áudio, se pretende-se contratar um compositor/artista para criação de trilhas sonoras e arte, se utilizaram técnicas de captura de movimentos para tornar os movimentos dos personagens mais reais no processo de animação.

- Detalhes da produção: esta seção trata de alguns detalhes da etapa de desenvolvimento do game, da descrição equipe, uma estimativa de orçamento, de cronograma, datas previstas de conclusão, prazos intermediários de protótipos, betas, alfas, etc., todos não finais sujeitos a alterações futuras, servem apenas para mostrar qual o comprometimento previsto no projeto, se é algo pequeno ou um projeto de larga escala.

- História anterior: descrição da história do game antes do seu início propriamente dito, fatos anteriores ao game, um resumo para contextualizar a história do game.

- Sinopse da história: basicamente a sinopse da história do game, de forma narrativa conte a história, limite-se a ideia principal, descreve aspectos únicos, que atraem o jogador, relate a relação do modo de jogar com a história, o que o jogador fara nela, em que ambientes ou cenários ele passará, etc.

- Descrição dos personagens: descrever cada personagem do game brevemente, relate seu nome, e um resumo de suas características físicas, de personalidade, antecedentes, história, relevância para a história do game, etc

- Análise de riscos: descreva tudo aquilo que pode dar errado com o projeto do game e como será feito para se preparar para lidar com eles, riscos comuns de serem descritos, dificuldade de recrutamento, atrasos de entrega de matérias, dependência de fontes externas em geral, etc., deve-se comentar também aquelas partes do projeto que em sua visão estão relativamente seguras.

- Orçamento de desenvolvimento: é nesta seção que as empresas descrevem suas estimativas de gastos e lucros com o possível projeto, custos que devem ser citado

aqui como custos diretos de gastos com grupo de desenvolvimento mensal, equipamentos, etc., custos das mercadorias vendidas, gastos de embalagem, disco, custo com marketing, estimativas de receita com cada unidade vendida, despesas com devoluções, direitos autorais, etc.

- Arte conceitual: essa seção deve conter ilustrações conceituais e esboços de personagens e cenas do game, personagens devem ser representados de frente, de perfil e costas, forneça também possíveis telas do game mostrando ambiente e personagens principais, descreva também qual o estilo artístico pretende ser usado no game.

2.2.2.2 Documento de guia de estilo da arte

O segundo documento da fase de pré-produção é o documento de guia de estilo da arte, este documento vem para se criar a base da identidade visual do jogo, como o jogo deverá ser abordado e qual o estilo empregado em seus elementos visuais.

"A finalidade do guia de estilo da arte ou plano de ilustração(consistindo basicamente em um conjunto de ilustrações) é estabelecer a aparência geral do game e fornecer uma referência para o trabalho de outros ilustradores[…]". (Novak, 2010, p 377)

Este documento então consiste das ilustrações que o projeto do game se guiara durante todo seu processo de desenvolvimento, sempre seguindo o estilo de arte aqui definidos. Segundo Novak (2010) este documento deve ser elaborado pelo artista conceitual e pelo diretor de arte do projeto, nele poderão ser incluídos todo tipo de desenho artístico do game desde esboços a lápis a imagens digitalizadas, que reflitam o visual do esperado do game, podem também ser adotado nele uma biblioteca de referência visual de terceiros, como sites da web ou imprensa, mas estas não deveram ser utilizadas no produto final, apenas como referência para o projeto.

2.2.2.3 *Game Design Document* (GDD)

O Game Design Document (GDD) ou Documento de Design do Game (DDG) é o documento de maior relevância do processo de desenvolvimento e o documento final da fase de pré-produção do projeto do game. Segundo Novak (2010 p.374),

> a finalidade do DDG é ser usado como guia de referência durante todo o processo de desenvolvimento do game. O DDG concentra-se no modo de jogar, na história, na interface, nas regras do game. Ele deve especificar as regras do game com um nível de suficiente de detalhamento para que você em tese possa jogar o game sem usar um computador.

De acordo com Schuytema (2008) este é o documento que é visto como a planta baixa verbal do projeto do game, é o coração e alma de todos os documentos que giram em torno do desenvolvimento de um game. Ainda de acordo com Schuytema este documento existe em diversas formas e modelos, de documentos pequenos, Wikis[4] detalhadas a enormes volumes com centenas de páginas, que é criado nesta fase do desenvolvimento mas que muda constantemente ao longo do projeto e que deve ser atualizado a conforme a necessidade.

Novak (2008) relata ainda que este documento deve estar disponível a todos os membros do projeto. Ainda segundo Novak este documento é uma combinação de alguns dos elementos presentes em documentos já criados, como o documento de proposta, de guia de arte, e de conceito, com a adição de outros elementos.

O documento que será apresentado nesta monografia é uma combinação dos elementos presentes nos documentos apresentado pelos autores Novak (2010) e Schuytema (2008), assim como a descrição de cada elemento presente.

1. Visão geral essencial: nesta seção se apresenta uma visão breve porém detalhada do game, o objetivo é apresentar e familiarizar o game a qualquer pessoa, sua ideia e jogabilidade.

---

[4] Wiki: é um site colaborativo que permite usuários criar, adicionar, modificar ou deletar conteúdo do site. (SHELLY; GUNTER; GUNTER, 2010)

- Resumo: apresentar um resumo, síntese, do que será apresentado no game, pode-se utilizar dos documentos já criados porém, este será voltado aos desenvolvedores então deve-se deixá-lo o mais compacto e eficiente possível.
- Aspectos fundamentais: apresentar a essência, os componentes fundamentais do game, a trama central para a experiência do jogador assim como sua diversão, mantenha o foco no *gameplay*[5] e no que o jogador fara dentro do jogo.
- Diferencial: outra seção que está presente nos documentos já gerados e que pode ser reutilizada aqui, basicamente é o que torna seu jogo diferente, melhor do que os concorrentes.
- Estilo/Gênero: novamente uma seção já abordada anteriormente no documento de conceito que pode ser aproveitado aqui, defina o estilo do jogo, seu gênero, se será de plataforma, FPS[6], RPG[7], etc.

2. Contexto do game: nesta seção devera ser descrito o mundo que rodeia o game, como por exemplo a floresta amazônica brasileira, ou a muralha da china, etc., será o contexto do game, ou seja, onde o jogador ira ter toda a ação, história do jogo.
    - Cenário: o ambiente, local, aonde o jogo acontece, pais, lugar, estação do ano, etc., o que for necessário para descrever o que rodeia o jogo.
    - Historia do game: uma seção que pode ser aproveitada do documento de proposta, descreve-se a história do começo ao fim do jogo.
    - Eventos anteriores: novamente outra que está presente no documento de proposta, basicamente descreve-se nesta seção o que aconteceu anteriormente a história do game, eventos, ações, que contextualizaram a história do game.
    - Principais jogadores: de forma geral essa seção está presente no documento de proposta, porém aqui apresente apenas aqueles personagens centrais do jogo, como por exemplo o herói da história(personagem o qual será controlado pelo jogador), seu(s) arque inimigo(s), sua princesa, etc., descreva aspectos como

---

[5] Gameplay: ou jogabilidade, está relacionado a experiência do jogador dentro do game. (NOVAK, 2010)

[6] FPS: *First Person Shooter* ou Tiro em primeira pessoa um gênero de jogos. (NOVAK, 2010)

[7] RPG:*Role-Playing Game,* jogo de representação de papéis, jogador assume o papel de um personagem no jogo tendo que explorar, matar monstros e coletar itens . (NOVAK, 2010)

bibliografia, habilidades especiais, motivação, o que for necessário para definir como agira durante o game.

3. Objetos essenciais do game: seção responsável por descrever os objetos que aparecem no game e que possuem um papel, ou seja, que afetem ao gameplay, que façam parte da experiência do jogo.
    - Personagens: descrever todos os personagens do jogo, dos principais aos NPCs, aliados, inimigos, dos mais importantes aos que apenas estão ali para contextualizar algo sem tem real importância ao *gameplay*.
    - Habilidades: descrever nesta seção habilidades dos personagens, inatas, habilidades de combata, e de defesa, que podem ser adquiridas/melhoradas, de todos os personagens, controláveis ou NPCs.
    - Armas: descreve-se nesta seção todos os tipos de armas que tem um papel no game, ou seja, que possam ser usadas, por personagens durante o jogo.
    - Estruturas: descreva aqui todas as estruturas que tenham algum efeito ao jogo, estruturas como casas de *checkpoint*[8], torres de defesa dos inimigos, etc.
    - Objetos / Itens: esta seção serve para descrever objetos/itens relevantes ao jogo que não se encaixam em outras seções, aqui pode0se descrever itens como itens para locais secretos mas não essenciais a linha principal da história do game, ou itens para *craft*[9], ou *powerups*[10], etc.
4. Conflitos e soluções: nesta seção deve ser descrito aspectos relacionados as iterações entre entidades do jogo, ou seja, como uma ação é feita, por exemplo, pelo jogador e o que acontece em seguida, exemplificando podemos dizer em um jogo FPS, devera ser descrito como será a mira da arma, qual a velocidade com que o jogador mirará e atirará, o que acontece com o outro jogador quando for acertado e como saberá que foi acertado, etc.
5. Inteligencia artificial: descreve nessa seção elementos relacionados a IA do game, descreva quais os comportamentos dos elementos controlados pela IA do game terão e quaisquer outras informações relacionadas sobre tais comportamentos.

---

[8] Checkpoint: são locais ou estruturas onde o jogador salva seu status, progresso no jogo.

[9] Craft: criação de itens dentro do jogo.

[10] PowerUp: item que aumentar ou melhorar o personagem, sua velocidade, força, etc

6. Fluxo do game: uma importante seção, porém que não deve apresentar nada novo, esta seção apenas devera descrever como será cada nível, missão, enigma do jogo, incluindo informações de itens encontrados nelas, condições de vitória/derrota, logo não devera apresentar nenhum objeto novo e sim explicar e definir como todos os objetos ate aqui definidos em outras seções serão apresentados para o jogador dentro do ambiente virtual do jogo.
7. GamePlay / Controles: está seção deve descrever a mecânica do game, como o jogador jogara o jogo em si, quais os controles que ele terá no game, as ações no game, também pode-se relatar mudanças nesses controles, caso o jogo for multiplataforma, de uma plataforma para outra.
8. Variação de jogo: esta seção é opcional pois depende se o jogo terá diferentes modos de jogo, como por exemplo *singleplayer*[11] ou *multiplayer*, *co-op*[12], online, etc., caso possua deve-se descrever como tal modo de jogo alterara a mecânica do jogo, o que altera no game, se haverá mudança nas regras, nas condições de vitória/derrota, se haverá alguma mudança no estilo do jogo, etc.
9. Interface: nesta seção devera ser descrito todas as interfaces presentes no game, interfaces ativas e passivas, ou seja, interface em possuem funcionalidades ou simplesmente estão ali para caracterizar o ambiente/tela do jogo, descreve cada função da interface, elementos interligados, característica e funcionalidade no game.
10. Mapas: nesta seção apresente os mapas e ambientes do jogo, apresente como será disposto o mapa, as cidades, vilas, como será a interligação entre o ambiente, por exemplo, casas, florestas, mostre possíveis efeitos cinemáticos, climáticos, apresente o cenário, fundo do jogo caso seja do estilo plataforma(sonic,mario), etc.
11. Áudio: apresente nesta seção as possíveis necessidades de sons do jogo, sons ambiente, músicas, ações, etc., pode-se apresentar uma trilha sonora específica criada para o game.

---

[11] Singleplayer: ao contrário de multiplayer um jogo singleplayer apenas um jogador joga o game. (NOVAK, 2010)

[12] Co-op: ou cooperativo é a quando mais de um jogador competem juntos contra o game. (NOVAK, 2010)

12. Definições: basicamente esta seção é o glossário do DDG, aqui descreva apenas os termos utilizados que não são muito claros para o entendimento de qualquer pessoa que leia o DDG.
13. Referencias: seção que contem matérias para contextualizar a ideia, clima do game, pode-se inserir nesta seção elementos como lista de filmes inspiradores, livros, ate mesmo games concorrentes para sanar qualquer dúvida quanto a essência do game, pode-se ainda adicionar aqui elementos criados em outros documentos, que sejam elementos artísticos como imagens, modelos, rascunhos conceituais, tudo aquilo que servira como referência no desenvolvimento do game.

2.2.3 Produção

A fase de produção, é aonde realmente o jogo que ate então era apenas rabiscos em papéis, ideias na cabeça de designers e textos em documentos, passa a ser criado, sair do papel, e seu desenvolvimento começa, tornando tudo aquilo em um jogo.

> Na produção, o game é construído. Os artistas criam modelos de personagens e níveis, e os programadores escrevem e revisam o código-fonte. O setor de marketing passa a desenvolver uma estratégia de propaganda, e o setor de teste começa a avaliar o game a cada nova versão. Os designers fazem o roteiro do gameplay, avaliando-o em termos de diversão e trabalhando de perto com o setor de arte e programação para garantir que a funcionalidade do game esteja coerente com os documentos de design.[…] (Schuytema, 2008, p. 13)

Segundo Schuytema (2008) é de grande importância que nessa fase do processo de desenvolvimento de um game, os designers continuem aperfeiçoando os documentos criados na fase de pré-produção, com informações, o *feedback*[13] provenientes dos setores de programação, teste e orientações de produtores e gerentes, para que assim se mantenha tudo atualizado e todas as equipes seguindo o mesmo "guia".

[13] FeedBack: é o retorno de informações sobre algo.

Esta fase é a mais longa em todo processo de desenvolvimento de m jogo digital e que também é nesta fase que ocorrem os maiores problemas. Segundo Novak (2010) na fase de produção aonde o game é efetivamente desenvolvido, geralmente tem uma duração de 6 meses como mínimo a 2 anos como media de desenvolvimento de um jogo bem-acabado.

Ainda segundo Novak (2010) está fase costuma ter três períodos bem definidos, o período chamado por ele de Alfa e um segundo momento chamado de Beta e um último chamado de Ouro.

2.2.3.1 Alfa

A fase alfa é onde o jogo torna-se jogável vindo do período de produção do game, nesta fase Novak (2010) define que "a fase alfa é o ponto em que um game pode ser jogado do começo ao fim", mas o jogo em si não esta totalmente completo, nesta fase o jogo ainda tem "pedaços" faltando, como por exemplo, imagens faltando, cenários incompletos, objetos sem textura, etc.

Durante esta fase há também ainda de acordo com Novak (2010) uma incorporação de uma equipe de testadores temporários de jogabilidade para identificarem problemas, algumas vezes esta equipe é composta por pessoas externas, jogadores comuns, que dispõem a testarem o jogo.

Novak (2010 p.347) seguindo o livro de Mark Mencher, *Get in the Game*, elege nove elementos que definem com sua conclusão o fim desta fase, são eles:

- Percurso completo de jogo(ou seja, ele é jogável do começo ao fim);
- Texto no idioma básico;
- Interface básica, com documentação preliminar;
- Compatibilidade com a maioria das configurações de hardware e software especificadas;
- Requisitos mínimos dos sistemas testados;
- Maioria das interfaces manuais testadas quanto a compatibilidade;

- Arte e áudio temporários;
- Recursos multijogadores testados(quando aplicável);
- Rascunho do manual do game;

2.2.3.2 Beta

Posteriormente a fase alfa esta fase entra-se em uma fase conhecida como a fase beta do game, nesta fase o jogo deve chegar praticamente pronto, esta fase segundo Novak (2010) tem como finalidade a correção de *bugs*, problemas eventuais no jogo, Novak (2010, p.348) ainda relata que "Na, fase beta, o objetivo é estabilizar o projeto e eliminar o maior número possível de defeitos antes que o produto comece a ser vendido".

Nesta fase os produtores de jogos costumam recrutar on-line um número considerável de jogadores dispostos a testar o jogo, em sua maioria esta decisão se dá pelo fato que os produtores querem verificar a estabilidade do jogo, testes de estresse são realizados graças ao número elevado de jogadores simultâneos, isto claro, em casos de jogos on-line, nos outros casos estes testes também servem para detectar problemas de jogabilidade, desempenho em diferentes tipos de hardware/plataformas, em casos de jogos para PC também aos diferentes tipos de sistemas operacionais. Para que assim se verifique que o jogo atende os padrões de qualidades dos fabricantes.

Ao fim da fase de testes Novak (2010) ainda relata uma fase secundaria dentro da fase beta que ele descreve como fase de "*congelamento do código",* onde todo trabalho ate então deve ser finalizado e preparado para as diversas mídias as quais o jogo será distribuído, pequenos ajustes para solução de problemas durante esta fase são os únicos permitidos.

Apos todo o trabalho nesta fase, uma das mais importantes em todo o desenvolvimento, Novak (2010, p.348) descreve os elementos que devem estar completos para que se possa sair da fase beta.

- Código;
- Conteúdo;

- Texto em diferentes idiomas;
- Navegação pelo percurso do game;
- Interface do usuário;
- Compatibilidade da interface manual;
- Arte e áudio;
- Manual do game;

2.2.3.3 Ouro

Tendo passado pelas fases de produção, alfa e beta, o game chega a fase de ouro, nesta fase segundo Novak (2010) o game passa pelos últimos testes em suas versões mídia/digital, e posteriormente a aprovação da administração e confirmação que o game está pronto, o game é lançado no mercado.

2.2.4 Pós-produção

Esta fase do processo de desenvolvimento de um game ocorre após seu lançamento, a fase de pós-produção ou pré-lançamento, cuida do que ocorra com o game a partir deste ponto, Schuytema (2008, p.13) relata que esta fase pode trazer ao game "[…] o design de conteúdo adicional para download, a criação de conteúdo para *patches*[14] que continuam o processo de balanceamento do *gameplay*, ou avaliação da receptividade ao game, de olho em futuras sequências ou pacotes de expansão".

Atualmente empresas desenvolvedoras começaram a se focar muito no desenvolvimento dos chamados DLCs(ou *Downloadable Content*) que nada mais é do que

[14] Patch: são atualizações que servem para corrigir ou adicionar novas funcionalidades ao jogo.

expansões com adição de novas funcionalidades, modos de jogo, itens, ao conteúdo original do game, aonde o jogador deve pagar para ter esses novos recursos adicionais, Novak (2010) relata que esses tipos de produto são "[…] lançadas para substituir e melhorar o game original, aumentando sua longevidade[…]", porém empresas vem se utilizando desse artifício para arrecadar mais dinheiro, com o lançamento de diversos DLC já anunciados com o lançamento do game original, ou em outros casos como podemos citar jogos como *World of Warcraft* da desenvolvedora *Blizzard*, o jogador além do pagamento mensal para jogar, deve ainda pagar por grandes expansões como *Cataclysm* e a mais *Mist of Pandaria*, pois nesse caso trazem recursos os quais deixam quem as não compra impossibilitado de progredir no game, pois sem elas não se pode, por exemplo, alcançar o nível máximo, o qual é constantemente aumentado nas expansões.

Com a chegada desta fase, o processo de desenvolvimento do game chega ao fim, aqui é desenvolvido *patches*/DLC ate o fim do game, ou do interesse da produtora no mesmo, para que assim se recomece todo o processo novamente para o desenvolvimento de um novo jogo ou sequência.

2.3 UNIFIED MODELING LANGUAGE

O desenvolvimento de softwares é, geralmente, um processo pouco arquitetado, caótico, sem orientações de natureza estratégica e planos de gerenciamento e controle. Esses problemas no desenvolvimento de um software são de tal dimensão, que se torna fundamental, definir e aplicar certos princípios, regras e estratégias que gerem melhorias significativas em todo processo de desenvolvimento do projeto.(RAMOS, 2006)

E segundo Ramos (2006 p.18), "Em qualquer desenvolvimento de sistemas de informação, é preciso definir passos, regras e coordenar corretamente as interações entre as pessoas, os procedimentos aplicados, as características do produto e o projeto que orienta as atividades a serem desenvolvidas.", ou seja é preciso construir algo que ira guiar o processo de desenvolvimento de um sistema, software.

Ainda de acordo com Ramos (2006) só pessoas motivadas e comprometidas com o projeto do software asseguram seu sucesso, só com procedimentos com técnicas e regras bem definidas podem chegar aos objetivos propostos.

A *Unified Modeling Language* (UML) é uma linguagem-padrão para a elaboração da estrutura de projetos de *software*. Ela pode ser utilizada para visualizar, especificar, construir e documentar elementos que faram parte de sistemas complexos de software. (BOOCH et al., 2000).

Segundo Booch et al. (2000) a UML é uma linguagem adequado para a modelagem de quaisquer sistemas, de sistemas web de empresas. Há grandes e complexos sistemas embutidos de tempo real, ela é muito abrangente pois possui todas as visões necessárias ao desenvolvimento e implantação de sistemas, e apesar de sua expressividade ela é uma linguagem de fácil compreensão e utilização.

> A ênfase da UML é na definição de uma linguagem de modelagem padrão(*standard*) e, por conseguinte, independente de linguagens de programação de ferramentas CASE, bem como dos processos de desenvolvimento. O objetivo da UML é, dependendo do tipo de projeto, da ferramenta de suporte ou da organização envolvida, poder adotar diferentes processos/metodologias, mantendo-se contudo, a utilização de uma única linguagem de modelagem(Ramos, 2006, p.8)

Ramos (2006) elenca como princípio básico da UML sua simplicidade, e coerências quanto a unificação de diferentes elementos existentes em seus vários métodos, e cita como seus elementos bases os seguintes itens:

- Mecanismos de extensão;
- Elementos para modelar processos e *threads*;
- Elementos para modelar distribuição e concorrência;
- Padrões de projeto e colaborações;
- Diagramas de atividades(para modelagem de processos de negócio);
- Refinamento(para tratar relações entre diferentes níveis de abstração);
- Interface e componentes;
- Linguagem de restrições(*Object Contraint Language*);

Além de citar os diagramas cuja linguagem UML possui:

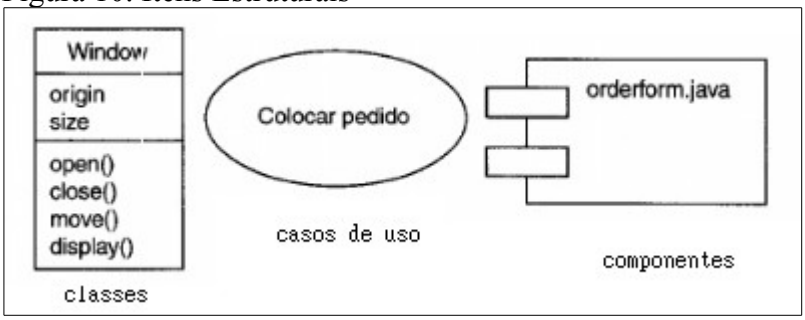
- Diagramas de casos de uso;

- Diagramas de classes e diagramas de objetos;
- Diagramas de comportamento;
- Diagramas de estados(*statechart*);
- Diagramas de atividades;
- Diagramas de interação (diagramas de sequência e diagramas de colaboração);
- Diagramas de arquitetura;
- Diagramas de componentes;
- Diagramas de instalação;

Nesse contexto a linguagem UML é algo importante no processo de desenvolvimento de softwares, e como em toda linguagem possuem um certo padrão/conceito a ser seguido. Booch et al. (2000) cita três importantes blocos que compõem a contexto dessa linguagem, são eles os Itens, os Relacionamentos e os Diagramas que já foram citados aqui, a onde, ele relata que "os itens são abstrações identificadas como cidadãos de primeira classe em um modelo; os relacionamentos reúnem estes itens; os diagramas agrupam coleções interessantes de itens".

- Itens: os itens são os blocos de construção básicos orientados a objetos da UML e serão utilizados para escrever os modelos, itens podem ser de quarto tipos estruturais, comportamentais, de agrupamento e anotacionais, alguns exemplos de itens nas Figuras 10, 11, 12 e 13:

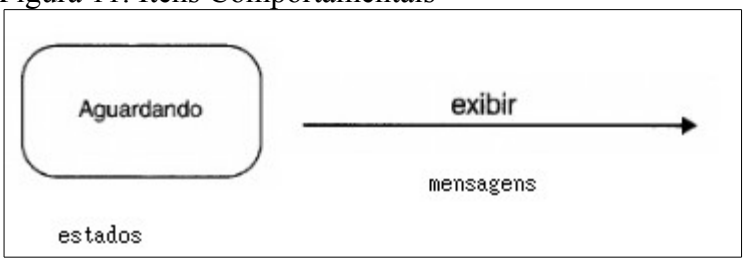
Figura 10. Itens Estruturais



Fonte: Adaptado de BOOCH et al., 2000.

Como observado na Figura 10 os itens estruturais podem ser, como exemplo as classes, os casos de uso e os componentes em diagramas.

Figura 11. Itens Comportamentais



Fonte: Adaptado de BOOCH et al., 2000.

Já na Figura 11 como itens comportamentais os estados e as mensagens dos diagramas.

Figura 12. Item Agrupamento

Fonte: Adaptado de BOOCH et al., 2000.

Na Figura 12 como exemplo de item de agrupamento as regras de negócio e na como pode ser visto na Figura 13 como itens anotacionais as notas

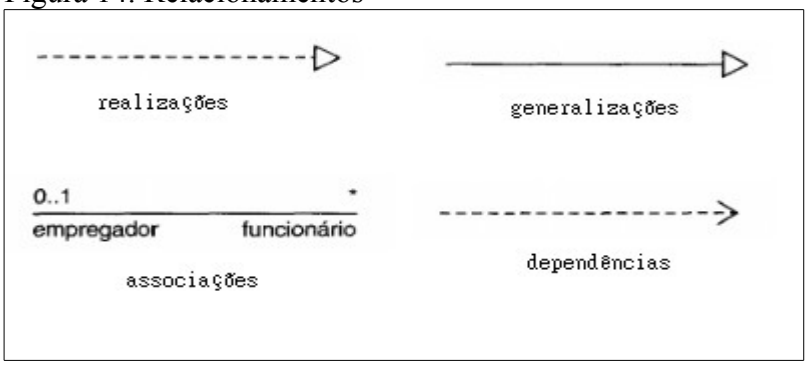
Figura 13. Item Anotacional

Fonte: Adaptado de BOOCH et al., 2000.

- Relacionamentos: os relacionamentos são os blocos relacionais básicos de construção da UML e serão utilizados para escrever os modelos, relacionamentos assim como

itens podem ser de quarto tipos dependência, associação, de generalização e realização, alguns exemplos de relacionamentos na Figura 14:

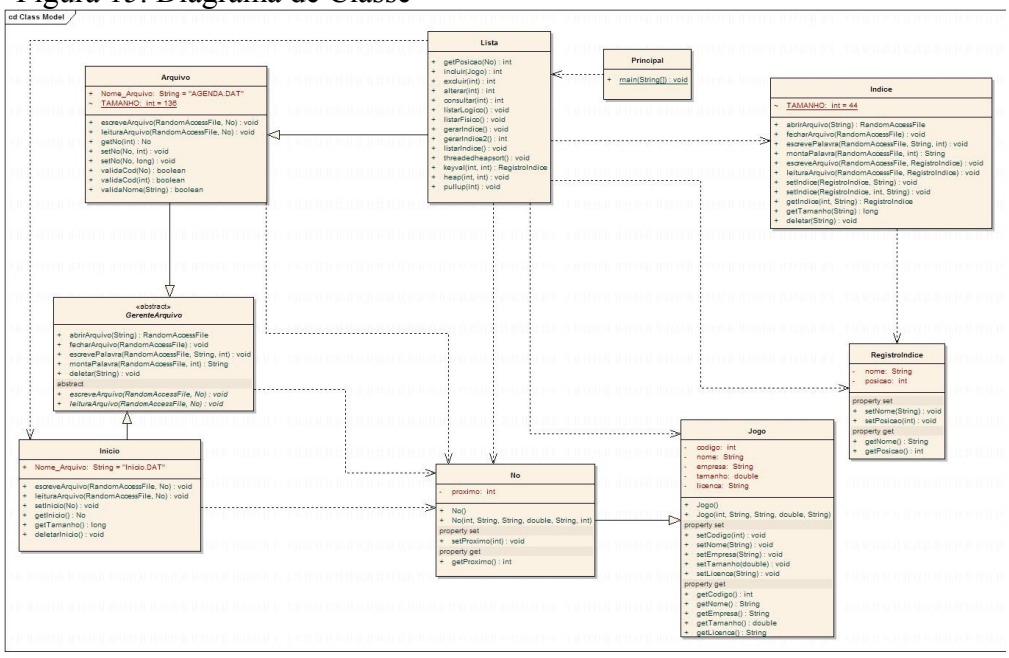
Figura 14. Relacionamentos



Fonte: Adaptado de BOOCH et al., 2000.

- Diagramas: um diagrama é a apresentação gráfica de um conjunto de elementos, geralmente representadas como gráficos de vértices(itens) e arcos(relacionamentos), um exemplo diagrama de classe pode ser visto na Figura 15:

Figura 15. Diagrama de Classe



Fonte: Elaboração dos Autores, 2013.

Como visto na Figura 15 os diagramas de classes desenhados para permitir a visualização de um sistema sob diferentes perspectivas, ou seja, um diagrama é uma projeção gráfica de um determinado sistema contendo seus relacionamentos e elementos estruturais.

2.4 GÊNEROS DE JOGOS

Com a grande crescente do mercado de games, viu-se a necessidade de dividir os jogos em gêneros, afinal cada tipo de jogo era diferente de outro. Nesta seção são apresentados os principais gêneros de games da atualidade.

Segundo Novak (2010), Gêneros de games são categorias baseadas em uma combinação de tema, ambiente, apresentação/formato na tela, perspectiva do jogador e estratégias de jogo.

Ainda de acordo com Novak (2010, p.96) alguns exemplos de gêneros utilizados em jogo são:

- Ação: O gênero de ação existe desde a febre do fliperama. De fato, praticamente todos os jogos de fliperama são jogos de ação. O objetivo da maioria dos games de ação é destruir rapidamente os inimigos. Exemplos de games de ação são: Pac-Man e *Asteroids*. (NOVAK, 2010 p.96)

- Games de Plataforma: O gênero de plataforma concentra-se na movimentação rápida dos jogadores em um ambiente. Os exemplos de games de plataforma incluem alguns dos primeiros games de fliperama, como Donkey Kong e Sonic the Hedghog. (NOVAK 2010, p.97)

- Games de Tiro: Os games de tiro podem ser divididos em duas vertentes, os games *first person shooter* ou FPS que são os jogos em que a visão do jogador é uma visão de primeira pessoa. A outra linha de jogos de tiro são os jogos em terceira pessoa, que permitem que o jogador veja seu próprio personagem, junto ao restante do mundo do game. Atualmente no mercado os dois grandes games de tiro FPS que disputam entre si são *Call of Duty*, da *Activision* e *Battlefield*, da *Eletronic Arts*.(NOVAK 2010, p.98)

- Corrida: Exemplos bem conhecidos de jogos de corrida são: *Need for Speed, Test Drive, Gran Turismo* e *Top Gear*. Um jogo de corrida abrange o veículo do jogador

competindo contra um ou mais adversários em uma ampla variedade de pistas ou terrenos. O jogador tenta dirigir à maior velocidade possível sem perder o controle do veículo.(NOVAK, 2010 p. 98).

- Luta: muitos títulos de luta são games para duas pessoas, e que no geral, cada jogador controla um personagem que tem como objetivo derrotar o adversário atacando e defendendo dos ataques do mesmo. *Mortal Kombat* e *Street Fighter* são bons exemplos de jogos de luta.(NOVAK 2010, p. 99)

- Aventura: características dos games de aventura incluem exploração, coleta de itens, solução de quebra-cabeças, orientação em labirintos e decodificação de mensagens. *Adventure* foi o primeiro jogo de aventura por texto.(NOVAK 2010, p.100)

- Ação-Aventura: o gênero de ação-aventura é híbrido, se consolidou como um gênero distinto por seus próprios méritos. O componente de ação requer reflexos rápidos nos movimentos do personagem para desviar dos inimigos e combatê-los, enquanto o componente de aventura adiciona quebra-cabeças conceituais e elementos da narrativa do game. A saga *God of War* é um exemplo de jogos de ação-aventura.(NOVAK 2010, p.101)

- Cassino: jogos de cassino são normalmente versões digitais de jogos populares, como roleta, poker e caça-níqueis encontrados em cassinos reais.(NOVAK 2010, p. 102)

- Quebra-Cabeça: Os jogos de quebra-cabeça são muito famosos por serem jogos casuais, que não exigem um longo período de tempo para a diversão. […] em um game de quebra-cabeça puro o jogador deve solucionar um problema ou uma série de problemas sem controlar um personagem. Tetris é um dos jogos de quebra-cabeça mais famoso da história dos games (NOVAK, 2010 p.102).

- Games de Representação de Papéis (RPG): Os *role-playing games* também conhecidos apenas como RPGs tem como base a criação de um personagem para o

jogador. Os games de representação de papéis originam-se da tradição iniciada na década de 1970 pela série de jogos de RPG *Dungeons & Dragons*, que eram jogados com papel e lápis. Neles, os jogadores assumiam papéis de guerreiros, magos, sacerdotes, elfos ou ladrões e exploravam calabouços, matavam monstros e coletavam tesouros. (NOVAK, 2010, p. 103).

- Simulação: Games de simulação incluem simulações de veículos, processos, e participativas. As regras associadas a todos os games de simulação são baseadas em situações e objetos do mundo real. As simulações tentam reproduzir sistemas, máquinas e experiências usando regras do mundo real. Dois jogos simuladores bem conhecidos são o *Flight Simulator* e o *The Sims*. (NOVAK, 2010 p.106)

- Estratégia: Os jogos de estratégia são divididos em 2 gêneros: Estratégia Baseada em Turnos (TBS) e Estratégia em Tempo Real (RTS). Bons exemplos de jogos de estratégia são *Age of Empires* (RTS) e *Civilization* (TBS). Os games de estratégia originaram-se dos jogos clássicos de tabuleiro, como xadrez, em que os jogadores devem administrar um conjunto limitado de recursos para atingir uma meta específica A maioria dos games de estratégia desenrola-se em um ambiente militar. (NOVAK, 2010, p. 110).

2.5 PLATAFORMA ANDROID

Segundo Pereira e Silva (2009), o Android é uma plataforma para tecnologia móvel completa, envolvendo um pacote com programar para celulares, já com um sistema operacional, *middleware*, aplicativos e interface do usuário.

> Android foi construído com a intenção de permitir aos desenvolvedores criar aplicações móveis que possam tirar total proveito do que um aparelho portátil possa oferecer. Foi construído para ser verdadeiramente aberto. Por exemplo, uma aplicação pode apelar a qualquer uma das funcionalidades de núcleo do telefone, tais como efetuar chamadas, enviar mensagens de texto ou utilizar a câmera, que permite

> aos desenvolvedores adaptarem e evoluírem cada vez mais estas funcionalidades.. (Pereira, Silva, 2009, p. 3).

Graças ao seu modelo *Open Source*, o Android possibilita que ele se adapte mais facilmente a novas tecnologias e possibilitando a melhoria das aplicações já existentes, por parte de desenvolvedores que não sejam da empresa.

Ainda segundo Pereira e Silva (2009), a plataforma estará sempre em evolução, já que as comunidades de desenvolvedores trabalharão em conjunto para construir aplicações móveis inovadoras. Logo graças as especificações, foi escolhida a plataforma Android para o desenvolvimento do jogo proposto neste trabalho.

O Android inclui um Sistema Operacional (OS) baseado em um kernel Linux, uma rica interface de usuário, aplicativos de usuário, bibliotecas de código, *frameworks* de aplicativos, suporte a multimídia e muito mais. Os aplicativos de usuário são escritos em Java. (Ableson et.al, 2012).

Para uma aplicação que está a ser desenvolvida acessar uma funcionalidade do aparelho, a mesma deve ter a permissão para o uso, facilitando assim a segurança da aplicação e a robustez da mesma sem necessidade de ter tal regra em código.

Quando às permissões, Pereira e Silva (2009) afirmam que, se uma aplicação precisa fazer o uso do GPS do aparelho, é preciso ter a permissão explícita no arquivo AndroidManifest.xml: android.permission.Access_Fine_Location.

Ainda segundo Pereira e Silva (2009), Cada processo da aplicação no Android é considerado uma sandbox. Só é possível acessar outras aplicações caso tenha as permissões explicitamente declaradas.

O Android também funciona com assinaturas, onde cada aplicação deve conter uma assinatura para poder ser publicada e para que a mesma tenha a "confiança das outras aplicações". Pereira e Silva (2009) confirmam isto dizendo que toda aplicação Android precisa ser assinada com um certificado cuja chave privada é mantida pelo seu desenvolvedor. Este certificado é utilizado apenas para estabelecer relações de confiança entre as aplicações.

2.6 LINGUAGEM DE PROGRAMAÇÃO JAVA

Para esta monografia os autores utilizaram a linguagem de programação Java. De acordo com Deitel (2003) o Java é uma linguagem de programação desenvolvida na década de 90, por James Gosling, originalmente com nome de *Oak*(carvalho), mas que foi alterado por descobrir-se que já havia uma linguagem chamada assim, Gosling então após visitar uma cafeteria local, foi-se sugeria a ele o nome Java(cidade de origem de um tipo de café importado), assim batizando sua linguagem com este nome.

A linguagem Java é uma linguagem de alto nível e que segundo Lemay e Cadenhead (2005 p.4) "Java é uma linguagem orientada a objetos, independente de plataforma e segura, projetada para ser mais fácil de aprender do que C++ e mais difícil de abusar que C e C++".

Um diferencial do Java é sua neutralidade quanto a plataforma onde o programa será executado, Horstmann (2003 p.40) relata que

> quando compilamos nosso programa, o compilador traduz o código fonte Java (isto é, o comando que escrevemos) para *bytecode*, que consiste em instruções para a máquina virtual e alguns outros itens de informação sobre como carregar o programa na memória antes da execução.

Lemay e Cadenhead (2005) ainda relatam que este *bytecode* pode ser executado sem qualquer modificação em diferentes ambientes, ou seja, em qualquer sistema operacional, software ou dispositivo com um interpretador Java, como, por exemplo, dispositivos móveis, independente de qual máquina foi criado ou gerado.

Uma das razoes para tal neutralidade é descrita por Deitel (2003), os programas gerados nessa linguagem, são passados para *bytecode* e após interpretados e executados sobre uma JVM (*Java Virtual Machine*), cujo é um software especifico que simula um computador, assim tornando indiferente sobre qual sistema o programa é rodado.

Quanto a programação orientada a objetos Lemay e Cadenhead (2005 p.4) ainda falam que,

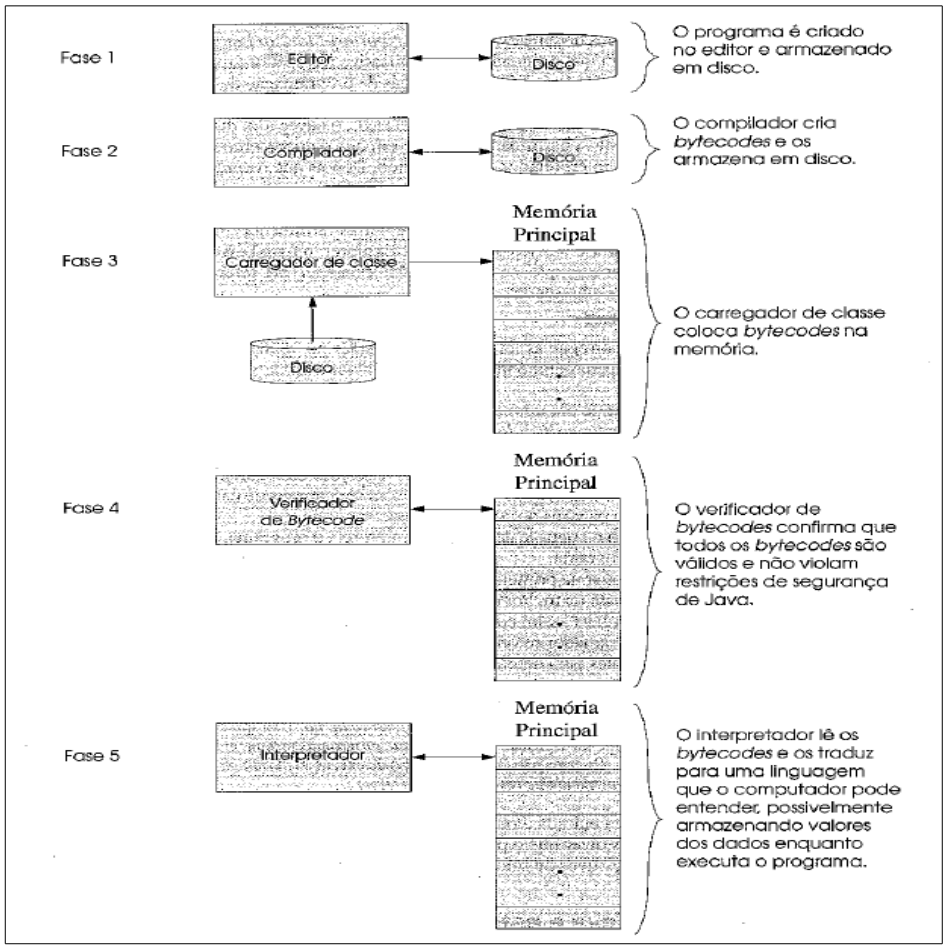
> é uma metodologia de desenvolvimento de software em que o programa é percebido como um grupo de objetos que trabalham juntos. Os objetos são criados com

> modelos, chamados classes, e contem os dados e as instruções necessárias para usar esses dados

Em outras palavras, um objeto é um elemento, entidade, que possui atributos e métodos, e uma classe é um conjunto de objetos com características semelhantes, que define por meio de outros métodos o comportamento desses objetos (Ramos, 2006).

Outra característica da linguagem Java é o modo como são desenvolvidos e executados seus sistemas, conforme descrito por Deitel (2003) eles passam por cinco etapas/fases como pode-se verificar na Figura 16:

Figura 16. Ambiente Java



Fonte: Deitel, 2003

Segundo Deitel (2003) essas fases são: a edição, quando o programa é escrito, gerando o código-fonte através de um editor de texto ou uma IDE(*Integrated Development*

Environments), a fase de compilação, momento o qual é responsável pela criação do *bytecode* e quando são validados a sintaxe e semântica do código-fonte, a terceira fase a de carregamento o qual é responsável pôr o programa na memória a partir do código compilado, ou seja, *bytecode* para que futuramente seja executado pela JVM, a quarta fase a de validação, que é responsável por verificar se os *bytecodes* são validos, se estão todos os *bytecodes* necessários e se não violam nenhuma restrição de segurança Java, e em sua última fase a de execução, aonde a JVM interpreta o programa, um *bytecode* por vez, e realizando as ações especificadas no programa.

3 MÉTODO

Neste capítulo é apresentado a metodologia definida para a criação desta monografia com relação a pesquisa e desenvolvimento deste projeto.

3.1 CARACTERIZAÇÃO DO TIPO DE PESQUISA

Conforme relata Silva e Menezes (2005), a metodologia científica pode ser caracterizada quanto ao tipo de pesquisa pode ser classificada do ponto de vista de sua natureza, da forma de abordagem do problema, dos seus objetivos e dos procedimentos técnicos.

No desenvolvimento desta monografia foi definido, como ponto de vista segundo sua natureza sendo uma metodologia de pesquisa aplicada que segundo Silva e Menezes (2005, p.20) "objetiva gerar conhecimentos para aplicação prática e dirigidos à solução de problemas específicos".

Já com relação a forma de abordagem do problema, foi escolhido como uma pesquisa qualitativa, definida por Silva (2005, p.20) como uma pesquisa aonde "[…]a interpretação dos fenômenos e a atribuição de seus significados são básicos no processo de pesquisa qualitativa. Não requer o uso de métodos e técnicas estatísticas[…] Os pesquisadores tendem a analisar seus dados indutivamente[…]".

Em relação aos objetivos de pesquisa foi relacionado a uma pesquisa descritiva pois busca-se descrever o processo de desenvolvimento de jogos, e conforme descrito por Gil (apud SILVA e MENEZES, 2005, p.21) a pesquisa descritiva "visa descrever as características de determinada população ou fenômeno ou o estabelecimento de relações entre variáveis".

Por último então foi definido quanto aos procedimentos técnicos a pesquisa sendo bibliográfica, que segundo Gil (apud SILVA e MENEZES, 2005, p.21) é "quando elaborada a

partir de material já publicado, constituído principalmente de livros, artigos de periódicos e atualmente com material disponibilizado na Internet".

3.2 ETAPAS

Aqui são apresentadas as etapas que descrevem o processo de desenvolvimento desta monografia.

Em um primeiro momento foi realizado a definição de um tema para o projeto, neste momento foi-se especificado o ambiente para o qual o projeto seria desenvolvido. Neste momento também foi definido após uma análise do tema escolhido como o projeto seria abordado em termos de objetivos, assim como sua justificativa para realização.

Já em uma segunda etapa, foi realizada a fundamentação teórica do projeto, aonde foram descritas aspectos teóricos quanto ao desenvolvimento do projeto, baseando-se em autores das áreas abordadas na produção da monografia.

Logo após foi desenvolvido um primeiro GDD, para definição do projeto do jogo e alguns outros aspectos referentes ao mesmo para dar continuidade ao trabalho.

A próxima etapa foi a realização da adaptação dos autores ao ambiente utilizado para o desenvolvimento do jogo e da biblioteca LibGDX, em conjunto foi realizado a finalização do GDD presente nesta monografia.

Em sequência foi desenvolvido um projeto UML referente ao desenvolvimento do jogo, contendo diagramas para visualização do projeto.

A etapa seguinte foi a realização do desenvolvimento do jogo em si, objetivo final desta monografia, nesta etapa foi se desenvolvido toda a codificação do jogo assim como as *sprites* e toda sonorizações presentes no mesmo.

E na etapa final de desenvolvimento desta monografia foram realizados os testes e a avaliação do produto final, o jogo para Android.

3.3 DELIMITAÇÃO

Para o presente projeto foram determinadas certas delimitações, que são apresentadas a seguir:

- O jogo desenvolvido apenas em linguagem de programação Java.
- Não é utilizado nenhum motor gráfico(*Engine)*.
- Utilização apenas de uma biblioteca gráfica a LibGDX.
- O jogo não se preocupará em gráficos de muita qualidade e apresenta apenas gráfico em 2d.
- Utilização de técnicas de inteligência artificial.

4 GAME DESIGN DOCUMENT DO PROJETO

Neste capítulo é apresentado o GDD do game, *The Three Sealed Elements,* desenvolvido durante a construção desta monografia, de maneira simplificada por não se tratar de um jogo produzido por uma grande empresa e baseado em elementos presentes nos documentos apresentado pelos autores Novak (2010) e Schuytema (2008).

4.1 VISÃO GERAL ESSENCIAL

Nesta seção é apresentado uma visão breve, porém detalhada do game a ser desenvolvido, o objetivo é apresentar e familiarizar o jogo a qualquer pessoa, sua ideia e jogabilidade.

4.1.1 Resumo

A história começa com o jovem herói retornando a sua cidade natal, lugar conhecido por ser um antigo monastério. Neste lugar monges antigos aprisionaram um mal nunca antes visto, que poderia ter acabado com toda a vida no planeta, e que a transformou em um local sagrado.

O herói que havia chegado de uma jornada de aventura pelo mundo atrás de uma gema, que provêm grandes poderes a pessoa que a utiliza, encontra sua cidade sendo ameaçada por terríveis monstros vindos de outra dimensão, com objetivo de tomar a cidade, e a utilizar como ponto inicial para destruir a humanidade.

O herói então provido de uma fúria para salvar o mundo tentará deter essa invasão. Com a ajuda do ancião da cidade, grande mestre mago, que o guiou anteriormente na sua jornada em busca da gema. E que o guia novamente com missões que o permitiram controlar e ampliar seus poderes. Para que assim ao final ele possa enfrentar o mal que será libertado, caso ele não consiga impedir a invasão, destruindo os cristais dos três elementos, que abrem o portal para invasão, antes que eles alcancem o que está escondido nas profundezas da cidade.

4.1.2 Aspectos fundamentais

Os aspectos fundamentais do jogo estão em volta da dinâmica de um *RPG* no estilo *Hack'nSlash*[15]. Onde o jogador deverá mostras suas habilidades no controle do herói para salvar sua cidade natal e o mundo da invasão que está ocorrendo.

Logo o envolvimento sentimental com tudo que acontece dentro do jogo, fundamentam todo ambiente de ação e aventura perante os desafios propostos, de quebrar os cristais dos três elementos, protegidos cada qual pelos seus guardiões elementais.

4.1.3 Diferencial

O jogo em si não apresentar diferencial quanto ao seu desenvolvimento ou técnicas revolucionarias, seu diferencial se dá pela sua história envolvente e a trama que muda conforme o jogador avança no game.

---

[15] *Hack'nSlash:* estilo de jogo que enfatiza o combate e reflexos rápidos, contendo elementos RPG, com objetivo limpar a sala e coletar itens para estar conseguir finalizar o jogo. (NOVAK, 2014, tradução nossa)

4.1.4 Estilo/Gênero

O game apresenta dois estilos principais que o caracterizam de forma geral o estilo RGP em conjunto com o *Hack'nSlash*.

4.2 CONTEXTO DO GAME

Nesta seção é descrito o mundo que rodeia o game, é a contextualização do game, ou seja, onde o jogador irá ter toda a ação, história do jogo.

4.2.1 Cenário

O ambiente do jogo se passa na idade média. Construções medievais, de pedra, castelos.

4.2.2 Historia do game

A aventura começa com o jogador escolhendo seu caminho, sua gema. Conforme a escolhe este seguirá o caminho do arqueiro, guerreiro, ou mago. Assim sendo o herói chega a sua cidade onde a encontra em perigo. Lá ele encontra seu mestre que anos antes havia lhe dado a missão de encontrar uma gema, que concederia grande poder a seu detentor. Ele lhe explica a situação que se encontra a cidade, o mundo e o universo, pois monstros de outra

dimensão encontraram uma forma de atravessar a passagem e chegar ao planeta com objetivo de soltar o antigo grande mal aprisionado nas profundezas da cidade. O grande mestre mago então dá ao herói uma nova missão salvar não apenas a cidade, mas o planeta e o universo que agora estão em perigo.

Em meio a esta aventura que está por começar o grande mestre sabe que o jovem herói ainda não está preparado para tal perigo. Porém, não há escolha, e ele revela como o herói deve proceder. O jovem herói deve destruir os três cristais que permitem aos monstros comandados por Hakkan, entrarem no mundo. Os cristais que representam os três elementos protegidos pelos seus respectivos guardiões Fogo, Terra, Água. O grande mestre então guia o herói nessa jornada, mas para chegar aos cristais ele precisa passar antes pelos monstros que também os defendem.

A medida em que o jovem herói consegue passar pelos desafios, e destruir os cristais algo estranho acontece: a cada cristal que ele destrói os monstros ficam mais fortes, devido a energia liberada por elemento. Mas a medida em que nosso jovem herói passa pelos desafios e monstros ele também se torna mais forte.

Assim que o herói consegue com grande dificuldade destruir os três cristais, algo imprevisível ocorre, ao contrário do que o mestre mago lhe contara ele não impede a invasão mas sim, liberta o grande mal. Em busca de respostas ele retorna ao mestre. Lá então lhe é revelado: o grande mestre estava na verdade todo tempo sendo controlado pelo terrível Mordecai com objetivo de libertar o mal. Lhe é revelado a verdadeira história que só um herói de coração puro seria capaz de quebrar os cristais e libertar Mordecai.

O jovem herói então sai correndo vendo o mal que acabara de fazer, até que então uma visão do verdadeiro mestre mago o chama. Ele então vai até ela pedindo perdão pelo que acabara de fazer, o mestre então lhe diz que “o que feito esta não importa, o que você fara a partir de agora é o que realmente importa”, então o mago lhe convence a parar de chorar e correr do havia feito e voltar e lutar pela humanidade e o universo.

O jovem herói volta correndo para a cidade, onde encontra destruída e repleta de monstros, mas sem medo ele vai enfrente derrotando todos aqueles que o tentam parar. Ao chegar no santuário onde o mal está a sugar energia do planeta para tornar a vida novamente, ele deve então enfrentar o seu mestre que ainda está sobe controle de Mordecai, após isto ainda tem que derrotar o comandante Hakkan para conseguir alcançar Infernus (o monumento

criado por Hakkan para libertar Mordecai e sugar toda a vida no planeta, com objetivo de libertar as forças do mal). Entretanto, após conseguir derrotar Hakkan e seguir ao Infernus ele chega tarde, pois o mesmo já está totalmente pronto e Mordecai libertado, mas mesmo assim, nosso herói. Sem medo vai o enfrentar. Mordecai é muito superior aos poderes do jovem héroi, sem nenhuma chance de vitória o herói cai e sua gema é destruída, e de repente o espírito de seu mestre aparece, o herói diz que agora não possui mas nenhum poder sua gema fora destruída, o mestre então lhe diz que para não desistir e confiar em si e não apenas na gema, que o verdadeiro poder está dentro dele, incumbido de coragem e força o herói então desfere um golpe com extremo poder que consegue destruindo ambos Mordecai e o Infernus.

4.2.3 Eventos anteriores

Os eventos anteriores a história do jogo é o que explica como nosso herói consegue seus poderes os quais utilizara para enfrentar os desafios da aventura do jogo.

Esses eventos ocorrem cerca de cinco anos antes da história do jogo, aonde o jovem herói guiado por seu mestre tem a missão de ir atrás de uma misteriosa gema descrita em uma profecia, nesta profecia o detentor desta gema ganharia poderes os quais um dia poderiam trazer a paz ao mundo salvando-o de algo misterioso descrito apenas como uma grande sombra.

Após essa busca que durou cerca de quatro anos o herói então antes de retornar a sua cidade passa um ano aprendendo a dominar os poderes em questão, definido pela cor da gema encontrada o qual é escolhido pelo jogador ao iniciar o jogo.

4.3 OBJETOS ESSENCIAIS DO GAME

Esta seção descreve os objetos que aparecem no game e que possuem um papel, ou seja, que afetem ao *gameplay*, que façam parte da experiência do jogo.

4.3.1 Personagens

O jogo tem como base 6 personagens, que são descritos a seguir:

Rei Alagos: Rei do reino de Tamerlane, em que se passa a trama do jogo, possui um grande exército e uma força sobre humana. Porém seu ponto fraco é sua filha Elina.

Princesa Elina: Elina é a filha única do Rei Alagos, ela se sente sufocada com o excesso de proteção que seu pai lhe obriga a ter, incluindo guarda-costas e soldados disfarçados que devem sempre a acompanhar.

Hakkan: Hakkan é um antigo súdito do mal supremo, que após sua queda, se apoderou de seus exércitos e busca incansavelmente despertar e libertar seu mestre.

Mordecai: Há muito tempo, Mordecai era um dos únicos três mestres magos existentes no planeta, porém uma força descontrolada se apoderou de sua mente fazendo-o assassinar um dos seus irmãos mestres e se tornar o Mal Supremo, que foi aprisionado no subterrâneo da cidade pelo Mestre Mago Elazar.

Elazar: Elazar é o último mestre mago restante e de coração bom no planeta, foi o responsável pela queda de Mordecai após sua tentativa frustrada de tomar o reino de

Tamerlane e assim se tornar o homem mais poderoso do mundo. Na trama do jogo Elazar é o responsável por treinar o Herói Brokk, pois sabe que forçar sombrias estão a cercar os muros de Tamerlane.

Brokk: Brokk é um jovem plebeu do reino de Tamerlane que foi escolhido por Elazar para ser seu aprendiz, pois segundo o mestre mago, Brokk tinha uma força interior que nenhum de seus irmãos, nem mesmo ele havia visto na humanidade. No Entanto, Brokk deve decidir como ele quer ser treinado e seguir sua aventura: se como Mago, Arqueiro ou Guerreiro.

4.3.2 Habilidades

Nas primeiras fases de desenvolvimento do jogo, nenhuma habilidade especial é adicionada ao personagem, apenas seus ataques básicos para a classe que o jogador desejar jogar:

- Mago: Projétil Místico.
- Arqueiro: Lançar Flecha.
- Guerreiro: Combate corpo a corpo com Espada.

Conforme o jogador for evoluindo no jogos habilidades especiais para cada classe serão adquiridas.

4.3.3 Armas

De início o jogador apenas tem a arma inicial da classe que o mesmo escolheu para jogar:

- Mago: Cajado.
- Arqueiro: Arco e Flecha.
- Guerreiro: Espada.

E conforme o jogador avança, matando os inimigos, estes podem deixar cair melhores itens com *status* melhores.

4.3.4 Estruturas

As estruturas principais do jogo são descritas como:

- Checkpoints: locais onde o jogador renasce após morrer.
- Cristais: estruturas principais da história que provocam os eventos relacionados a história.
- Vila: local onde encontram-se npcs da história e venda de equipamentos.

4.4 CONFLITOS E SOLUÇÕES

Nesta seção são descritos aspectos relacionados as iterações entre entidades do jogo, ou seja, como uma ação é feita, por exemplo, pelo jogador e o que acontece em seguida.

No jogo referente a esta monografia, os conflitos e soluções foram relacionados como:

- Movimentação o jogador se movimenta livremente pelo mapa sendo apenas limitado pelo tamanho do mapa e das estruturas presentes no mesmo.
- Ataque básico, o qual será o ataque normal do jogador ele possui um dano baseado em seus atributos e equipamentos, o qual produz um som específico e animação, este caso acertando um inimigo reduz a vida do mesmo pelo dano calculado.
- Ataque mágico, cada classe distinta possui habilidades que custaram mana e causaram maior dano, cada uma com sua própria fórmula de dano baseado novamente nos atributos e itens do personagem, e assim como o ataque básico possui animação e sons específicos.
- Inventario o jogador pode utilizar de um inventário o qual possuirá os itens coletados durante o jogo e os itens que atualmente estão equipados no personagem.

4.5 INTELIGENCIA ARTIFICIAL

A inteligência artificial utilizada no jogo, é feita com técnicas que tem como objetivo controlar os inimigos presentes do jogo e suas ações. Foram então usadas duas formas de controle, movimentos padronizados e previamente mapeados como movimentação dos inimigos enquanto em estado de não perseguição, e uma segunda técnica que se pode caracterizar como uma técnica de *GameIA* chamada *cheating*, onde o jogo passa aos outros objetos do game como os inimigos/estruturas a posição do jogador, e assim promove a movimentação caso seja um inimigo, por exemplo, em direção a posição do jogador caso esteja em sua área de atuação assim mudando o seu estado para perseguindo por exemplo.

4.6 FLUXO DO GAME

O fluxo do jogo é iniciado pela escolha do personagem, progressão de níveis durante o caminho até os três cristais serem destruídos, lutas com os chefes de cada cristal guardiões representantes de cada elemento, luta contra o seu mestre que está sendo controlado pelo grande mal, a luta contra o chefe inimigo, a luta contra o mal libertado e a destruição do monumento que absorver energia do planeta, finalizando o jogo e dando a possibilidade de recomeço da história com maior dificuldade..

4.7 GAMEPLAY / CONTROLES

No dispositivo móvel haverá seta no canto esquerdo da tela que servem para movimentação do personagem, do lado direito há 4 botões de ações ataque, habilidade, inventário, poção.

4.8 INTERFACE

A interface do game é basicamente composta pelo jogador sempre no centro da tela, no canto inferior esquerdo só dispositivo botões direcionais dos movimentos do jogador direita, cima, baixo e esquerda, e no canto inferior direito botões para ataque básico, inventário, botão da habilidade secundária, e mapa.

4.9 ÁUDIO

O jogo tem uma música de fundo, e 9 sons básicos:

1. Ataque do personagem(Cada Personagem possui um som).
2. Ataque do Monstro. (Dano no personagem).
3. Morte de Monstro.
4. Morte do Personagem.
5. Habilidades
6. Poção

4.10 REFERENCIAS DO GAME

No jogo desenvolvido para está monografia foi de maneira geral levado como principal referência os jogos de RPGs e MMORPG em geral, e *Hack'nSlash* com destaque aos jogos da série Diablo de 1996 da produtora *Blizzard Entertainment,* Ragnarok Online de 2002 da produtora *Gravity Corporation* e PathOfExile de 2013 da produtora *Grinding Gear Games*.

5 MODELAGEM DO SISTEMA

Neste capítulo é apresentado uma série de informações, definições e diagramas relacionados a modelagem do sistema desenvolvido, ou seja, do jogo desenvolvido para esta monografia.

Logo são apresentados os requisitos do sistema, requisitos estes funcionais e não funcionais, além de diagramas de atividade, de pacote, de classes referentes ao jogo desenvolvido.

5.1 LEVANTAMENTO DE REQUISITOS

O levantamento de requisitos é o momento aonde são definidos tudo aquilo que o software atende, o que ele deve fazer e respeitar.

Para Sommerville(2004, p. 82) “requisitos de sistema estabelecem detalhadamente as funções e as restrições de sistemas”. Para Paula Filho(2000, p. 13) “Os requisitos são as características que definem os critérios de aceitação de um produto”.

Nesta seção são apresentados os requisitos funcionais e os requisitos não funcionais referentes ao software desenvolvido, no caso desta monografia o jogo.

5.1.1 Requisitos Funcionais

De acordo com Sommerville (2004, p. 83) requisitos funcionais são “são declarações de funções que o sistema deve fornecer, como o sistema deve reagir a entradas

específicas e como deve se comportar em determinadas situações". Paula Filho (2000, p. 13) ainda completa "representam os comportamentos que um programa ou sistema deve apresentar diante de certas ações de seus usuários".

Na Tabela 1, apresenta se os requisitos funcionais definidos para o jogo desenvolvido nesta monografia.

Tabela 1 – Requisitos Funcionais

ID	NOME	DESCRIÇÃO
RF01	Selecionar personagem	O jogo deve permitir a seleção do personagem.
RF02	Movimentação	O jogo deve permitir através de direcionas presentes na tela a movimentação do personagem principal pelo mundo do jogo.
RF03	Vida	O jogo deve possuir um contador de vida do jogador.
RF04	Atacar	O jogo deve permitir a ação de atacar através de um botão na interface do jogo.
RF05	Poção	O jogo deve permitir a ação de utilizar poção através de um botão na interface do jogo.
RF06	Status	O jogo deve possuir status do personagem, que interferem diretamente no dano causado, e na quantidade de vida/mana do jogador.
RF07	Equipamentos	O jogo deve possuir uma área de equipamentos que estão sendo utilizados pelo personagem.
RF08	Inventario	O jogo deve possuir uma área de controle dos itens que o jogador possui.
RF09	Itens	O jogo deve possuir um sistema de itens conseguidos através de abates dos inimigos do jogo.
RF10	Experiencia	O jogo deve possuir um sistema de experiência para progressão do personagem através de abates de inimigos.
RF11	Nível do Personagem	O jogo deve possuir um sistema de nível de personagem e pontos de status para progressão do personagem.
RF12	Inimigos	O jogo deve possuir um sistema de inimigos presentes no mundo do jogo.
RF13	Chefes	O jogo deve possuir um sistema de chefes em determinadas localizações.
RF14	Inteligencia Artificial	O jogo deve possuir um sistema de inteligência artificial que controle movimentação e ataques de inimigos presentes no jogo.

RF15	Progressão de dificuldade	O jogo deve possuir um sistema de progressão conforme nível do personagem para balancear os inimigos e manter o desafio conforme se progrida no jogo.
RF16	Sem Fim	O jogo deve possuir um sistema aonde permita o jogador após conclusão da história central do jogo retorne ao início do mesmo para recomeçá-la com maior dificuldade.

Fonte: Elaboração dos autores, 2013.

Estes foram os requisitos funcionais levantados para se alcança os objetivos propostos para o jogo desenvolvido nesta monografia.

5.1.2 Requisitos Não Funcionais

Conforme descrito por Sommerville (2004, p. 83) os requisitos não funcionais são "restrições sobre os serviços ou as funções oferecidas pelo sistema. Entre eles destacam-se restrições de tempo, restrições sobre processo de desenvolvimento, padrões, entre outros". Paula Filho (2000, p. 13) ainda relata que são responsáveis por quantificar "determinados aspectos do comportamento" do sistema.

Na Tabela 2 são apresentados os requisitos não funcionais referentes ao jogo desenvolvido nesta monografia.

Tabela 2 – Requisitos Não Funcionais

ID	NOME	DESCRIÇÃO
RNF01	Linguagem	Linguagem utilizada Java.
RNF02	Plataforma	Plataforma de destino Android.
RNF03	Biblioteca Gráfica	Utilização da biblioteca gráfica LibGDX.
RNF04	Execução	O jogo deve rodar liso sem travamentos.
RNF05	*Sprites*	O jogo utilizara de *sprites* da internet dando os devidos créditos a seus criadores.
RNF06	Quantidade de Itens	O jogo deve possuir um mínimo de três tipos de armas e,

		pelo menos, um tipo para os outros equipamentos como armaduras.
RNF07	Quantidade de Inimigos	O jogo deve possuir um mínimo de quatro diferentes tipos de inimigos.
RNF08	Quantidade de Chefes	O jogo deve possuir um mínimo de quatro diferentes chefes.
RNF09	Desafio	O jogo deve manter os desafios sempre balanceados conforme progressão do jogador.
RNF10	Controle	O jogo deve ser controlado por *touchscreen.*

Fonte: Elaboração dos autores, 2013.

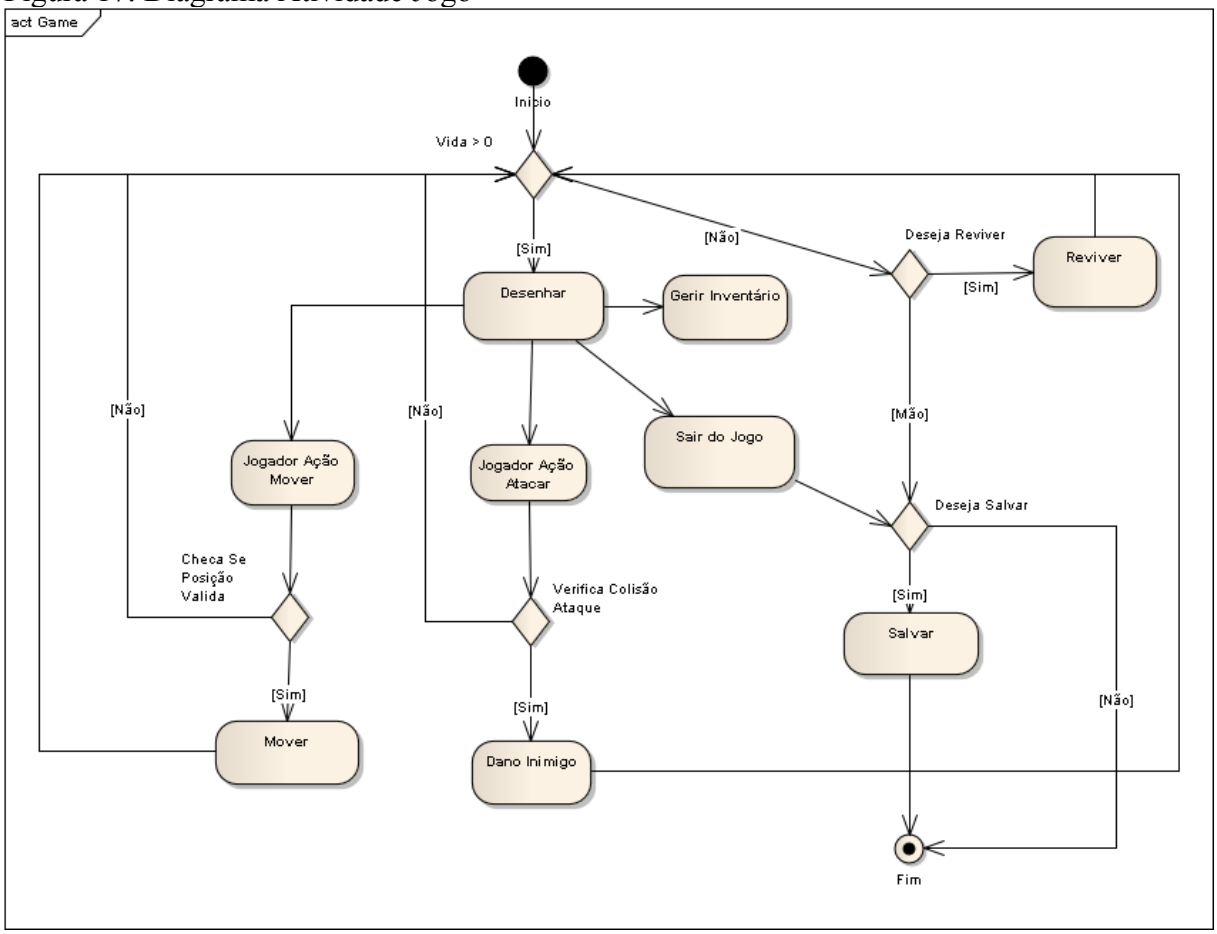
Logo, estes são os requisitos não funcionais levantados para se chegar aos objetivos propostos para o jogo desenvolvido nesta monografia.

5.2 DIAGRAMAS DE ATIVIDADES

Para Paula Filho (2000, p. 38) diagrama de atividade é uma "variante dos fluxogramas, sendo geralmente usado para descrever processos de negócio. Os fluxos podem ser encarados como processos de negócio dos desenvolvedores de software".

Sendo assim, é apresentado na Figura 17 o diagrama de atividade correspondente ao funcionamento do jogo desenvolvido.

Figura 17. Diagrama Atividade Jogo

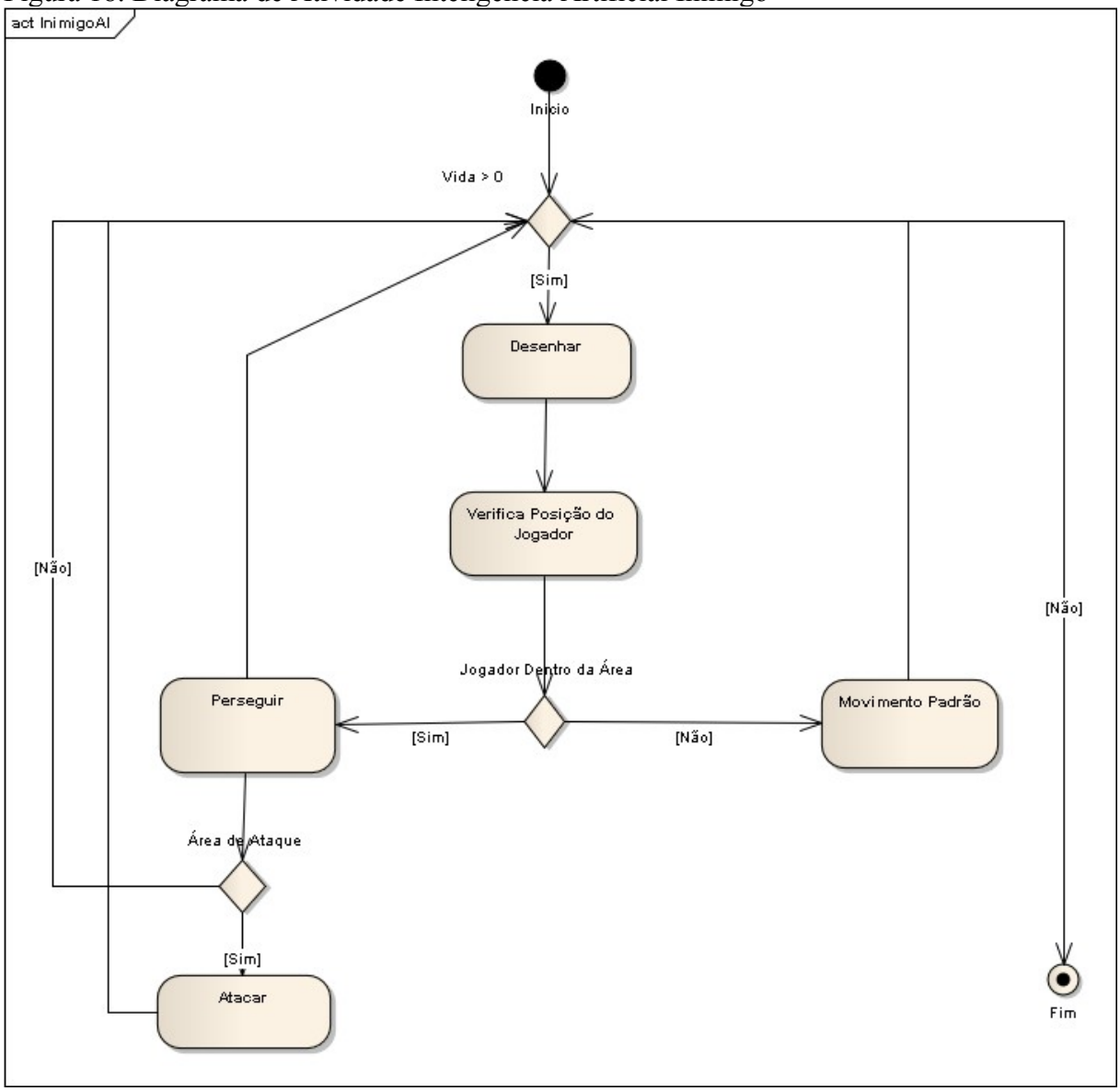


Fonte: Elaboração dos autores, 2013.

O diagrama mostrado corresponde ao funcionamento do jogo, aonde tudo gira em torno da vida do jogador, que se for maior que zero ele desenha o personagem e consequentemente tudo em voltar, permitindo assim todas as outras ações do jogo, sendo encerrado apenas através da vontade do jogador por meio de três possíveis maneiras, a morte do personagem, salvamento do jogo e encerramento, segundo maneira salvamento e encerramento do jogo, ou apenas encerramento do jogo sem salvar.

Já na Figura 18 é apresentado o diagrama de atividade relativo ao funcionamento da inteligência artificial dos inimigos presentes no jogo.

Figura 18. Diagrama de Atividade Inteligência Artificial Inimigo

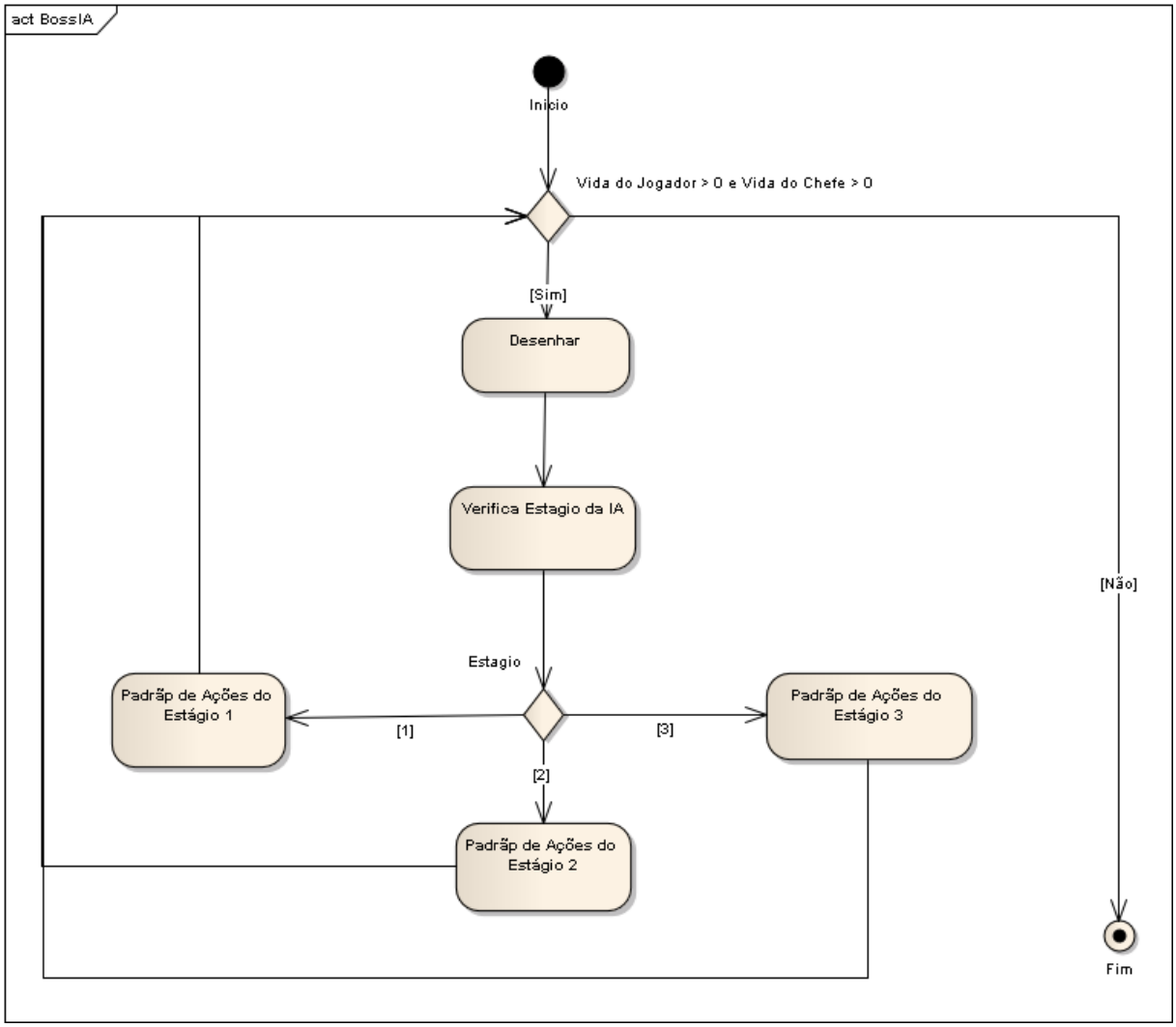


Fonte: Elaboração dos autores, 2013.

O diagrama apresentado relativo a IA do inimigo, refere-se basicamente a sua existência, no caso se sua vida for maior que zero ocorre a sua programação, que é definida como duas possíveis iterações, uma aonde realiza uma movimentação pre definida caso o jogador esteja fora de área de visão, e a outa é definida a perseguição ao jogador dentro dessa área, sendo finalizada somente se sua vida alcançar zero.

Outro importante ponto relativo ao jogo podes ser visto na Figura 19, aonde é representado a IA da luta dos chefões.

Figura 19. Diagrama de Atividade Inteligência Artificial do Chefe



Fonte: Elaboração dos autores, 2013.

Diferentemente da inteligência artificial de inimigos normais dentro do jogo, quando se trabalha com chefões deve-se ter o cuidado de que a batalha seja ao mesmo tempo desafiadora e possível de sucesso para o jogador, logo nesses casos a estratégia adotada é a definição de padrões de movimentos/ações desses chefes, como visto na Figura 19, toda vez é feita a verificação da vida do chefe e do jogador, e caso estejam acima de zero ocorre sua programação para ver em que estagio ela está, e realizando o que está programado, assim definindo exatamente de que forma a batalha ocorrera, não controlando o resultado mas permitindo ao jogador a chance de ganhar e perder dependendo de sua habilidade para contornar as ações do chefe e fazer com que sua vida chegue a zero, proporcionando assim

uma jogabilidade melhor e mais estável da batalha removendo o fator de aleatoriedade da movimentação/ações e a padronizando.

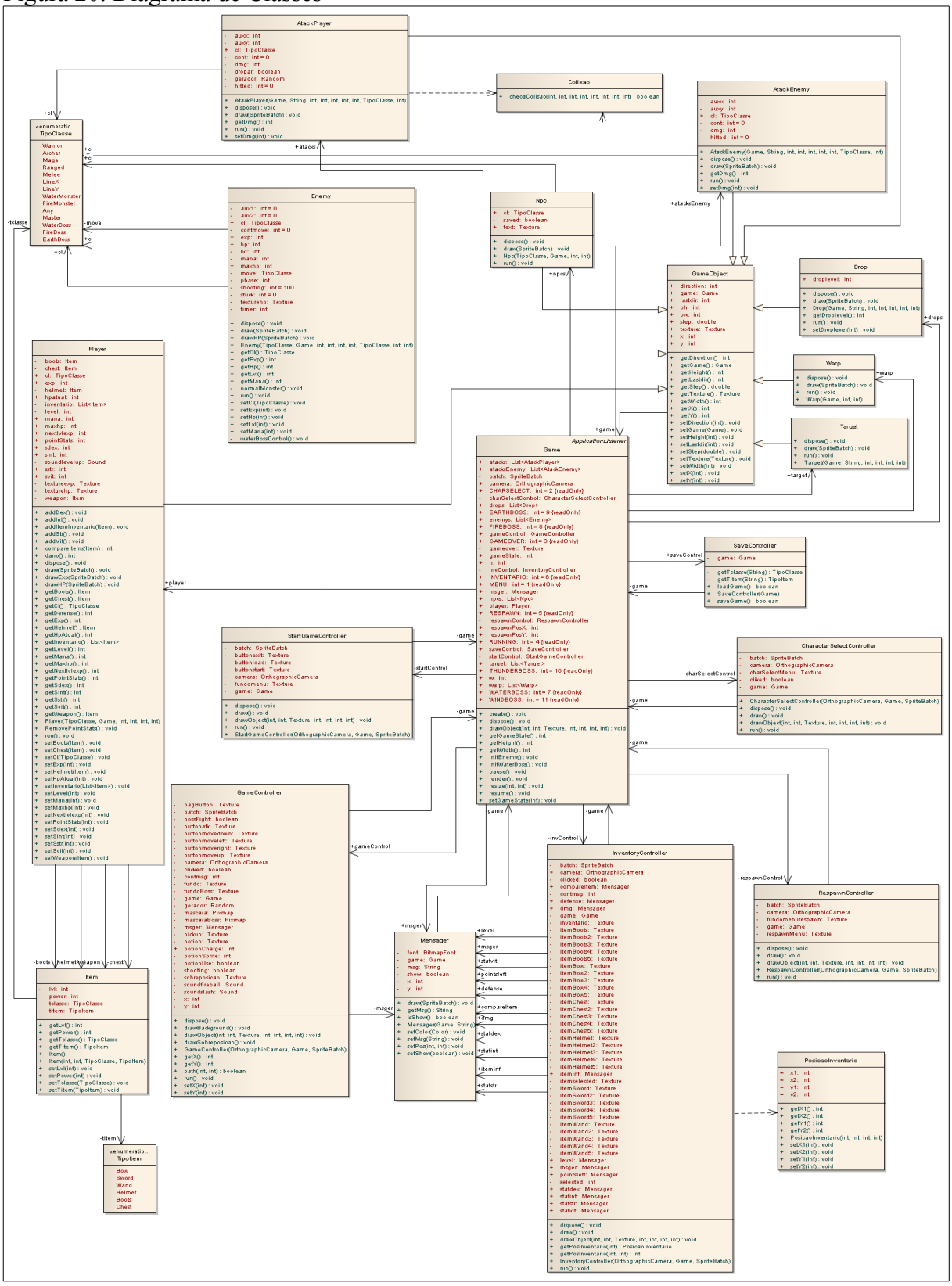
A aleatoriedade sim pode ser usado para escolher qual ação ele executara, porém que essas ações sejam controladas para não tornar uma batalha impossível, de sorte, ou fácil de mais para definir se o jogador vence ou não, além de que quando padronizamos a movimentação/ação desse chefe permitimos que caso o jogador falhe, em seu próximo encontro com o esse chefe este esteja mais preparado e tenha mais chances de sucesso.

5.3 DIAGRAMA DE CLASSES

Segundo Booch et al. (2000 p.96) “Um diagrama de classe mostra um conjunto de classes, interfaces e colaborações e seus relacionamentos. Os diagramas de classes são os diagramas mais encontrados em sistemas de modelagem orientados a objetos.”.

Logo na Figura 20 é apresentado o diagrama de classe do projeto desenvolvido, e em seguida este é apresentado em partes para melhor explicação e visualização do mesmo.

Figura 20. Diagrama de Classes



Fonte: Elaboração dos autores, 2013.

No diagrama apresentado anteriormente, é mostrado o digrama de classe referente ao jogo, aonde a classe Game responsável pelo controle geral do jogo, é nesta classe que é controlado o estado em que o jogo está, chamando o controle especifico para cada estado do jogo, e nela também aonde estão presentes os campos referentes ao jogador, inimigos, ataques de ambos jogador e inimigos, último *respawn* entre outros campos das diversas funções do jogo. Como pode ser observado melhor na Figura 21.

Figura 21. Classe Game

Fonte: Elaboração dos autores, 2013.

Outras estruturas presentes no diagrama de classe visto na Figura 20 estão o pacote de controle, ou as classes de controle do jogo, nela estão como o jogo deve se

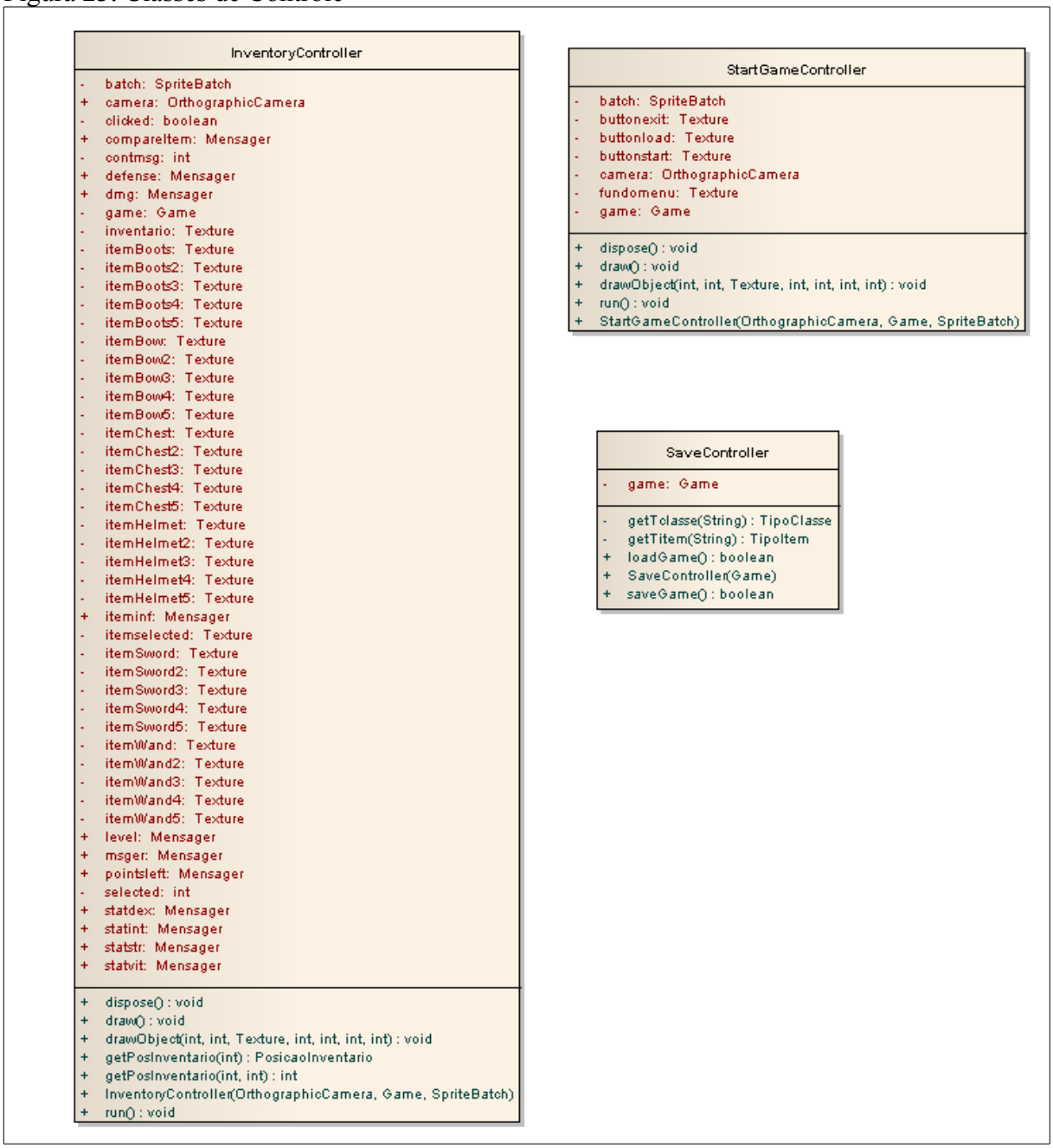
comportar e o que deve fazer em cada momento, desde a tela do *start game*, passando pela escolha da classe, o jogo, ate quando o jogador morre, salva, ou da *load game* em seu jogo salvo, essas classes podem ser melhor a seguir nas Figuras 22 e 23.

Figura 22. Classes de Controle

Fonte: Elaboração dos autores, 2013.

Na Figura 22 estão presentes as classes responsáveis por controlar todas as funções de *respawn*, seleção de personagem e do jogo e a seguir na Figura 23 estão presentes as classes responsáveis pelo controle da tela inicial de *startgame*, de controle do inventário e a de salvar o jogo.

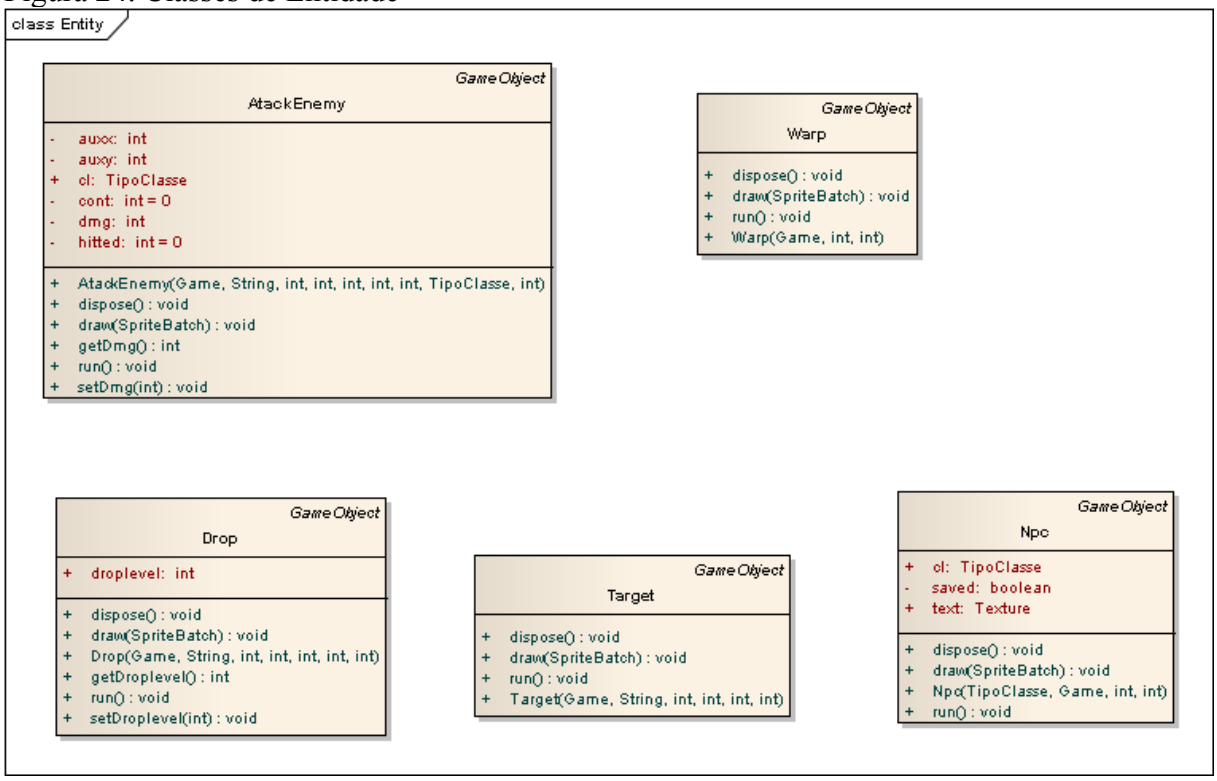
Figura 23. Classes de Controle



Fonte: Elaboração dos autores, 2013.

Também presentes no diagrama de classes estão as classes entidade do projeto, as classes que representam tudo aquilo que está dentro do jogo, e que podem ser observadas nas Figuras 24,25 e 26.

Figura 24. Classes de Entidade



Fonte: Elaboração dos autores, 2013.

Na Figura 24 estão presentes as classes entidades referentes aos ataques dos inimigos, uma estrutura de teleporte utilizada para encontros com os chefes, os *drops* que são o baú dos inimigos derrotados que em um segundo momento são gerados os itens a partir dele quando o jogador as pegas, o *target* que são alvos utilizados nas batalhas de chefes como elementos para um ataque teleguiado que persegue o jogador por certo tempo e os *npcs* que são personagem que realizam alguma iteração do com o jogador como salvar.

Em seguida na Figura 25 as entidades do jogo que são apresentadas, o *player* a principal objeto do jogo refere-se ao jogador, seu dano, localização, itens, posição, etc, assim como a classe de ataque do jogador que trata os ataques e quem acerta assim como o dano causado ao inimigo que é acertado o qual é baseado e calculado na classe do *player*, além da classe referente ao inimigo que trabalha como a do *player* porém refente ao inimigo, cuja diferença é que não possui itens.

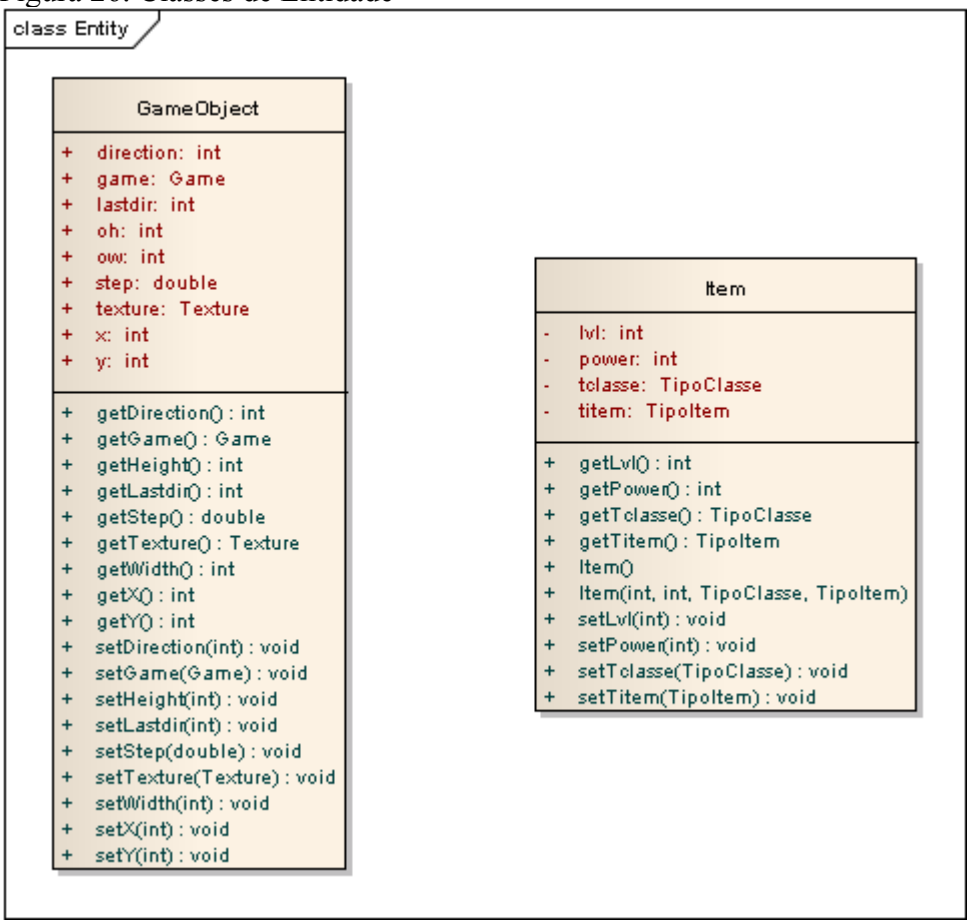
Figura 25. Classes de Entidade

Fonte: Elaboração dos autores, 2013.

Anteriormente foram apresentadas a maior parte das entidades do jogo estas por sua vez tem como classe base a entidade GameObject aonde herdam ela, cujo possui a base do posicionamento, direção para movimentação e direção dos ataques, e a base para utilização

das *sprites* como tamanho, largura, além da classe referente aos itens que são utilizados pela classe do jogador.

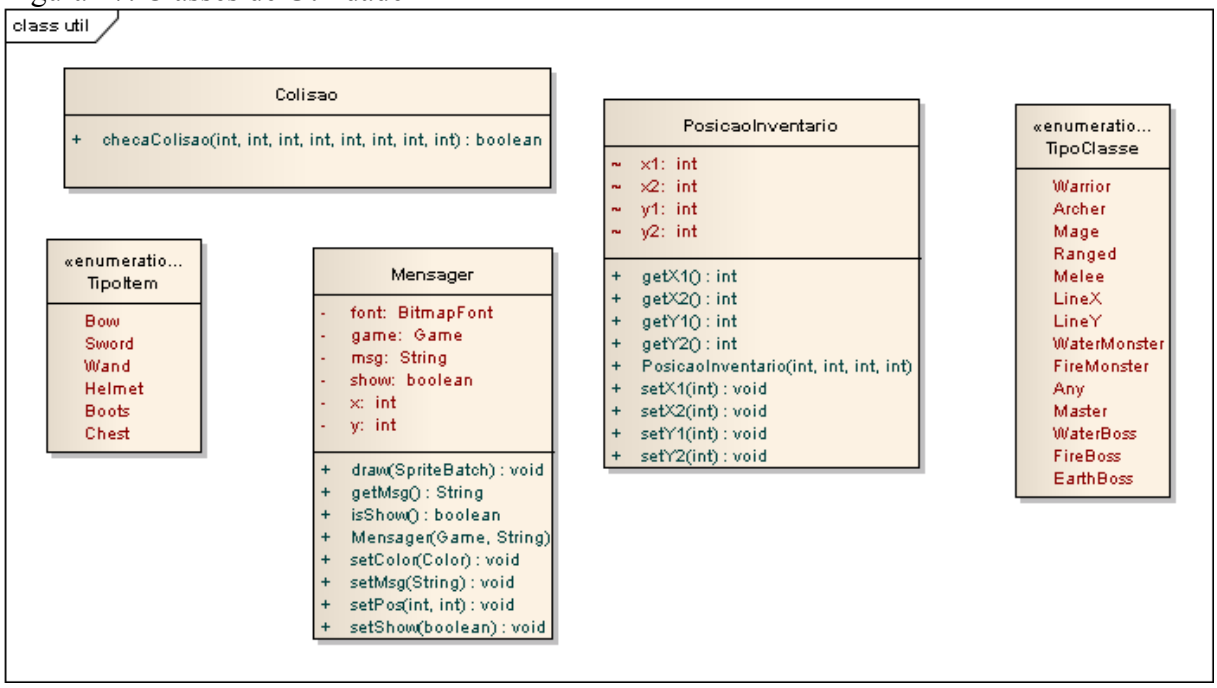
Figura 26. Classes de Entidade



Fonte: Elaboração dos autores, 2013.

E por último é apresentado na Figura 27 as classes do pacote de utilidade, são as classes com estruturas utilizadas para alguma funcionalidade específica, como a colisão utilizada nas classes de ataque e inimigo para movimentação, a classe utilizada para auxiliar no posicionamento das *sprites* no inventário, duas classes *enum* para nomear os tipos de itens e outros objetos do jogo e a classe *mensager* responsável pela escrita nas telas como inventário e alguns textos durante o jogo.

Figura 27. Classes de Utilidade



Fonte: Elaboração dos autores, 2013.

Logo, basicamente esta foi a estrutura das classes de modo geral, definidas para projeto refente a esta monografia.

6 IMPLEMENTAÇÃO / AVALIAÇÃO

Ao longo desta monografia foram apresentados os processos e elementos para elaboração de um jogo digital para dispositivos móveis, entre eles as etapas, o GDD e a modelagem. Neste capítulo é apresentado sua implementação, os testes e a avaliação, relativo a tudo apresentado ate agora.

São descritos neste capítulo o desenvolvimento do jogo, as tecnologias usadas, o processo de desenvolvimento, a lógica por trás dos elementos do jogo, as fases do desenvolvimento, elaboração/edição dos elementos audiovisuais, a fase de testes e a avaliação do jogo.

6.1 TECNOLOGIAS UTILIZADAS

O desenvolvimento de um jogo digital para dispositivos móveis exigi um vasto número de recursos e ferramentas tecnológicas para sua elaboração. A seguir na Figura 28 é apresentado de maneira geral todos os *softwares* e tecnologias utilizadas no desenvolvimento deste projeto.

Figura 28. Tecnologias Utilizadas

Fonte: Elaboração dos autores, 2013.

Na figura anterior são apresentados os programas e tecnologias utilizadas no projeto, são elas, a linguagem de programação Java, a ferramenta de desenvolvimento Eclipse em composição do ADT(*Android Developer Tools*) e Android SDK, a ferramenta de edição e criação de imagens GIMP, o programa de modelagem EA(*Enterprise Archithect*), a biblioteca gráfica LibGDX, a ferramenta online de geração de sons as3sfxr, e o programa utilizado para edição de imagens para o mapa do jogo Tiled.

Algumas dessas ferramentas já foram relatadas nesta monografia e não são novamente citadas neste capítulo, outras como GIMP e as3sfxr também não são necessárias maiores informações pois são apenas ferramentas edições de imagem/som.

6.1.1 Eclipse

Segundo NETO(2009), o Eclipse é uma ferramenta muito poderosa e totalmente gratuita para desenvolvimento Java, lançada pela Eclipse.org em parceria com a IBM e outros fornecedores de produtos para Java. Esta distribuição é perfeitamente legal, visto que a Eclipse.org não cobra licença de uso para a utilização do Eclipse.

Usando o Eclipse como ferramenta IDE para desenvolvimento, não é preciso ficar recompilando as classes, pois sempre que uma classe ou interface alterada for salva, automaticamente o Eclipse irá recompilá-la e validar a sintaxe.(NETO,2009).

6.1.2 SDK Android

O Android SDK é o software utilizado para desenvolver aplicações no Android, que tem um emulador para simular o celular, ferramentas utilitárias e uma API completa para

a linguagem Java, com todas as classes necessárias para desenvolver as aplicações. (LECHETA,2013).

Segundo LECHETA(2013), embora o SDK tenha um emulador que pode ser executado como um aplicativo comum, existe um *plug-in* para o Eclipse que visa justamente integrar o ambiente de desenvolvimento Java com o Emulador, chamado ADT.

6.1.3 ADT

O ADT é um *plug-in* para a ferramenta Eclipse que integra o SDK Android e os emuladores de aparelhos ANDROID com a interface do Eclipse, visando a facilidade no desenvolvimento.

Segundo LECHETA(2013), Existe também a possibilidade de realizar o download do ADT *Bundle*, que é nada mais do que um Eclipse turbinado, já com o *plugin* ADT de desenvolvimento do Android configurado.

6.1.4 LibGDX

A LibGDX é um *framework* de desenvolvimento de jogos para Android, PC, HTML e IOS escrito em Java. As vantagens são o custo, pois é distribuído sobre a licença Apache 2.0, tem suporte para funcionalidades gráficas 2D/3D, de áudio, matemática/física e *touch screen*.(DETTENBORN,2014).

O primeiro passo para começar a desenvolver com o LibGDX é realizar o download do .jar no site oficial.

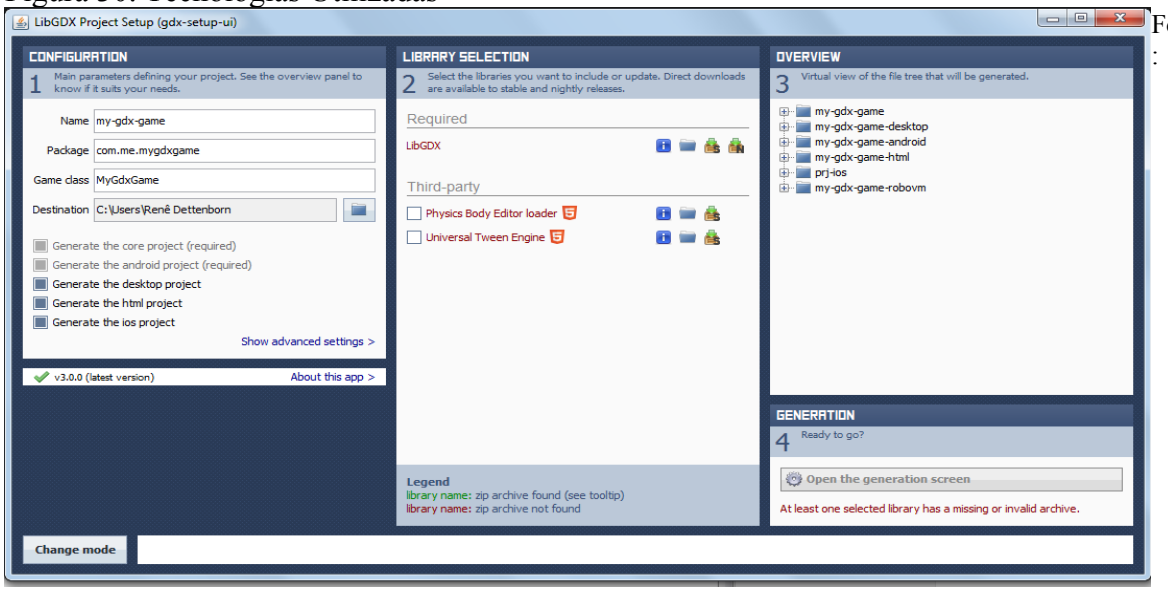
Em seguida descompacte o .zip e execute o gdx-setup-ui.jar, pois este executável cria projeto pré configurados para iniciar o desenvolvimento, seja no ADT *Bundle* ou no Eclipse configurado com o *Plugin* do ADT como pode ser visto na Figura 29.

Figura 29. Tecnologias Utilizadas

Fonte: http://renedet.blogspot.com.br/2014/01/tutorial-libgdx-configurar-ambiente-de.html.

Ao clicar no botão *Create*, deverá aparecer a seguinte tela conforme a Figura 30.

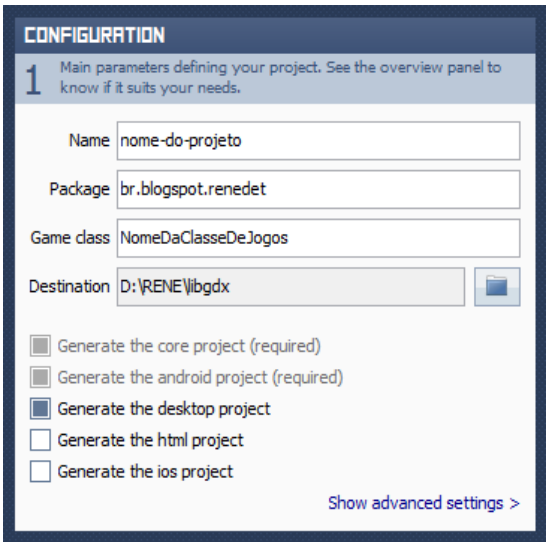
Figura 30. Tecnologias Utilizadas



Fonte: http://renedet.blogspot.com.br/2014/01/tutorial-libgdx-configurar-ambiente-de.html.

Segundo DETTENBORN(2014) no próximo passo, deve-se configurar algumas opções do projeto, conforme Figura 31 o exemplo, o mais importante é *Destination*(onde são criados os projetos) e quais os tipos de projetos que são criados.

Figura 31. Tecnologias Utilizadas



Fonte: http://renedet.blogspot.com.br/2014/01/tutorial-libgdx-configurar-ambiente-de.html.

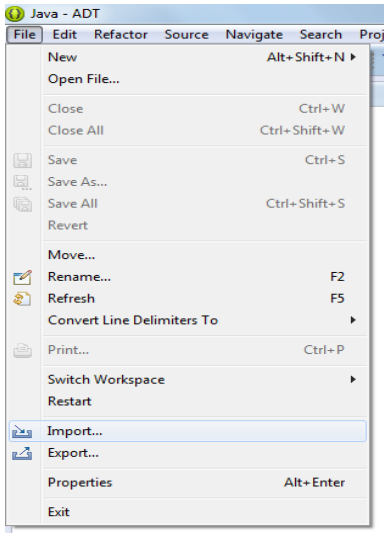
Na parte LIBRARY SELECTION deve-se informar onde está o .zip do libgdx como visto na Figura 32.

Figura 32. Tecnologias Utilizadas

Fonte: http://renedet.blogspot.com.br/2014/01/tutorial-libgdx-configurar-ambiente-de.html.

Após criação do projeto, é necessário importá-lo no Eclipse ou ADT *Bundle*. Na tela do Eclipse ou do ADT *Bundle* clique em FILE->Import, assim como pode ser observado na Figura 33.

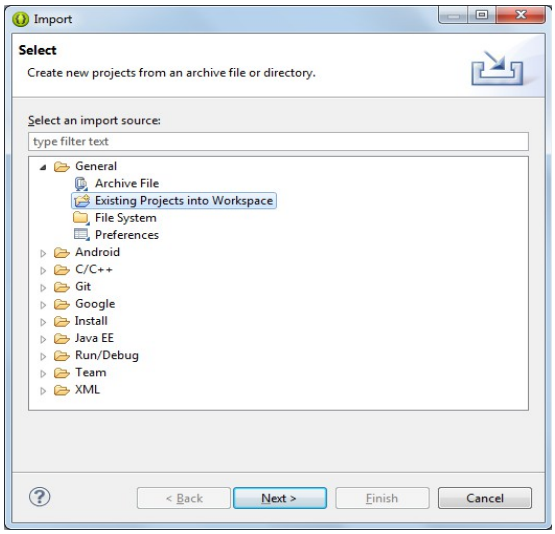
Figura 33. Tecnologias Utilizadas



Fonte: http://renedet.blogspot.com.br/2014/01/tutorial-libgdx-configurar-ambiente-de.html.

Na tela posterior selecione General e depois *Existing Projects into Workspace*, como na Figura 34:

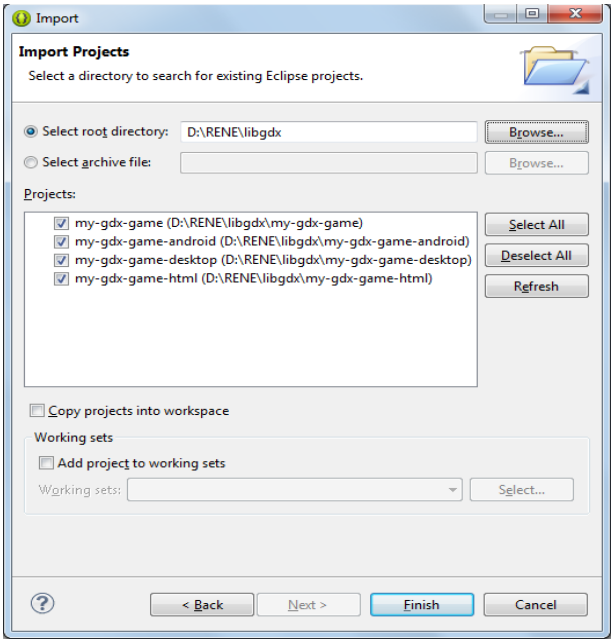
Figura 34. Tecnologias Utilizadas



Fonte: http://renedet.blogspot.com.br/2014/01/tutorial-libgdx-configurar-ambiente-de.html.

Na tela seguinte selecione o diretório em que os projetos foram criados, logo em seguida clique em Finish como na Figura 35.

Figura 35. Tecnologias Utilizadas



Fonte: http://renedet.blogspot.com.br/2014/01/tutorial-libgdx-configurar-ambiente-de.html.

Ao clicar em *finish* o eclipse importa os projetos necessários para o início do desenvolvimento de jogos para Android.

6.2 DESENVOLVIMENTO

Esta seção descreve o processo de desenvolvimento realizado na elaboração do jogo relativo a esta monografia, mostrando todo o processo de criação e desenvolvimento da estrutura do jogo assim como dificuldades e soluções dos problemas encontrados durante o processo. Ao fim é apresentado de forma breve, o jogo em seu formato final.

Para se iniciar o desenvolvimento de um jogo três estruturas bases são necessárias, o *draw*, ou seja, desenhar na tela o que acontece no jogo, a colisão e o fluxo do jogo. Apos ter conhecimento nessas três estruturas se começa então a pensar no resto do jogo.

6.2.1 Função Draw

No desenvolvimento de um jogo uma das estruturas mais importantes é a função que tem como objetivo desenhar o que ocorre no jogo para o jogador.

Nela é onde esta a parte do código responsável por ir a um arquivo pre definido relacionado a pôr exemplo o personagem principal e o desenhá-lo na tela. Nesta função deve-se definir seis importantes variáveis que são a posição x, a posição y, a altura da imagem a ser desenhada, a largura da imagem a ser desenhada, uma variável que determine qual pedaço da imagem é desenhado conforme a animação do andar por exemplo, uma variável responsável por definir qual trecho da imagem é utilizado conforme a direção que o personagem se move e uma variável para definir o arquivo de imagem/textura a utilizar.

Com a utilização da biblioteca LibGdx esta função requer uma variável a mais chamada *Spritebatch* utilizada pela biblioteca para realizar o desenhar na tela. Um exemplo desta função completa pode ser observado na Figura 36.

Figura 36. Função Draw

```
public void draw(SpriteBatch batch) {
        TextureRegion region = new TextureRegion(texture,step * ow, direction * oh, ow, oh);
        Sprite sprite = new Sprite(region);
        sprite.setPosition(x, y);
        sprite.draw(batch);
}
```

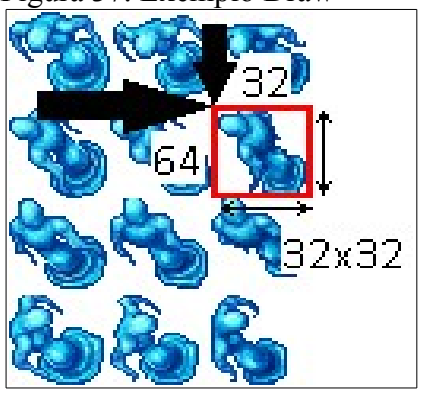
Fonte: Elaboração dos autores, 2013.

Como pode ser observado na figura 36, com a utilização da biblioteca LibGdx a função de desenhar na tela se torna bem simples, nela está presente a variável *region* que é a região na imagem a ser utilizada, esta variável é criada com base quatro outras variáveis que

são a texture uma variável a qual foi feita o *load* da imagem a ser utilizada, *step* a variável responsável por definir qual quado da imagem a animação está e multiplicado pela variável ow que representa o *object width* ou largura do objeto em questão assim determinando a exata posição horizontal do pedaço da imagem a ser utilizado, *direction* responsável pelo trecho da imagem será utilizado e multiplicado pela variável oh que representa o *object height* ou altura do objeto em questão assim determinando a exata posição vertical ou a linha da imagem a ser utilizado, além de novamente as variáveis ow e oh definindo a tamanho a ser pego da imagem. Posteriormente é definido então a *Sprite* o "pedaço" da imagem que vai ser desenhado, e logo em seguida é definida a posição dentro do jogo onde ela será desenhada e, por fim, utilizando o *draw* da própria biblioteca para desenhar a imagem no jogo.

Para exemplificar a ação de escolher qual "pedaço" da imagem será desenhado suponhamos querermos desenhas o monstro do elemento água cujo tamanho de AlturaXLargura é de 32x32, e este monstro está andando para esquerda representado pelo número 1, pois é utilizado o formato de 0 a 3, e esta em seu terceiro *frame* da animação de andar representado novamente pelo número 1, logo teremos 32x1 para step*ow, 32x2 para *direction**oh, 32 para largura e 32 para altura, conforme representado na Figura 37.

Figura 37. Exemplo Draw



Fonte: Elaboração dos autores, 2013.

Como visto a função de desenhar é simples, que requer apenas algum cuidado ao definir os valores das variáveis, sendo ela a estrutura que é provavelmente a mais utilizada em todo o desenvolvimento e execução do jogo.

6.2.2 Função Colisão

Ao desenvolver um jogo quer-se na maioria dos casos alguma forma de detectarmos se um elemento do jogo colidiu com outro elemento, como, por exemplo, um disparo do jogador acertar o inimigo.

Há diferentes formas de realizar este procedimento, o mais simples e utilizado no desenvolvimento deste projeto, foi a utilização de uma classe cujo único objetivo é possuir uma função para verificar se houve a colisão entre esses elementos.

Para que esta função funcione é necessário oito parâmetros de entrada, estes quatro referentes a um elemento e os outros quatro ao segundo elemento, estes parâmetros são as variáveis referentes ao posicionamento x e y e de tamanho, altura e largura ou h e w.

Basicamente a função representada na Figura 38 faz é pegar o posicionamento referente a um elemento definir seu tamanho e comparar com o mesmo referente ao outro elemento.

Figura 38. Função de Colisão

```
public boolean checaColisao(int obj1X, int obj1Y, int obj1W, int obj1H,
        int obj2X, int obj2Y, int obj2W, int obj2H) {
    if ((obj1X >= obj2X && obj1X <= obj2X + obj2W)
            && (obj1Y >= obj2Y && obj1Y <= obj2Y + obj2H)) {
        return true;
    } else if ((obj1X + obj1W >= obj2X && obj1X + obj1W <= obj2X + obj2W)
            && (obj1Y >= obj2Y && obj1Y <= obj2Y + obj2H)) {
        return true;
    } else if ((obj1X >= obj2X && obj1X <= obj2X + obj2W)
            && (obj1Y + obj1H >= obj2Y && obj1Y + obj1H <= obj2Y + obj2H)) {
        return true;
    } else if ((obj1X + obj1W >= obj2X && obj1X + obj1W <= obj2X + obj2W)
            && (obj1Y + obj1H >= obj2Y && obj1Y + obj1H <= obj2Y + obj2H)) {
        return true;
    } else {
        return false;
    }
}
```

Fonte: Elaboração dos autores, 2013.

Outro fator relacionado a colisão é que esta mesma função pode ser utilizada não somente como *hitdetection*, assim como utilizamos para ver se um elemento tocou/colidiu com outro podemos com um simples modificação ao enviar os parâmetros para esta função aumentarmos o campo de que queiramos verificar em um elemento para utilizarmos por exemplo como área de ação para o inimigo começar a perseguir o jogador por exemplo.

Neste caso em vez de enviar as exatas coordenadas e tamanho de um elemento deslocamos as variáveis x e y, por exemplo, para a esquerda e baixo em um valo de pôr exemplo -100 e dobramos esse valor e adicionamos a largura e altura criando assim uma área em volta do elemento e através da verificação de colisão nesta área em volta podemos iniciar a movimentação do inimigo em direção ao jogador. Pode se observar melhor na Figura 39 com ambos *hitdection* e área de perseguições representadas.

Figura 39. Exemplo de Colisão

Fonte: Elaboração dos autores, 2013.

Na Figura 39 é representado a colisão ou *hitdetection* do elemento com parâmetros sem alteração pela caixa de cor vermelha e em cor verde a representação da colisão para uma área de ação descrita anteriormente.

Esta função é de grande importância na maioria dos jogos desenvolvidos, a função apresentada nesta monografia é uma função simples de fácil entendimento e utilização, mas

existem outras formas para se calcular e verificar a colisão em jogos algumas um pouco mais complexas. Fica a cargo do desenvolvedor escolher qual utilizar.

6.2.3 Fluxo do Jogo

A terceira estrutura considerada fundamental para o desenvolvimento de um jogo é o que chamamos de fluxo do jogo, ela na verdade é composta de três diferentes métodos dentro do jogo quando trabalhamos com a biblioteca LibGdx, eles são o *create* onde assim que o jogo é executado esse método é chamada sendo ele apenas chamado uma única vez, normalmente utilizado para inicializar por exemplo o *load* das imagens/texturas do jogo, o *render* esse método é o responsável pelo jogo em si, é este método que é chamado constantemente durante a execução do jogo e é nele que se coloca todas as estruturas do jogo como por exemplo a verificação de que o jogador pressionou a tala *touch* em certa posição onde esta a função de ataque entre outras, e por último o método *dispose*, responsável por "limpar" as diversas variáveis que desenham na tela as imagens assim permitindo um novo desenhar sem o que havia nelas antes.

Como exemplo, pode ser observado na Figura 40 um *create* simples. Este é composto de w e h, variáveis que recebem respectivamente através da biblioteca o tamanho definido para a tela do jogo, câmera que é instanciada um tipo de câmera que é utilizada, *batch* que controla o que aparece na tela, logo em seguida são definidos a imagem/textura do fundo, a máscara que é usada pra determinar onde o jogador pode ou não andar, e a imagem/textura do jogador, em seguida é definido o tamanho do jogador como explicado na seção anterior do *draw* e a posição inicial do jogador x e y.

Figura 40. *Create*

```
public void create() {
    w = Gdx.graphics.getWidth();
    h = Gdx.graphics.getHeight();
    camera = new OrthographicCamera();
    camera.setToOrtho(false, w, h);
    batch = new SpriteBatch();

    fundoTexture = new Texture(Gdx.files.internal("fundo.png"));
    fundoTexture.setFilter(TextureFilter.Linear, TextureFilter.Linear);

    mascara = new Pixmap(Gdx.files.internal("mascara.png"));

    playerTexture = new Texture(Gdx.files.internal("player.png"));
    playerTexture.setFilter(TextureFilter.Linear, TextureFilter.Linear);
    ow = 32
    oh = 64
    x = 30;
    y = 240
}
```

Fonte: Elaboração dos autores, 2013.

Já o método render é um pouco mais difícil de se definir como pode ser observado na Figura 41, pois nele iram diversas funcionalidades do jogo e cada jogo é diferente, mas basicamente o que esta normalmente nele são a definição de onde será desenhado o fundo, as diversas funcionalidades representadas na figura como as funcionalidades de andar para cima e para baixo e uma condicional para resetar o *step* para primeira posição ao completar o ciclo da animação do andar, funções de limpar a tela, em seguida a definição do tipo de projeção definida para o 2d, logo após é definido o começo das funções para desenhar na tela seguido do chamado para o método *draw* e o fim da função para desenhar.

Figura 41. *Render*

```
public void render() {
    TextureRegion fundoRegion = new TextureRegion(fundoTexture, 0, 0, 800,600);
    Sprite fundoSprite = new Sprite(fundoRegion);
    fundoSprite.setPosition(0, 0);

    if (Gdx.input.isKeyPressed(Keys.UP)) {
        direction = 1;
        if (path(x + 16, y + oh)) {
            step++;
            y += 1;
        }
    }
    if (Gdx.input.isKeyPressed(Keys.DOWN)) {
        direction = 0;
        if (path(x + 16, y)) {
            step++;
            y -= 1;
        }
    }
    if (step >= 4) {
        step = 0;
    }
    Gdx.gl.glClearColor(1, 1, 1, 1);
    Gdx.gl.glClear(GL10.GL_COLOR_BUFFER_BIT);
    batch.setProjectionMatrix(camera.combined);
    batch.begin();
    fundoSprite.draw(batch);
    drawPersonagem(batch, x, y, step, direction);
    batch.end();
}
```

Fonte: Elaboração dos autores, 2013.

Para finalizar a estrutura base de um jogo, o método *dispose* função é de apenas limpar os recursos utilizados, logo cada recurso deve chamar a função *dispose* da biblioteca como pode ser visto na Figura 42.

Figura 42. *Dispose*

```
public void dispose() {
    batch.dispose();
    fundoTexture.dispose();
    avatarTexture.dispose();
}
```

Fonte: Elaboração dos autores, 2013.

Com isso a estrutura base de um jogo é definida, basta agora criar os outros elementos e recursos do jogo, estes dependentes do estilo do jogo e escolha de cada desenvolvedor.

Nas próximas seções são apresentados diversas funções presentes no jogo desenvolvido nesta monografia que podem ser utilizados de diferentes formas e em diversos projetos.

6.2.4 Caminho/*Path* do Mapa

Ao desenvolver um jogo deve-se utilizar alguma forma de definir por onde o jogador pode ou não andar, isso para que não atravesse paredes, árvores, etc, para que assim o jogo se torne um pouco mais realista.

Existem diferentes formas de se trabalhar para conseguir isto, com a utilização de mapeamento da área e definição de onde ele pode ou não passar, por exemplo, mas requer um grande esforço. A escolha para este projeto foi a utilização de uma máscara como exemplo na Figura 43, para o mapa do jogo onde se tem o mapa do jogo em apenas duas cores preto e branco, onde o branco refere-se ao caminho o qual o jogador pode andar e o preto onde há algum obstáculo, parede, árvore, etc.

Figura 43. Máscara

Fonte: Elaboração dos autores, 2013.

Logo com a utilização de uma máscara a essa definição de caminhos se torna mais fácil e simples. Mas para sua utilização precisa-se de um método capaz de interagir e dizer ao jogo se naquela posição com base na máscara o jogador pode ou não ir antes de ele atualmente ir a posição, esta função pode ser observada na Figura 44.

Figura 44. *Path*

```
public boolean path(int px, int py) {
        Color color = new Color();
        int valor = 0;
        valor = mascara.getPixel(px, mascara.getHeight() - py);
        Color.rgb888ToColor(color, valor);
        int r = (int) (color.r * 255f);
        int g = (int) (color.g * 255f);
        if ((r > 20) && (g > 20)) {
            return true;
        }
        return false;
}
```

Fonte: Elaboração dos autores, 2013.

O que acontece é que, para verificar se a próxima posição do *player* é permitida ou não, chama-se o método da Figura 37 toda vez em que o *player* for alterar suas coordenadas x e y, passando então a posição que este assumirá quando o jogador utiliza a movimentação, o método em questão então verifica se esta é uma posição valida verificando a cor correspondente a esta posição na máscara, informando assim se há algo no caminho ou não e permitindo assim a alteração da posição para as novas coordenadas x e y.

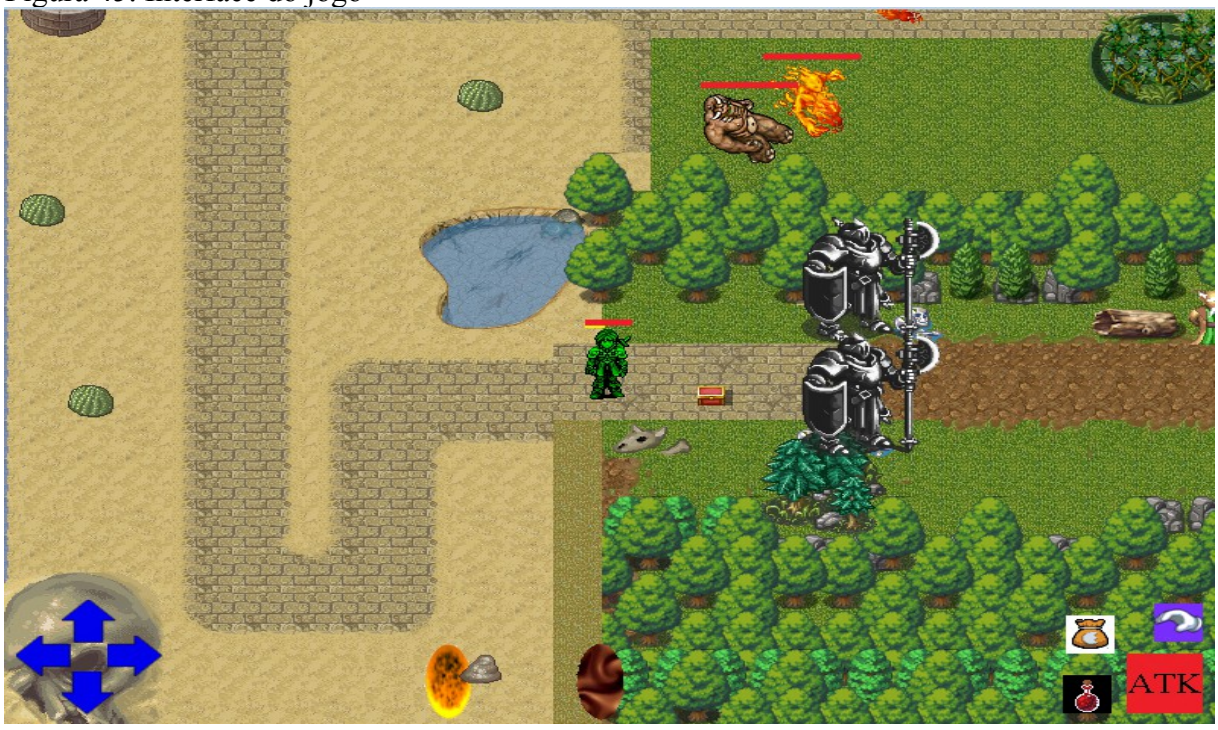
6.2.5 Controles do Jogo

A jogabilidade de todo os jogos é feita através de como o jogador interage com o game e como ele controlara seu personagem através dos desafios propostos durante o mesmo.

Há diferentes formas de realizar ou controlar como o personagem é controlado, em jogos onde há um controle físico os botões nele o fazem, quando se fala de jogos de computador mouse e os botões do teclado são comumente utilizados nos formatos WASD ou setas para movimentação e N outros comandos utilizados pelos mais variados estilos, porém ao tratamos de dispositivos móveis apenas possuímos duas formas para realizar este controle do personagem a primeira seria com o a utilização de acelerômetros e a outra o *touchscreen* do aparelho.

No desenvolvimento desta monografia foi definido que o jogo projetado seria um RPG *action*, logo a escolha pela utilização do *touch* foi obvia pela comodidade pra se controlar o personagem. Basicamente o que foi feito foi a utilização de elementos gráficos na tela definindo-se uma posição/local específico das funções que jogador teria, no caso representação gráfica de botões para movimentação cima, baixo, direita, esquerda, botões para ataque, poção, inventário e pegar itens do chão, como pode ser visto na interface do jogo na Figura 45.

Figura 45. Interface do jogo



Fonte: Elaboração dos autores, 2013.

Logo para a utilização desses tipos de estrutura a biblioteca nos permite verificar quando o jogador pressionou a tela e qual a exata posição, nos permitindo realizar a função que desejarmos para um certo ponto da tela, essa funcionalidade foi usada nesse projeto tanto para movimentação, e outras funções da interface, quanto para utilização do inventário de forma iterativa. A Figura 46 apresenta o trecho de código refente a funcionalidade do *touch* para movimentação.

Figura 46. Funcionamento do *Touch* Para Movimentação

```
if (Gdx.input.isTouched()) {
    Vector3 touchPos = new Vector3();
    touchPos.set(Gdx.input.getX(), Gdx.input.getY(), 0);
    game.camera.unproject(touchPos);
    // up
    if (touchPos.x >= 47 && touchPos.x <= 67 && touchPos.y >= 71 && touchPos.y <= 106) {
        direction = 2;
        if (game.gameControl.path(x + (ow / 2), y + oh)) {
            step = step + 0.2;
            y += 2;
            lastdir = 2;
        }
    }
    // down
    if (touchPos.x >= 47 && touchPos.x <= 67 && touchPos.y >= 12 && touchPos.y <= 47) {
        direction = 0;
        if (game.gameControl.path(x + (ow / 2), y)) {
            step = step + 0.2;
            y -= 2;
            lastdir = 0;
        }
    }
    // right
    if (touchPos.x >= 70 && touchPos.x <= 104 && touchPos.y >= 50 && touchPos.y <= 69) {
        direction = 3;
        if (game.gameControl.path(x + ow, y + (oh / 2))) {
            step = step + 0.2;
            x += 2;
            lastdir = 3;
        }
    }

    // left
    if (touchPos.x >= 10 && touchPos.x <= 45 && touchPos.y >= 51 && touchPos.y <= 69) {
        direction = 1;
        if (game.gameControl.path(x, y + (oh / 2))) {
            x -= 2;
            step = step + 0.2;
            lastdir = 1;
        }
    }
}
```

Fonte: Elaboração dos autores, 2013.

Na figura anterior pode-se ver a implementação da funcionalidade da movimentação do personagem onde, verifica-se se houve o toque, e em seguida verifica-se se este toque foi na posição referente ao botão do movimento e em seguida é realizado seu funcionamento definição de uma direção, verificação se é um caminho valido, o movimento, o aumento do step para animação do caminhar, e a definição de uma nova variável de direção para validação de outra funcionalidade.

6.2.6 Personagem

O personagem de um jogo é uma das principais peças no desenvolvimento do mesmo. Sua estrutura segue o padrão, aonde se tem uma classe cujo possui N elementos referentes ao personagem em forma de variáveis, que compreendem desde sua vida, status, nível, equipamentos, a lista de itens do inventário, etc. Sua estrutura segue o padrão do fluxo do jogo onde o construtor refere-se ao *create*, o método *run* referente ao render, o qual contem suas definições para movimentação vistas anteriormente, assim como controle do ganho de experiência e nível, etc., e o *dispose* para seus recursos.

Esta classe assim como padrão possui sua respectiva função *draw* para desenhar-se seguindo o mesmo formato já relatado porém com pequena modificação pois possui uma verificação para que conforme a classe escolhida pelo jogador a sua imagem possui uma cor diferenciada. Além disso esta classe ainda possui duas outras funções *draw* referentes a barra de vida e de experiência do jogador, esses métodos se alteram apenas pelo fato de serem definidos por fórmulas cujo desenham-se para no máximo (vida completa) ao tamanho da imagem do jogador, descendo conforme o jogador perde a vida.

Em adição a classe do personagem possui algumas funções extras para funcionamento de outros elementos do jogo, como métodos que possuem a definição de quanto de dano o personagem dará ao desferir seus ataques baseando-se em sua arma e status do jogado, assim como método para redução de dano recebido dos inimigos baseada em seus

itens e status e um método para comparação de itens refentes a utilização do inventário para comparar se o item selecionado pelo jogador é melhor ou pior que o equipado no momento.

6.2.7 Inimigos

Os inimigos assim como o personagem é de grande importância para o jogo, sua estrutura é basicamente a mesma a do personagem com algumas diferenças. Entre as principais diferenças podemos destacar a ausência de itens e de inventário, em vez de variáveis de experiência do inimigo essas funcionam para experiência que ele dará ao jogador ao ser morto, entre outras variáveis que são utilizadas no funcionamento da IA.

De modo geral os inimigos possuem o mesmo fluxo padrão do personagem onde o construtor refere-se ao *create*, o método *run* referente ao render, o qual contem agora as definições para movimentação segundo a IA do inimigo, e o *dispose* para seus recursos, assim como o *draw*.

A inteligência artificial(IA) do inimigo trabalha da seguinte forma, existem dois blocos um referente ao seu padrão de movimento cujo entra fica em execução sempre que o jogador não estiver na área de perseguição, esse padrão de movimento é definido ao se criar o inimigo podendo ser por exemplo *STATIC* para parado, *LineX* ou *LineY* para movimentos no eixo x ou y, etc., conforme a necessidade, já caso o jogador esteja na área de perseguição o bloco de perseguição é executado este por sua vez funciona no formato *cheating*, ou seja, ele comunica-se com o jogo e pega a posição do jogador e basicamente move-se para essa direção se desviando dos obstáculos no caminho, e ao chegar perto o suficiente entra em execução a função de ataque onde a IA começa a atacar na tentativa de acertar o jogador caso este esteja em sua área de ataque.

6.2.8 NPC

O *npc* é um elemento do jogo o qual possui alguma função, neste projeto os *npc's* tem a função apenas de falar algo refente ao contexto da história do jogo, mas sua principal função é o *savegame*, durante o jogo o único modo de se salvar o jogo é entrando na área do *npc*, que em sua estrutura que também segue o padrão *create*, *run*, *draw*, *dispose*, possui a chamada para a função de salvar o jogo que posteriormente será melhor explicada.

6.2.9 Ataques

Os ataques do jogo assim como os outros elementos do dele possui o padrão *create*, *run*, *draw*, *dispose*. Os ataques são divididos em duas estruturas diferentes mas com basicamente mesmo funcionamento, o ataque do jogador e o ataque dos inimigos, basicamente essas estruturas são elementos que são desenhando a partir da posição de um outro elemento jogador ou inimigo, o qual possui em sua estrutura uma variável de direção essa variável é enviada a ataque que por sua vez é desenhado em função dessa direção e continuara indo nesta direção ate que colida com uma parede segundo a mascara ou o elemento inverso, ou seja, ataque do jogador tem colisão com inimigo, e ataque do inimigo colide com jogador, assim chamando a função para dar o dano seja ao jogador ou ao monstro, no caso de ser o ataque do monstro ao jogador há antes da dedução da vida um calculo do dano a ser causado deduzindo-se o dano pelo montante de armadura do jogador, e no caso de ser um ataque do jogador ao monstro será chamado o dano do jogador que é baseado em seus itens e status.

Há ainda entre os ataques dos inimigos os ataques que são utilizados por chefes estes já possuem estruturas diferenciadas desenvolvidas exclusivamente para cada batalha como explosões, danos contínuos como chamas, etc.

6.2.10 Animação

A animação já foi mencionada ao decorrer deste capítulo mas ainda não foi totalmente explicada. Há diversas formas de se trabalhar com animação dos elementos de um jogo, neste jogo foi utilizado *spritesheets* para cada elemento, ou seja, um arquivo o qual pertence por exemplo ao jogador, nele há o quadro a quadro de sua animação, como se trabalha em um cenário 2d e sem animações para ataques há apenas o quadro a quadro do seu movimento como pode ser observado na Figura 47.

Figura 47. Personagem

Fonte: Jogo *Hero of Allacrost*, 2013.

Para adicionar esta animação ao jogo o que se faz é o seguinte, tem-se duas variáveis básicas para o seu funcionamento e pode ser visto na Figura 36, o *step* e a *direction,* aonde o *direction* define qual a linha a ser pega e o *step* a coluna, e como explicado na seção sobre a função *draw* estes são multiplicados pelo *height* e *width*, altura e largura da imagem a ser desenhada, respectivamente. Assim pelo princípio do jogo atuar em um *loop* gera-se o

chamado para a função *run* do personagem repetidamente, assim se tem o desenho continuo de um pedaço da imagem respectivo aos valores das variáveis, o que ocorre para realizar a animação do caminhar é que ao se chamar o *run* do personagem e neste método for verificado a ação do jogador de pôr exemplo andar para a direita, enquanto esta ação estiver sendo executada, ou seja, o jogador estiver pressionando o botão para andar para direita *step* será incrementado, em um valor de 0.2 e a *direction* será definida como 3, assim a progressão dos quadros da animação para direita sempre que este valor aumentar em 1, mas porque não utiliza o valor 1, vem do princípio que esta função é chamada um número x de vezes por segundo e se colocar um valor alto este provocara uma animação muito rápida em comparação a sua movimentação na tela, mas é claro isso dependera do jogo e do melhor ajuste de cada desenvolvedor para seu jogo. Este processo ocorre para cada elemento presente no jogo, seja o jogador, *npc*, ataques, etc.

6.2.11 Recompensa

No jogo quando o jogador consegue zerar o valor referente a vida do inimigo seja ele um monstro comum ou um chefe, este deve randomicamente decidir se deixará algo ou não, e mais se deixar o que deixar também de forma randômica para que assim entre também o elemento surpresa na equação, isto pode ser observado no trecho de código da Figura 48.

Figura 48. *Drop*

```
Random gerador= new Random();
boolean dropar=gerador.nextBoolean();
if(dropar){
    game.drops.add(new Drop(game, "chest.png", game.enemys.get(i).getX(),
        game.enemys.get(i).getY(), 23, 16, game.enemys.get(i).getLvl()));
}
```

Fonte: Elaboração dos autores, 2013.

O que ocorre na Figura 48 é a definição de uma variável *random* que é utilizada para gerar um valor verdadeiro ou falso para variável *dropar* e logo em seguida caso essa variável seja verdadeira, ou seja, que o monstro ira deixar algo ao jogar ele adiciona a uma lista de elemento do tipo *drop* na posição em que o monstro foi morto, e com o nível do monstro.

Mas o que foi feito foi adicionar um elemento ao jogo para que quando o jogador utilize o botão de pegar itens e esteja no local onde tenha um item, um item seja adicionado a seu inventário como pode ser observado na Figura 49.

Figura 49. *PickUp*

```
int lvl = gerador.nextInt(game.drops.get(i).droplevel + 5) + 1;
int power = gerador.nextInt(game.drops.get(i).droplevel + 5) + 1;
TipoClasse tclasse = null;
TipoItem titem = null;
int n = gerador.nextInt(6);
switch (n) {
    case 0:
        tclasse = TipoClasse.Warrior;
        titem = TipoItem.Sword;
        break;
    case 1:
        tclasse = TipoClasse.Mage;
        titem = TipoItem.Wand;
        break;
    case 2:
        tclasse = TipoClasse.Archer;
        titem = TipoItem.Bow;
        break;
    case 3:
        tclasse = TipoClasse.Any;
        titem = TipoItem.Chest;
        break;
    case 4:
        tclasse = TipoClasse.Any;
        titem = TipoItem.Helmet;
        break;
    case 5:
        classe = TipoClasse.Any;
        titem = TipoItem.Boots;
        break;
}
game.drops.remove(game.drops.get(i));
game.player.addItemInventario(new Item(power, lvl,tclasse, titem));
```

Fonte: Elaboração dos autores, 2013.

Neste trecho de código que é chamado quando o jogador aperta o botão de pegar item do chão perto de um elemento do tipo *drop*, é utilizado mais três variáveis randômicas que definem o nível do item, o poder que é o dano ou a defesa dependendo do item, e depois é utilizado uma variável n para definir qual o tipo do item, em seguida é removido o *drop* da lista de itens no chão e adicionado o item gerado com os valores randômicos ao inventário do jogador.

6.2.12 Inventário

Neste projeto por se tratar de um jogo RPG cujo possui itens e status que devem ser cuidadosamente geridos pelo jogador foi necessário a implementação do inventário, muito comum nesse gênero de jogo.

Apesar de ao primeiro momento se pensar que o inventário possa ser uma estrutura completamente diferente de ser desenvolvida, isto esta errado, pois ele continua a apresentar o mesmo padrão *create*, *run*, *draw*, *dispose*, porém não se está trabalhando algo que se movimenta na tela, ataque, etc., deve-se sim controlar as ações que o jogador pode realizar. Como pode ser visto na Figura 50, o inventário aberto, com os itens do jogador, status, opções, e informações.

Figura 50. Inventário

Fonte: Elaboração dos autores, 2013.

Como observado na Figura 50 há uma série de ações a serem consideradas ao implementar este inventário. A primeira seria seu funcionamento, ou seja, o controle do jogador, deve-se cuidadosamente mapear as posições de todos os elementos o qual o jogador terá interação, botões, os itens no inventário, os itens equipados, o que ocorre é a adição de uma função para cada posição, ou seja, por exemplo ao tocar na tela na posição x, y referente a um item no inventário este é desenhando um quadrado amarelo na posição informando que foi selecionado, apresentando as informações do item que são pega na lista de itens do inventário do personagem, e assim habilitando a possibilidade de equipar este item, que será validado na questão da classe para qual é e qual classe o jogador é.

A segunda seria o carregamento de todos os itens do inventário do jogador e os itens que estão equipados, com as posições já mapeadas apenas têm que ir a lista de itens do inventário percorrer ela e ir adicionando nas posições as *sprites* dos itens, selecionando o tipo, e o *tier* que é definido pelo nível do item assim selecionando entre os diferentes itens do mesmo tipo como elmo de madeira, de ferro, de ouro, etc., o mesmo ocorre com os itens equipados.

O terceiro seria os status, em que é necessário ir ao personagem pegar o status e os mostrar na tela, faz-se o mesmo com os pontos que se possui para colocar, e é valido sempre que o jogador pressionar o botão para adicionar um ponto se há o ponto para por, fazendo a adição no status selecionado dependo a posição pressionada e a dedução nos pontos de status disponíveis.

O último fator a ser tratado é a ação dos botões principais, equipar e desequipar, e deletar, o que ocorre é que se deve apenas trabalhar no personagem pelo fato de se alterar os itens, trocando da lista para os equipados e vice-versa, e tendo em mente que a função para carregar estes itens já esta pronta, ao alterar os itens de posição ou deletar eles da lista, na próxima passagem pelo método *run* ele automaticamente faz a troca na tela.

6.2.13 Progressão

Neste projeto tem-se três diferentes formas de progressão, a primeira é o nível do personagem este é adquirido ao matar monstros e ganhar pontos de experiência que ao se chegar a um certo número se passa de nível ganhando pontos de status que podem ser utilizados para melhorar seu personagem e deixar como o jogador preferir, por exemplo mais dano, mais vida, etc.

O segundo se trata dos itens do personagem, é utilizado um formato de jogo que incentiva o jogador a matar um grande número de monstros para que consiga itens melhores para que tenha a capacidade de enfrentar os próximos desafios, e os chefes.

A terceira forma de progressão implementada é o nível de dificuldade dos monstros, a medida em que se avança no jogo os monstros ficam mais fortes têm maior nível, mais vida, mais dano, e melhores recompensas ao serem derrotados, e a medida que o jogador avança ele enfrenta os chefes que elevam ainda mais a dificuldade de se avançar e completar o jogo.

6.2.14 Chefes

Os chefes são de grande importância para a história e progressão do jogo desenvolvido, pois eles fazem o papel de ponte entre um desafio e outro elevando a dificuldade para o próximo estágio que vem a seguir.

Os chefes devem promover um desafio ao jogador porém devem ser capazes de serem derrotados o que se acaba criando grande dificuldade no seu balanceamento dentro do jogo desenvolvido. Outro fator que dificulta sua implementação é que eles continuam sendo considerados monstros do jogo porém devem ser desenvolvidos individualmente cada qual com seu próprio estilo, ataques, e ações.

Para este projeto a implementação dos chefes foi feita seguindo o plano de fases da batalha, onde a luta ocorre em três fases, que se alternam ao passar do tempo, ate que o jogador ou o chefe seja derrotado.

Não há muito o que se falar pois muitas são as formas de se conseguir o resultado que o desenvolvedor quer, e este é um trabalho de tentativa e erro, implementa-se testa, modifica-se, testa não há maneira certa ou errada quando se fala desse tipo de situação.

Por estarem dentro da classe inimigos continuam sobre o mesmo padrão de todos os elementos, *create*, *run*, *draw*, e *dispose*, o que importa é a criatividade, para desenvolver os elementos da batalha.

6.2.15 Save/Load Game

Por se tratar de um jogo do tipo RPG, para este projeto foi necessário criar alguma forma de o jogador salvar seu progresso para que pudesse jogar novamente de onde parou sem perder nada. A forma escolhida para o jogo foi o desenvolvimento de uma classe que ficasse responsável pelo gerenciamento de um arquivo do tipo "*.properties"* comumente utilizado em projetos para salvar informações de banco de dados como localização, *driver*, etc. Em outros projetos, mas neste caso foi utilizado este mesmo tipo de arquivo que é facilmente manipulável em java para armazenar todas as informações necessárias para realizarmos o *save* e *load* game.

Basicamente a ideia é bem simples, o que foi feito foi a implementação de dois métodos um para salvar e o outro para restaurar as informações. O que ocorre é que ao se chamar o método de salvar esta classe verifica a existência de um arquivo "*save.properties*" caso não exista ele o cria, caso exista ele apenas ira atualizar todas as informações contidas nele, já o segundo método ao ser chamado verifica a existência do arquivo caso não exista ele iniciara um novo jogo, caso exista ele começara o jogo aonde o jogador salvou pela última vez com todos os seus itens, níveis, status, e chefes mortos, sendo que o método de *load* só pode ser chamado na tela de incio do jogo, e o *save* game será chamado sempre que o jogador

entrar na área do *npc* responsável pelo salvamento ou quando ele morrer há a opção de salvar o jogo antes de sair.

As informações que são salvas são todas aquelas que tem impacto sobre a progressão do jogador, elas são a classe que o jogador escolheu, seu nível, cada um de seus status, seus itens equipados, os pontos ainda não gastos nos status, sua vida máxima, sua experiência atual, a experiência para o próximo nível, os chefes já derrotados, e cada item no inventário.

6.2.16 Game Over

O *game over* ou fim de jogo para este projeto se refere a morte do jogador, ou vencer o último chefe, o que acontece é no momento em que o jogador chega a vida zero o jogo lhe oferece três opções sair, reviver ou salvar e sair, basicamente a primeira fecha o jogo e tudo que ele vez entre o último *save game* é perdido, a segundo faz ele reviver no último local onde foi salvo, e a terceira opção salva o jogo e sai.

Já ao vencer o jogo derrotando o último chefe o jogo informa ao jogador seu feito e que este não é o fim e sim apenas o recomeço, o jogo então oferece ao jogador a possibilidade de continuar seu jogo começando no inicio do jogo novamente, mas dessa vez ele começa já com seus itens, níveis, etc., porém tudo está mais forte, mais desafiador, tem mais vida, mais dano, um novo jogo.

6.3 PROCESSO DO DESENVOLVIMENTO DO JOGO

Primeiramente iniciou-se o desenvolvimento de uma versão inicial do jogo para se ter uma base para produção da versão final do jogo, nesta primeira versão foi desenvolvido a

base do jogo, o personagem e sua movimentação pela tela foi o primeiro elemento a ser criado, com a utilização de um mapa e mascara de teste, em seguida desenvolvemos o ataque do personagem, em seguida foi desenvolvido um inimigo. Apos isso foi criado a colisão entre o ataque e o inimigo ate então sem qualquer funcionalidade, esta foi a primeira versão do jogo.

Apos o desenvolvimento da primeira versão do jogo, ocorre o início do desenvolvimento da versão final do jogo. Foi realizada uma reformulação nas estruturas criadas na primeira versão do jogo para a implementação dos diversos outros elementos presentes nas seções anteriores. Partindo da implementação das ações dos inimigos, movimentação, perseguição e ataque.

A partir deste momento possuindo os dois elementos fundamentais do jogo se começou o desenvolvimento de suas iterações, ou seja, o que acontecia quando o inimigo acertado e morto, primeiro foi implementado o sistema de vida tanto do personagem quanto do inimigo, em seguida foi criado a experiência do personagem e seu ganho através da morte do inimigo. Logo após foi implementado o elemento *drop* que ao se abater um inimigo tinha a chance de gerar uma caixa, mas ate então sem função alguma.

A implementação de um sistema de itens foi o próximo estagio, foi criado a estrutura dos itens e como funcionariam, e sua modificação no personagem. Neste ponto foi definido o próximo passo a criação de um inventário. Então foi criação o *background* do inventário e sua funcionalidade, ate este ponto, os itens eram então inseridos via código para aparecerem na tela e atuarem no personagem.

Em seguida foi então completada a implementação do sistema de *drop* e de *pickup* de itens randomicamente gerando seu nível e poder.

Apos isto a implementação do status do personagem e seu funcionamento no inventário foram os passos seguinte no desenvolvimento. Em conjunto com esta fase era desenvolvido então a funcionalidade de poção para o jogador recuperar vida durante o jogo e recarregar a poção conforme ia se matando os monstros.

As últimas etapas do desenvolvimento foram focadas no ampliamento do número de monstros que ate então era de dois e mo desenvolvimento dos chefes, que acabaram sendo os elementos mais complicados de se implementar no jogo, pois cada chefe possui um

conjunto de ações unicas que deviam ser implementadas uma a uma. Assim como o acesso à arena de batalha contra os chefes.

O desenvolvimento do mapa para o jogo foi então focado e conforme se desenvolvia se notava que a melhor opção para o jogo não seria um mapa de mundo aberto e sim diversos mapas interligados por portais.

Logo a implementação de portais para o funcionamento do jogo era necessária, para que assim a última fase do desenvolvimento do jogo fosse feita.

Apos o desenvolvimento desses portais a fase de planejamento e disposição dos inimigos pelos mapas foi feita chegando ao fim do desenvolvimento.

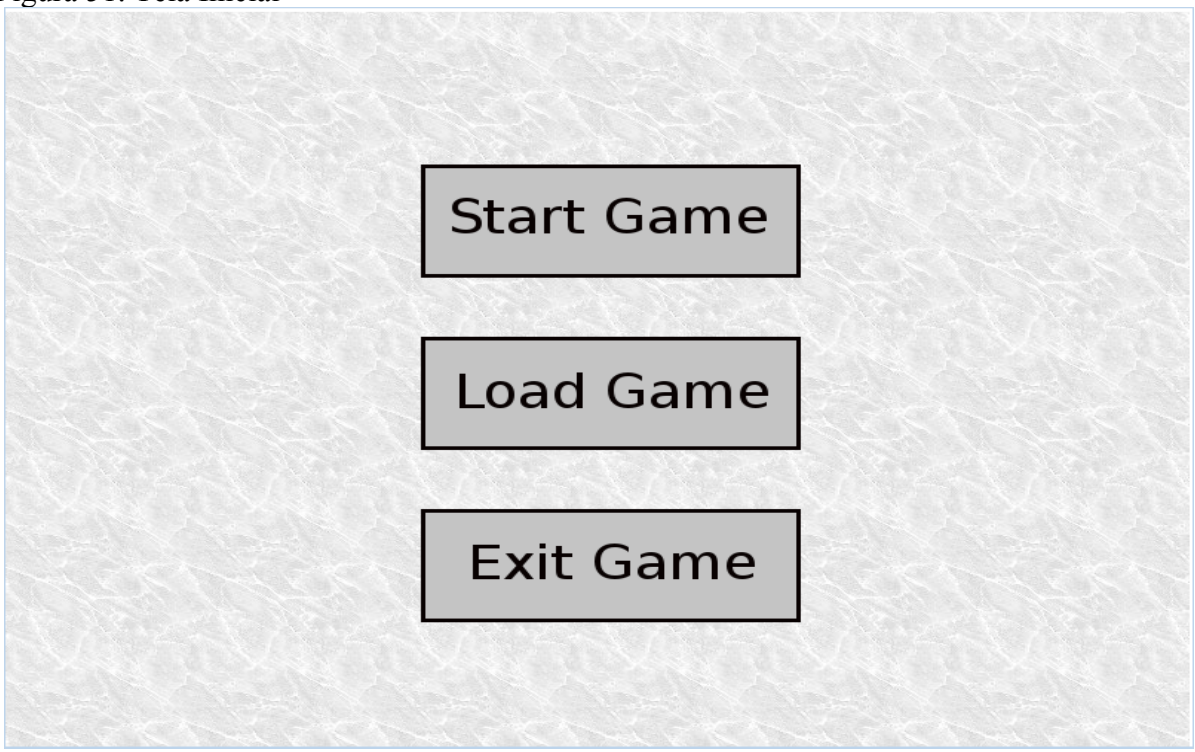
6.4 VERSÃO FINAL - THE THREE SEALED ELEMENTS

Nesta seção é apresentado o resultado final do projeto, apresentando o jogo de forma geral em seus diversos momentos desde o início do jogo ao seu fim, apresentando também as suas duas PC e Android possibilitada pela biblioteca LibGdx.

6.4.1 Tela Inicial

Assim que o jogo é iniciado a tela que pode ser observada na Figura 51, é apresentada ao jogador.

Figura 51. Tela Inicial



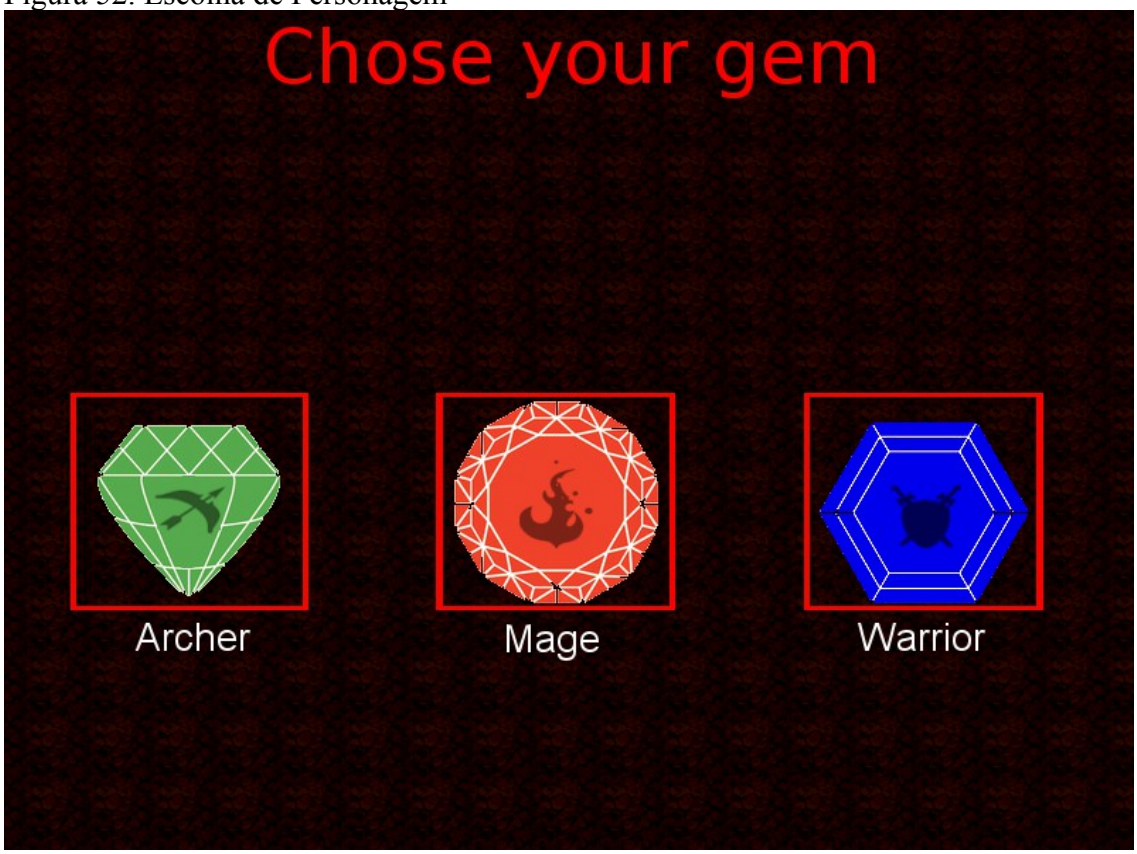
Fonte: Elaboração dos autores, 2013.

Nesta tela o jogador pode escolher entre começar um novo jogo aonde ele será redirecionando para uma nova tela de escolha de personagem ou recomeçar em seu último *savegame*, ou optar por sair.

6.4.2 Escolha Personagem

Assim que o jogador selecionar a opção de *start game* na tela inicial tela representada na Figura 52 é apresentada ao jogador.

Figura 52. Escolha de Personagem



Fonte: Elaboração dos autores, 2013.

Nesta tela o jogador é convocado a escolher entra as três possibilidades de personagens do jogo representadas por suas gemas conforme a história do jogo, arqueiro, mago, ou guerreiro. Logo após a seleção o jogo é iniciado.

6.4.3 Fases / Interface

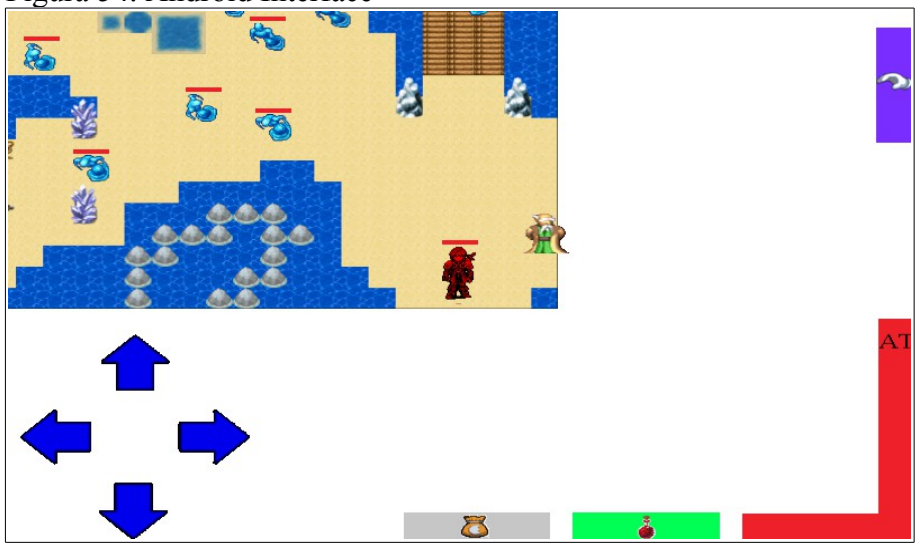
Após a escolha do personagem o jogo é iniciado em sua primeira fase a fase da Água como pode ser visto na Figura 53.

Figura 53. Início do Jogo Fase Aguá

Fonte: Elaboração dos autores, 2013.

Na Figura 53 é apresentado a fase da água e além disso a interface do jogo na versão de PC, já na Figura 54 na versão de Android.

Figura 54. Android Interface



Fonte: Elaboração dos autores, 2013.

As diferenças entre as duas versões são mínimas, o jogo é o mesmo em ambas o que muda realmente é como é jogado, na versão de Android o jogo é inteiramente controlado através da tela *touch*.

Ao fim da primeira área temos o portal que nos leva ao chefe da fase conforme a Figura 55.

Figura 55. Portal Chefe Água

Fonte: Elaboração dos autores, 2013.

Ao entrar no portal o jogador deve enfrentar o chefe da água que ao ser derrotado, abre o portal para a próxima área, que pode ser vista na Figura 56.

Figura 56. Fase Fogo

Fonte: Elaboração dos autores, 2013.

Assim como na fase anterior o jogador deve passar pela área e chegar ao portal que o levará para a batalha com o chefe, e vencendo-a ira a próxima fase a área da terra conforme a Figura 57.

Figura 57. Fase Terra

Fonte: Elaboração dos autores, 2013.

A penúltima área a da terra, assim que o jogador conseguir enfrentar os monstros no caminho e alcançar o portal ele é levado a penúltima batalha com o chefe da terra, que ao ser derrota abre caminho a última fase, que pode ser observada na Figura 58.

Figura 58. Fase Final

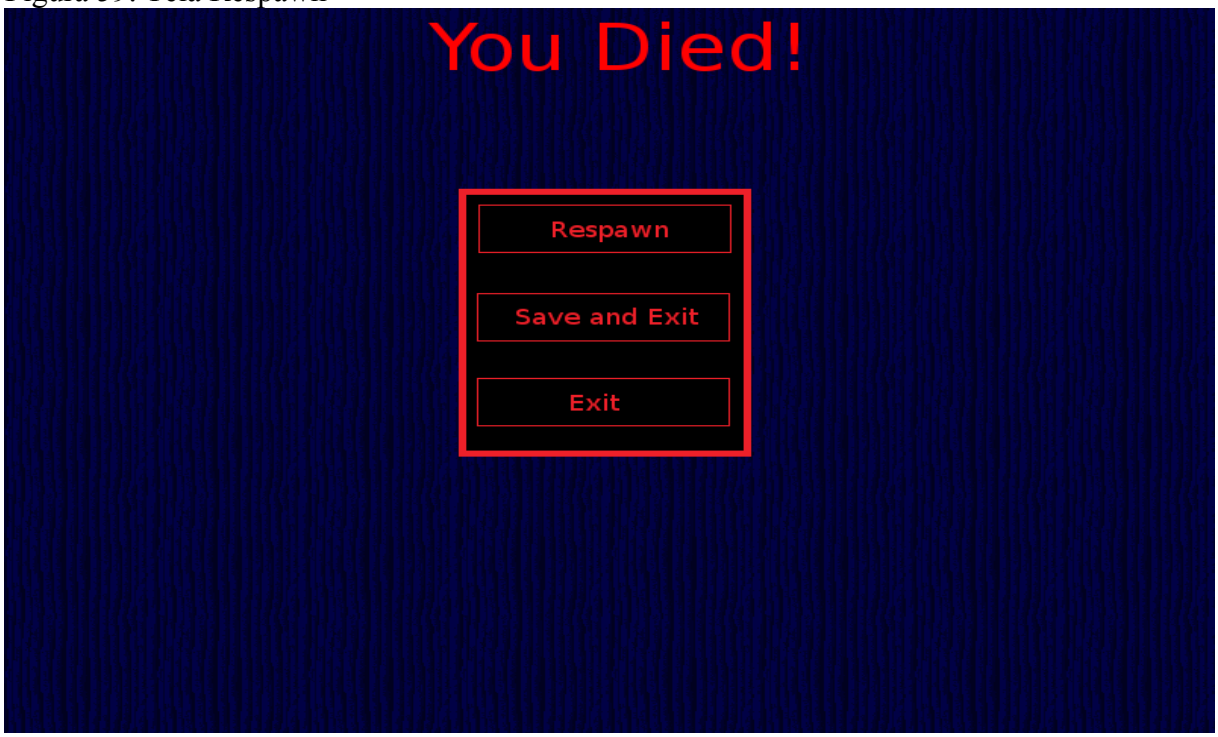
Fonte: Elaboração dos autores, 2013.

A fase final é a última área do jogo que ao conseguir chegar ao topo do templo, após passar por todos os inimigos no caminho, o jogador enfrentara a luta derradeira o chefe final, e fica surpreso com sua face.

6.4.4 Respawn

Quando o jogador deixar que a vida do seu personagem chegue a zero a tela referente a Figura 59, é mostrada ao jogador.

Figura 59. Tela Respawn



Fonte: Elaboração dos autores, 2013.

Assim que a vida do jogador chegar a zero ele será encaminhado a tela de *respawn* que permite ao jogador três opções renascer, salvar e sair ou apenas sair.

6.4.5 Chefes

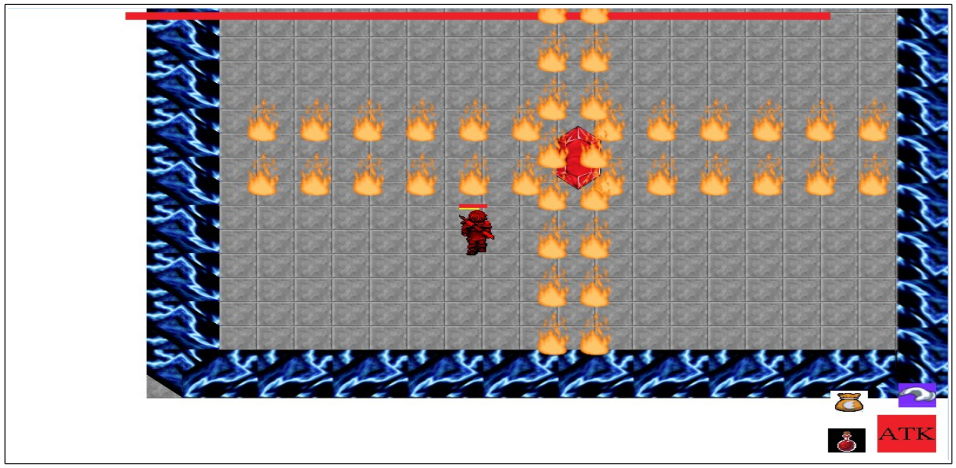
Os chefes partes de grande importância dentro do jogo, todas as áreas são finalizadas com seus respectivos chefes, estas baseadas em sua área água, fogo, terra, e o chefe final que contempla os três elementos unidos.

Figura 60. Chefe Água

Fonte: Elaboração dos autores, 2013.

Na Figura 60 é apresentado o chefe relativo ao elemento água o Cristal da Água conforme a história do jogo, o primeiro e o mais fraco chefe, este possui as mecânicas mais simples por ser o primeiro chefe.

Figura 61. Chefe Fogo



Fonte: Elaboração dos autores, 2013.

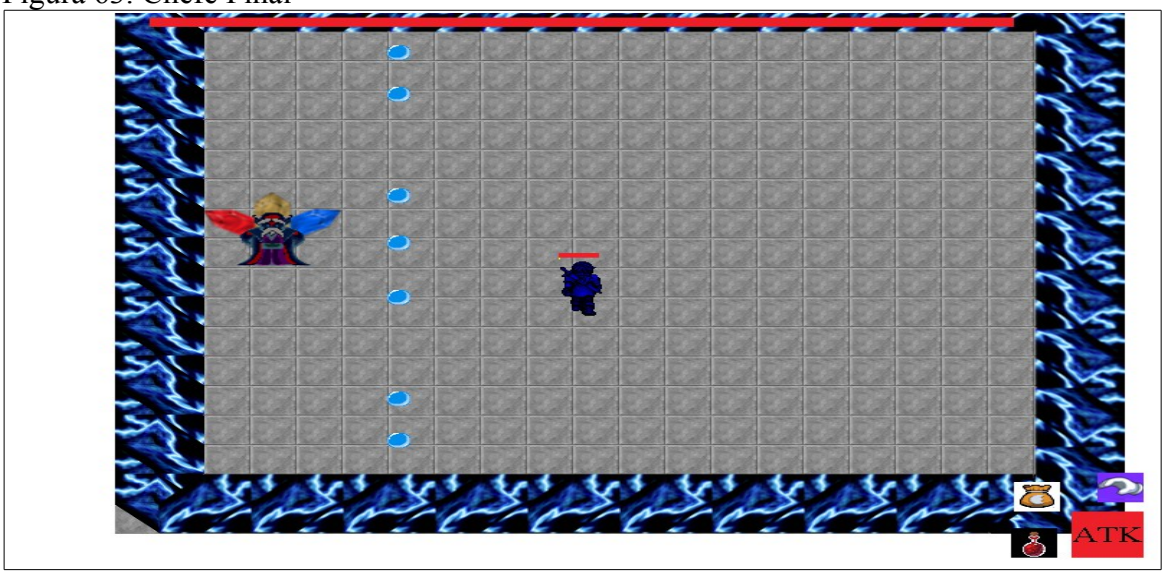
O chefe do elemento Fogo ou o Cristal do Fogo, o segundo chefe guardião da segunda área do jogo, este chefe possui mecânicas um pouco mais difíceis.

Figura 62. Chefe Terra

Fonte: Elaboração dos autores, 2013.

Com mecânicas mais complexas e que requerem um pouco mais de atenção do jogador o chefe da área da terra o Cristal da Terra mostrado na Figura 62 é o terceiro chefe a ser enfrentado pelo jogador.

Figura 63. Chefe Final



Fonte: Elaboração dos autores, 2013.

O chefe final o último desafio do jogo, este chefe é seu Mestre como observado na Figura 63, corrompido pelo mal e utilizador dos três cristais elementais, nesta luta além de habilidade com os golpes dos três elementos o jogador necessitara de sorte no golpe escolhido pelo chefe para usar.

6.4.6 Fim de Jogo

Ao vencer todos os desafios ao longo do jogo o jogador chega ao fim do jogo ao derrotar o último chefe seu Mestre e após isso é lhe apresentado a tela final que pode ser vista na Figura 64.

Figura 64. Fim de Jogo

Fonte: Elaboração dos autores, 2013.

Ao finalizar o jogo e chegar a tela final o jogador ainda tem a possibilidade de continuar jogando com seu personagem reiniciando o game do início em um maior dificuldade, onde os monstros são mais fortes e resistentes.

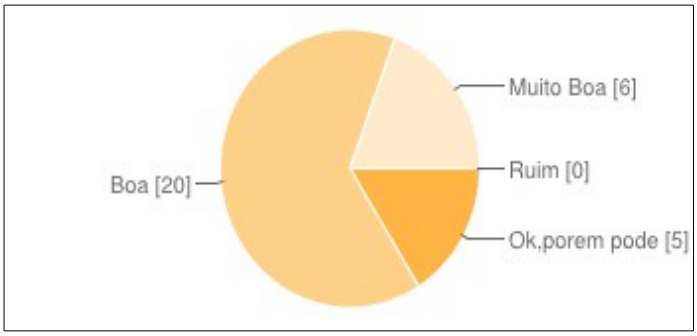
6.5 AVALIAÇÃO

Para a avaliação deste trabalho, o resultado do jogo versão considerada como beta foi colocada em um período de testes e avaliação para o público em geral, o objetivo deste questionário foi apenas definir se o aplicativo desenvolvido é considerado um jogo válido e seu funcionamento fosse analisado e possíveis problemas e/ou *bugs* encontrados.

O jogo em sua versão beta, foi colocado a disposição do público através um blog para que pudessem baixar, testar e avaliar o game. O blog pode ser acessado através do endereço http://tcc-ttse.blogspot.com.br, nele é encontrado uma breve introdução sobre o projeto, o resumo da história do jogo e os links, de download e do questionário para avaliação. Após certo período de tempo, cerca de seis dias foram realizados trinta e uma, avaliações do jogo.

A primeira questão foi perguntado de forma geral o que os usuários haviam achado da ideia do jogo, com as opções ruim, ok porém poderia ser melhor abordado, boa ou muito boa, que gerou o resultado da Figura 65.

Figura 65. Resultado Pergunta 1

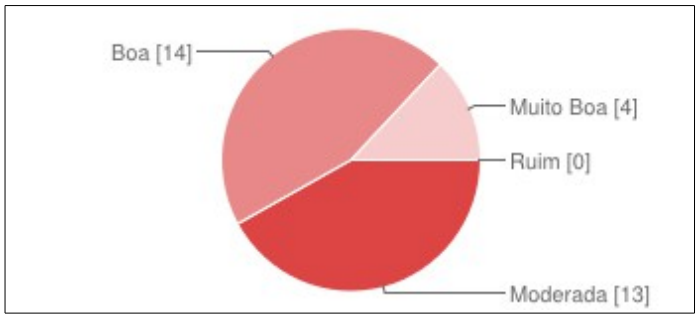


Fonte: Elaboração dos autores, 2013.

Nesta primeira pergunta os resultados foram 20 ou 65% avaliaram com a opção boa, 6 ou 19% muito boa, 5 ou 16% que poderia ter sido melhor abordada, e nenhum resultado para opção ruim. Assim resultando em uma avaliação positiva quanto a ideia do jogo desenvolvido.

A segunda pergunta apresentada aos jogadores foi em relação a jogabilidade do jogo, com as opções de ruim, moderada, boa, ou muito boa. Os resultados dessa pergunta podem ser observados na Figura 66.

Figura 66. Resultado Pergunta 2

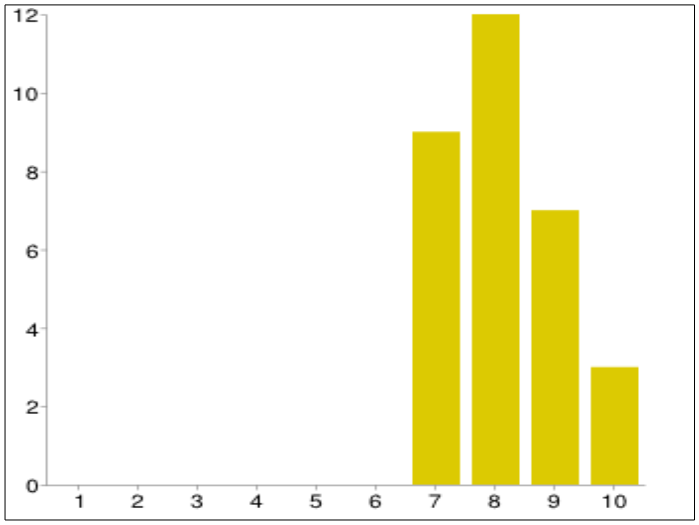


Fonte: Elaboração dos autores, 2013.

Conforme visto na Figura 66, os resultados apresentam 14 ou 45% das respostas para opção boa, 13 ou 42% para opção moderada, 4 ou 13% para muito boa, e nenhum resultado para opção ruim. Novamente os resultados apresentados foram positivos.

A terceira questão foi relacionada ao conjunto mapa, monstros e dificuldade, em forma de nota entre 1 a 10, este resultado pode ser visto na Figura 67.

Figura 67. Resultado Pergunta 3

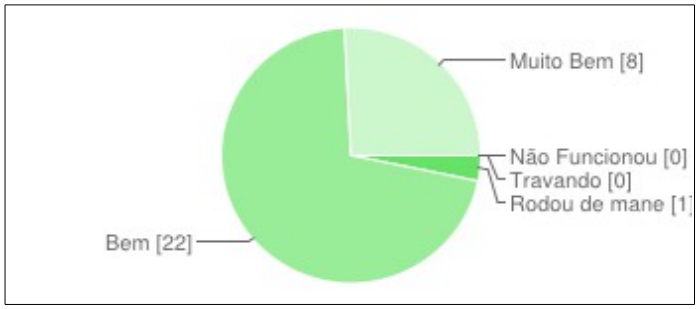


Fonte: Elaboração dos autores, 2013.

A Figura 67 apresenta 12 votos ou 39% dos resultados para nota 8, 9 ou 29% dos resultados para nota 7, 23% ou 7 votos para nota 9 e 3 ou 10% para nota 10, nenhum das outras opções foi escolhida, o que acabou em uma avaliação positiva quanto ao balanceamento e designer dos mapas e monstros.

A quarta pergunta apresentada foi em relação ao funcionamento do jogo, com as opções não funcionou, travando, rodou de maneira aceitável, bem e muito bem, o resultado é apresentado na Figura 68.

Figura 68. Resultado Pergunta 4

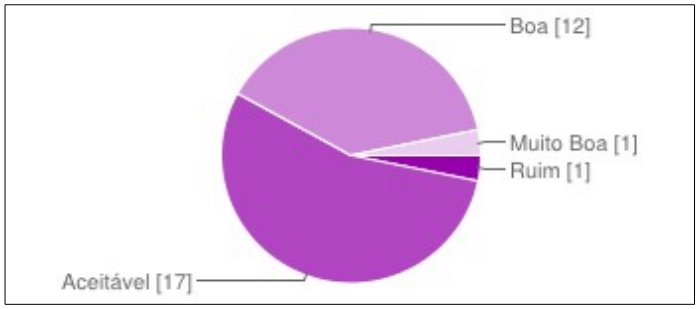


Fonte: Elaboração dos autores, 2013.

Conforme a Figura 68, a questão quatro apresentou as opções, rodou de maneira aceitável com 1 voto ou 3%, bem com 22 ou 71% dos votos e muito bem com 8 ou 26%, o que confirmou de maneira positiva o funcionamento do jogo.

Já a pergunta número cinco foi referente a interface do jogo, com as opções ruim, aceitável, boa e muito boa, seu resultado pode ser visto na Figura 69.

Figura 69. Resultado Pergunta 5



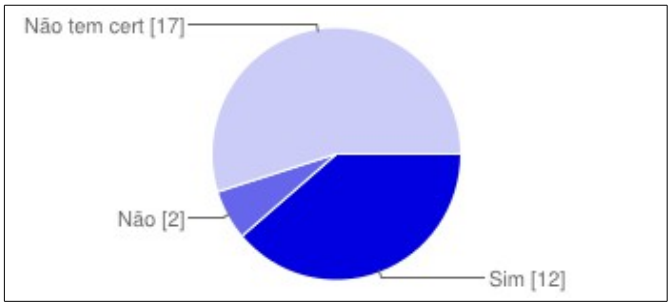
Fonte: Elaboração dos autores, 2013.

Como observado na Figura 69, todos as opções tiveram resultados, ruim com 1 ou 3% dos resultados, aceitável com 17 ou 55%, boa com 12 ou 39% e muito boa com 1 ou3% dos resultados, esse resultado já nos mostra que a interface foi o ponto em que mais houve

variação, o que pode ocorrer pelo gosto de cada usuário e pelo tamanho dos aparelhos que pode ter atrapalhado a utilização da interface.

A pergunta número seis foi referente a se o testador jogaria um jogo como aquele, com as opções sim, não e não tem certeza, o resultado é observado na Figura 70.

Figura 70. Resultado Pergunta 6

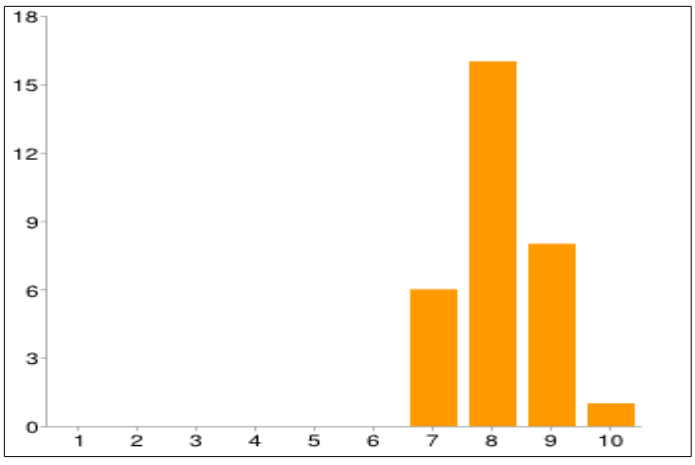


Fonte: Elaboração dos autores, 2013.

Os resultados da pergunta número sete apresentam sim com 12 ou 39% dos resultados, não com 2 ou 6%, e não tem certeza com 17 ou 55%, está pergunta é muito afetado pelo gosto pessoal de quem testou o jogo, porém ainda apresenta-se 39% como favorável ao jogo, o que é um resultado positivo por se tratar de um RPG.

Já a sétima pergunta foi em para dar uma nota geral para o jogo entre 1 a 10, seu resultado pode ser visto na Figura 71.

Figura 71. Resultado Pergunta 7



Fonte: Elaboração dos autores, 2013.

Como visto na Figura 71 os resultados foram nota 7 em 6 ou 19% dos casos, nota 8 em 16 ou 52% dos casos, nota 9 em 8 ou 26% e nota 10 em 1 ou 3% dos casos, o que foi um

resultado muito bom, pois não foram dadas notas abaixo de 7 o que mostrou que o jogo foi avaliado de forma positiva perante os testadores, o que mostrou que o jogo tem potencial.

As duas últimas perguntas feitas foram para os jogadores darem sua opinião quanto ao que poderia ser melhorado e a sua opinião quanto ao jogo. A penúltima eles apresentaram sugestões sobre a história do jogo que poderia ter sido mais abordada, ao número de classes e novas funções as classes como habilidades, alguns *bugs* pequenos quanto à detecção do terreno, e sobre a fala do *npc* estar muito pequena nos dispositivos móveis e uma possível versão em português. Na última pergunta todas as repostas foram positivas quanto ao jogo, em relação ao lutas, jogabilidade, estilo, relatando que foi divertido, e em alguns casos ressaltando o que poderia ser melhorado relativo a questão anterior.

De forma geral, o jogo foi avaliado positivamente em praticamente todas as perguntas de múltipla escolha, que tratavam principalmente da qualidade do jogo, o que fez com que se chega-se a conclusão que o jogo ficou bom e que teve uma boa receptividade por parte do público. Além disso com as perguntas descritivas foram identificados alguns *bugs* e possíveis melhorias que podem ser implementadas em futuras versões do jogo, já que o mesmo está em sua versão beta, onde o objetivo é identificar o maior número possível de problemas com o jogo. Portanto, acredita-se que a avaliação serviu com seu propósito, de avaliar o jogo, verificar seu funcionamento e descobrir seus problemas. Assim validando como objeto de estudo desta monografia.

7 CONSIDERAÇÕES FINAIS

Este trabalho circundou sob a compreensão e contextualização do processo para o desenvolvimento de um jogo digital, mais especificamente para a plataforma dos dispositivos móveis, Android.

Parte dos objetivos circundaram sobre a perspectiva de apresentar este processo de desenvolvimento de um jogo digital, desde a criação do *Game Design Document* (GDD), da modelagem do sistema e de sua implementação.

Dessa forma, para alcançar os objetivos, foram necessárias algumas etapas, entre elas, a pesquisa sobre os conhecimentos da estrutura, do processo e das etapas que compreendem o desenvolvimento de jogos em geral com foco em dispositivos móveis.

Para tal, foi efetuada uma pesquisa sobre jogos digitais e sobre a construção de um GDD como forma de embasamento para o projeto em questão. Em tempo, também foram comentadas sobre as tecnologias utilizadas para a implementação.

Para o desenvolvimento, inicialmente foi criado o GDD do jogo, demonstrando seu *design*. Em seguida, foi apresentado um modelo da implementação, tal como um software, através dos diagramas em *Unified Modeling Language* (UML), elencando e explicando cada elemento dentro da estrutura do jogo desenvolvido, assim como o resultado final e a avaliação do mesmo.

Em tempo, com o GDD e o modelo em UML, foi implementada uma primeira versão do protótipo do jogo em linguagem de programação Java para Android, *Android Development Tool* (ADT), e utilizando a biblioteca LibGDX.

O jogo, então, passou por uma fase de testes e avaliação abertas em que o jogo desenvolvido foi colocado a disposição do público através de um blog, em duas versões, Android e PC, e por meio de um questionário para obtenção da validade do jogo e de seu funcionamento. Com os resultados dos questionários, foi constatada uma boa receptividade do jogo para ambas as plataformas.

Por fim, a proposta do trabalho supracitada, de apresentar e experimentar um processo de desenvolvimento de um jogo digital, incluindo a criação do GDD, a modelagem em UML, a implementação, e uma análise de aceitação, foi cumprida. Assim pode se concluir

a importância de todo o processo para a criação de um jogo, pois torna-se algo fluido, organizado e documentado para seu desenvolvimento, com cada etapa complementando a anterior ate sua finalização. Este trabalho proporcionou o conhecimento e as experiências necessárias sobre todo o processo de desenvolvimento de um jogo, o que permiti então que novos profissionais possam começar a ingressar nesta área que está em constante expansão.

7.1 SUGESTÕES PARA TRABALHOS FUTUROS

Durante o processo desse trabalho, alguns pontos de melhorias foram percebidos, mas que estavam fora do contexto e/ou do tempo hábil para o desenvolvimento do mesmo. Dessa forma, são apresentadas aqui sugestões para trabalhos futuros:

- Pesquisa de técnicas avançadas e diferenciadas para cada gênero de jogos digitais e/ou ferramentas específicas para cada;
- Estudo de motores de jogos (*game engine*) para elaboração de jogos mais complexos;
- Estudo sobre a inserção de jogos no mercado *indie* e mundial de jogos digitais.
- Pesquisa das técnicas de desenvolvimento de jogos multijogadores on-line.
- Estudo sobre usabilidade e acessibilidade em jogos e as técnicas necessárias para sua implantação.
- Estudos sobre o paradigma do desenvolvimento multiplataforma.

REFERÊNCIAS

ANATEL - Agência Nacional de Telecomunicações. **Quantidade de Acessos/Plano de Serviço/Unidade da Federação – Janeiro/2013**. 2013. Disponível em: <http://sistemas.anatel.gov.br/SMP/Administracao/Consulta/AcessosPrePosUF/telaConsulta.asp>. Acesso em: 30 set. 2013.

ASSOCIAÇÃO BRASILEIRA DOS DESENVOLVEDORES DE JOGOS ELETRÔNICOS (Org.). **A indústria brasileira de jogos eletrônicos Um mapeamento do crescimento do setor nos últimos 4 anos.** 2008. Disponível em: <http://www.abragames.org/wp-content/uploads/2013/04/Abragames-Pesquisa_2008.pdf>. Acesso em: 15 set. 2013.

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML**: Guia do Usuário. Rio de Janeiro: Campus, 2000. 472 p.

DEITEL, H. M.; DEITEL, P. J. **Java: como programar**. 4. ed. São Paulo: Bookman, 2003. 1386 p.

DETTENBORN, Renê. **Tutorial LibGDX:** Configurar ambiente de desenvolvimento de jogos Android. 2014. Disponível em: <http://renedet.blogspot.com.br/2014/01/tutorial-libgdx-configurar-ambiente-de.html>. Acesso em: 10 maio 2014.

FAHS, Travis. **IGN Presents the History of SEGA**. 2009. Disponível em : <http://www.ign.com/articles/2009/04/21/ign-presents-the-history-of-sega>. Acesso em: 18 ago. 2013.

HORSTMANN, Cay. **Conceitos de Computação com O Essencial de Java.** 3. ed. São Paulo: Bookman, 2003. 783 p.

LECHETA, Ricardo R.. **Google Android:** Aprenda a criar aplicações para dispositivos com o android sd. 3. ed. São Paulo: Novatec Editora, 2013. 819 p.

LEMAY, Laura; CADENHEAD, Rogers. **APRENDA EM 21 DIAS JAVA 2.** 4. ed. Rio de Janeiro: Campus, 2005. 529 p.

MENDES, Antonio. **Arquitetura de Software**: desenvolvimento orientado para arquitetura. Rio de Janeiro: Campus, 2002. 212 p.

NETO, Oziel Moreira. **Entendendo e dominando o Java para internet.** São Paulo: Digerati Books, 2006. 312 p.

NOVAK, Jeannie. **Desenvolvimento de games**. Tradução Pedro Cesar de Conti. São Paulo: Cengage Learning, 2010.

NOVAK, Jeannie. **The Official GameSalad Guide to Game Development**. New York: Cengage Learning, 2014.

PAULA FILHO, Wilson de Pádua. **Engenharia de Software: fundamentos, métodos e padrões.** Rio de Janeiro: Ltc, 2000.

PEREIRA, Lúcio Camilo Oliva; SILVA, Michel Lourenço da. **Android para Desenvolvedores**. Rio de Janeiro: Brasport, 2009. 223 p.

QUINN, Robert E. et al. **Competências Gerenciais**. 5. ed. Rio de Janeiro: Elsevier Editora, 2011.

RABIN, Steve. **Introdução ao desenvolvimento de games: vol.4: a indústria de jogos**. São Paulo: Cengage Learning, 2012.

RAMOS, Ricardo Argenton. **Treinamento Prático em UML.** São Paulo: Universo Dos Livros, 2006. 144 p.

SAMPAIO, Cleuton; ROGDRIGUES, Francisco. **Mobile Game Jam**. Rio de Janeiro: Brasport, 2012. 284 p.

SAMPAIO, Henrique**. Jogos casuais tomam conta da indústria**. Disponível em: <http://jogos.uol.com.br/reportagens/ultnot/2008/06/24/ult2240u131.jhtm>. Acesso em: 31 Agosto. 2013

SCHUYTEMA, Paul. **Design de Games: uma abordagem prática**. Tradução Cláudia Mello Belhassof. São Paulo: Cengage Learning, 2008.

SHELLY, Gary; GUNTER, Glenda; GUNTER, Randolph. **Teachers Discovering Computers: Integrating Technology and Digital Media in Classroon**. 6. ed. Boston: Cengage Learning, 2010.

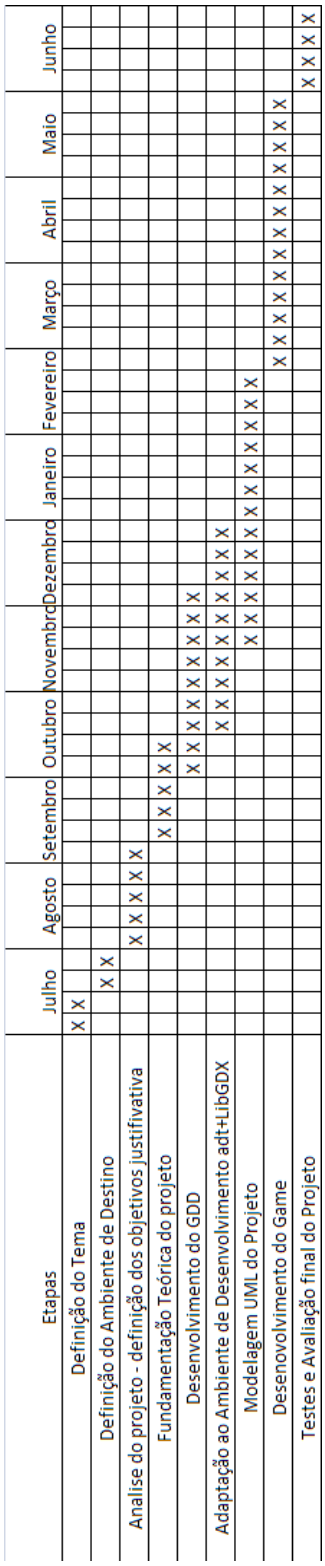
SILVA, Edna Lúcia da; MENEZES, Estera Muszkat. **Metodologia da pesquisa e elaboração de dissertação**. 4 ed. Florianópolis: UFSC, 2005.

SOLLITTO, André. **A maior diversão da terra**. Disponível em: <http://revistaepoca.globo.com/ideias/noticia/2012/02/maior-diversao-da-terra.html> Acesso em: 18 ago. 2013

SOMMERVILLE, Ian. **Engenharia de Software.** 6. ed. São Paulo: Pearson Education do Brasil, 2004.

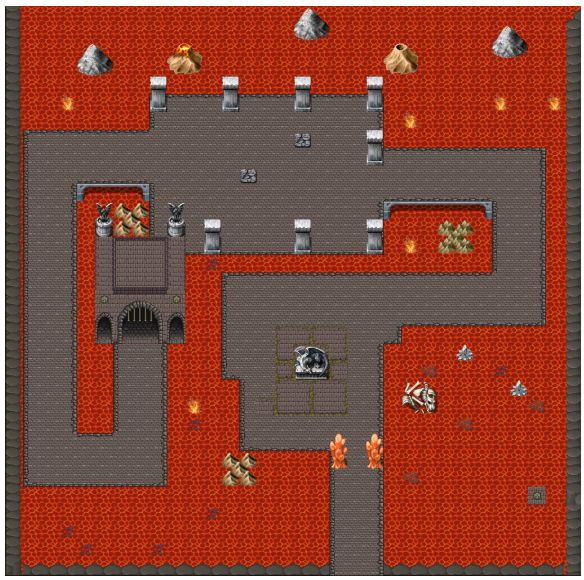
APÊNDICES

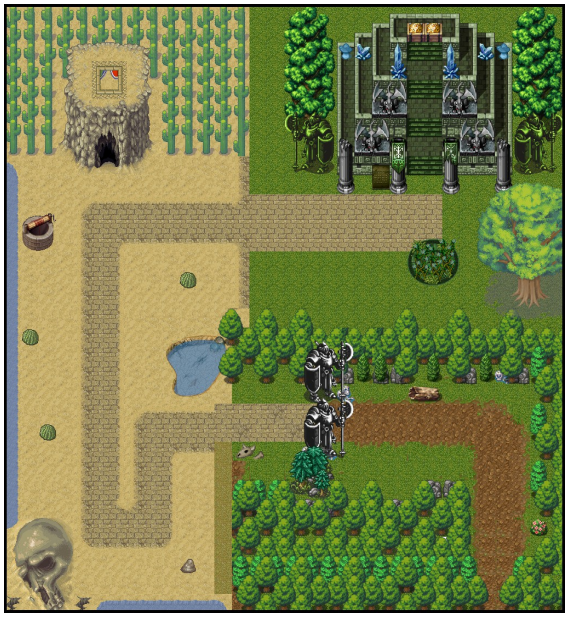
APÊNDICE A – Cronograma



Etapas	Julho	Agosto	Setembro	Outubro	Novembro	Dezembro	Janeiro	Fevereiro	Março	Abril	Maio	Junho
Definição do Tema	X X											
Definição do Ambiente de Destino	X X											
Analise do projeto - definição dos objetivos justifivativa		X X X X	X									
Fundamentação Teórica do projeto			X X X X	X								
Desenvolvimento do GDD				X X X X	X X X X	X						
Adaptação ao Ambiente de Desenvolvimento adt+LibGDX				X X	X X X X	X X X X	X X X X	X X X X				
Modelagem UML do Projeto					X X	X X X X	X X X X	X X	X X X X	X X X X	X X X X	
Desenovolvimento do Game									X X	X X X X	X X X X	X X
Testes e Avaliação final do Projeto												X X X X

APÊNDICE B – Mapas





APÊNDICE C - Questionário

Validação do Jogo do TCC

Este formulário é parte da validação do trabalho de conclusão de cursos dos Acadêmicos Jhonattan e Rogério para validação do jogo desenvolvido.

*Obrigatório

De forma geral o que achou da ideia do jogo? *
- Ruim
- Ok,porem poderia ser melhor abordada
- Boa
- Muito Boa

Quanto a Jogabilidade *
- Ruim
- Moderada
- Boa
- Muito Boa

Quanto ao conjunto de mapa, monstros e dificuldade *

1 2 3 4 5 6 7 8 9 10

Ruim ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Muito Bom

Como o jogo rodou? *
- Não Funcionou
- Travando
- Rodou de maneira aceitável
- Bem
- Muito Bem

Quanto a interface? *
- Ruim
- Aceitável
- Boa
- Muito Boa

Você jogaria um jogo como este? *
- Sim
- Não
- Não tem certeza

De maneira geral o que achou do jogo? *

1 2 3 4 5 6 7 8 9 10

Ruim ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ Muito Bom

O que você acha que poderia ser melhorado?

Sua opnião final sobre o jogo? *

Enviar

APÊNDICE D – Blog

TCC - The Three Sealed Elements

quarta-feira, 28 de maio de 2014

TCC

Este blog foi criado com intuito de colocar a disposição o jogo The Three Sealed Elements criado ao longo do desenvolvimento do trabalho de conclusão de curso do curso de Ciência da Computação da Universidade do Sul de Santa Catarina - UNISUL, pelos acadêmicos Jhonattan e Rogério.

O jogo se encontra para download na versão de Android KitKat sem fins lucrativos.

Gostaríamos de convidá-lo a baixar o jogo no link a seguir, testar o jogo e responder a enquete que será utilizada para validação do trabalho.

Queremos agradecer a todos aqueles que se dispuserem a nos ajudar testando e respondendo ao questionário.

Versão Android: https://www.dropbox.com/s/evb03ampxuz344b/game-android.apk

Questionário:https://docs.google.com/forms/d/1ppzZ9cQnpLjpVstlo5lkBekt7OkpbjzjxkyPMZHiOMY/viewform

O Jogo:

A história começa com o jovem herói retornando a sua cidade natal, lugar conhecido por ser um antigo monastério. Neste lugar monges antigos aprisionaram um mal nunca antes visto, que poderia ter acabado com toda a vida no planeta, e que a transformou em um local sagrado.

O herói que havia chegado de uma jornada de aventura pelo mundo atrás de uma gema, que provêm grandes poderes a pessoa que a utiliza, encontra sua cidade sendo ameaçada por terríveis monstros vindos de outra dimensão, com objetivo de tomar a cidade, e a utilizar como ponto inicial para destruir a humanidade.

O herói então provido de uma fúria para salvar o mundo tentará deter essa invasão. Com a ajuda do ancião da cidade, grande mestre mago, que o guiou anteriormente na sua jornada em busca da gema. E que o guia novamente com missões que o permitiram controlar e ampliar seus poderes. Para que assim ao final ele possa enfrentar o mal que será libertado, caso ele não consiga impedir a invasão, destruindo os cristais dos três elementos, que abrem o portal para invasão, antes que eles alcancem o que está escondido nas profundezas da cidade.

Controles:

Todas as funções são acessadas através da tela *touch* do aparelho em seus respectivos botões.

- Setas : movimentação do personagem.
- Botão vermelho : ataque do personagem.
- Botão verde : utiliza a poção para regenerar HP, utilizando uma carga da poção que é recarregada a cada cinco monstros.
- Botão roxo :pega os itens próximos ao personagem.
- Botão cinza: abre o inventário do personagem

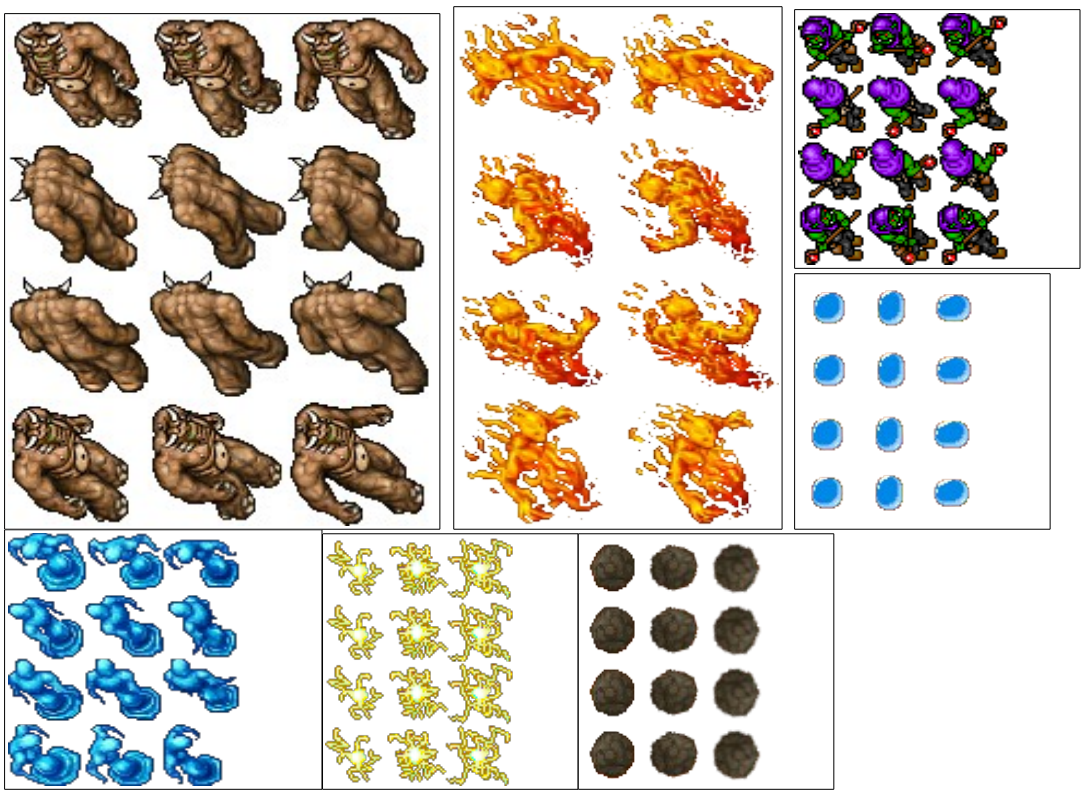
Créditos:

Este jogo foi desenvolvido utilizando de imagens de outros jogos encontradas na internet respectivamente:

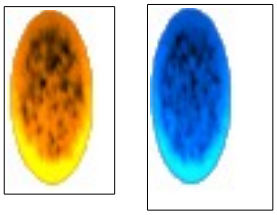
- Tibia desenvolvido por CipSoft.
- Hero of Allacrost.
- Portal desenvolvido por Valve Corporation
- Hozz texture pack para minecraft.
- Tomba! Playstation 1 jogo desenvolvido pela Whoopee Camp.
- Kirby and the Amazing Mirror desenvolvido Flagship.

ANEXOS

ANEXO A – SPRITES SHEETS

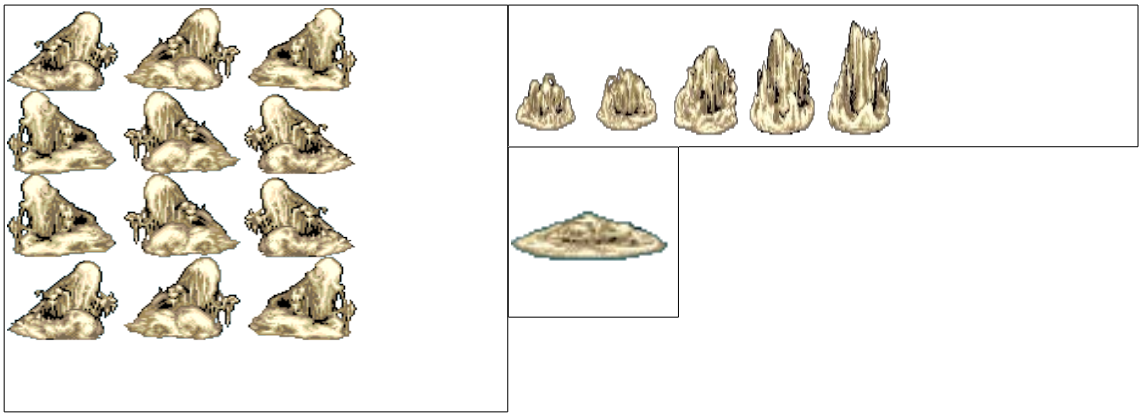


Fonte : Tibia desenvolvido por CipSoft

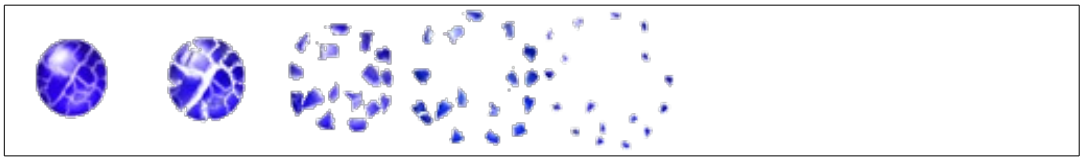


Fonte : Portal desenvolvido por Valve Corporation

Fonte:Hero of Allacrost.



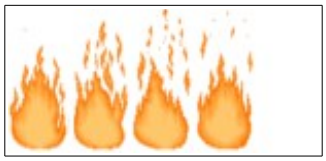
Fonte:Naruto: Saikyō Ninja Daikesshu desenvolvido por D3 Publisher



Fonte: http://gamedevjuice.wordpress.com/2008/03/07/bitmapmovieclip/

Fonte: http://www.rpgmakervx.net/lofiversion/index.php/t8973.html

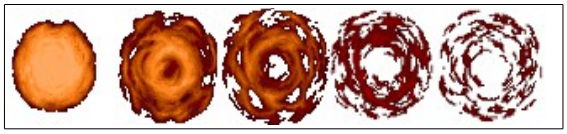
Fonte:Tomba! por Whoopee Camp.



Fonte:http://spritefx.blogspot.com.br/2013/04/fire-sprites.html

Fonte: Kirby and the Amazing Mirror por Flagship

Fonte: Captain Claw por Takarajimasha.



Fonte:http://forum.zdoom.org/viewtopic.php?f=37&t=35895

Fonte: Game Development Unlimited.

Fonte: Hozz texture pack para minecraft