UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA

WALLAS ANDRADE PEREIRA

EDDY
DEMO DE UM JOGO PARA TELEFONE CELULAR

VITÓRIA DA CONQUISTA

2012

WALLAS ANDRADE PEREIRA

EDDY
DEMO DE UM JOGO PARA TELEFONE CELULAR

Monografia apresentada à Universidade Estadual do Sudoeste da Bahia como requisito parcial para obtenção do título de Bacharel em Ciência da computação.

Orientador: Prof. Dr. Fabio Moura Pereira

VITÓRIA DA CONQUISTA

2012

FOLHA DE APROVAÇÃO

WALLAS ANDRADE PEREIRA

EDDY
DEMO DE UM JOGO PARA TELEFONE CELULAR

Monografia do Curso de Ciência da Computação da Universidade Estadual do Sudoeste da Bahia, campus de Vitória da Conquista - BA, como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Aprovado em: ____/____/____.

BANCA EXAMINADORA

_____

Prof. Dr. Fabio Moura Pereira (Orientador)

_____

Prof.ª. Ma. Maísa Soares dos Santos Lopes

_____

Prof.ª. Ma. Maria Silva Santos Barbosa

Vitória da Conquista, **MARÇO** de **2012**

Aos criadores de mundos e seres fantásticos

Dedico

AGRADECIMENTOS

Aos meus familiares pelo apoio, confiança e empolgação com que recebiam cada nova vitória, mesmo sem exatamente entender o que significavam;

Ao meu orientador pela motivação e confiança constantes;

A todos os criadores de todos os jogos que me cativaram desde a infância e que me motivaram a investigar essa arte fascinante;

A Deus o primeiro arquiteto de um mundo fascinante.

“Qualquer um pode tornar o simples em complicado.
Criatividade é tornar o complicado em simples.”

Charles Mingus

RESUMO

A facilidade de acesso aos telefones celulares, acompanhada pela crescente evolução da capacidade computacional desses dispositivos tem promovido uma grande penetração dessa plataforma no cotidiano e possibilitado a execução de aplicativos o que só era possível nos computadores. Os *games* ou jogos eletrônicos também fazem parte desses aplicativos e destacam-se não só por apresentar interação imersiva e atraente para o jogador, mas também por proporcionar uma experiência de desafio acadêmico e criativo para o desenvolvedor. Este trabalho apresenta as diferentes mídias requeridas para um jogo e as principais características de seu processo de desenvolvimento, com enfoque nos telefones celulares. São também apresentadas as principais plataformas disponíveis para o desenvolvimento para dispositivos móveis. Finalmente é apresentado um estudo de caso de um jogo para telefone celular.

Palavras chave: Jogos Eletrônicos. Telefones celulares. Desenvolvimento.

ABSTRACT

The easy access to the mobile phones along with the raising processing capability of these devices have improved the use of these kind of platform in everyday life and have permitted execution of applications, which was only possible on the computers. Games are also part of those applications excelling not only at presenting immersive interaction to the player but also at offering an experience of academic and creative challenge to the developer. This paper presents the different media required in a game and characteristics of its development process, with emphasis on mobile phones. The main mobile devices development platforms are also presented. Finally a mobile game demo version is presented.

Keywords: Electronic games. Mobile Phones. Development.

SUMÁRIO

CAPITULO 1

INTRODUÇÃO

O jogo é um elemento cultural universal que pode não só se manifestar de diferentes formas, mas também de uma mesma forma em diferentes grupos humanos (MOITA, 2004). Ele tem papel importante na formação cultural de um povo; é através dele que a civilização surge e desenvolve-se (HUIZINGA, 1999 apud MOITA, 2004), além disso, manifestações culturais importantes como música, dança e poesia têm seu surgimento no jogo (HUIZINGA, 2000 apud ABREU, 2003).

Os jogos causam fascínio e atração aos seres humanos cativando-os não só no período da infância, mas durante toda a vida. Essa atração se dá principalmente porque os jogos podem oferecer a possibilidade de escape voluntário da realidade e imersão em uma nova realidade. Esse aspecto não só os torna atrativos e fascinantes, mas também é preceito fundamental para a formação cultural humana. (ABREU, 2003).

Definir precisamente um jogo é algo complexo, pois os jogos se manifestam de diversas formas, podendo envolver diversas atividades e recursos que os tornam demasiadamente diversificados (RAMOS, 2006). Segundo Moita (2004) o jogo é uma atividade regida por regras que definem derrota ou vitória, atividade essa que pode ser tanto de natureza física quanto mental. Huizinga (1993, apud RAMOS, 2006) afirma que esta atividade é dada de maneira voluntária e consciente, de maneira não séria e externa à vida habitual, sendo capaz de envolver o jogador intensa ou totalmente. Diante disso, podemos identificar um jogo pela presença de atividades regidas por um conjunto de regras alheias às do mundo real, que definem condições de vitória ou derrota. É ainda característica do jogo a capacidade de simular uma nova realidade envolvente para o jogador, que é alguém que se propõe voluntariamente a ser envolvido pelo ambiente proposto pelo jogo.

Considerando as características apresentadas é possível inserir os jogos eletrônicos ou games no conjunto mais abrangente dos jogos junto às brincadeiras e jogos tradicionais, sendo possível destacá-los dentre os últimos pela sua capacidade de integrar diversas linguagens e mídias, apresentando um ambiente virtual multimídia (RAMOS, 2004).

A evolução tecnológica dos jogos eletrônicos possibilitou o avanço no detalhamento

dos ambientes e interatividade, possibilitando uma maior imersão dos usuários, desta forma aprimorou a construção de uma realidade alternativa. Essa maior virtualização favorece a visão dos jogos eletrônicos como algo mais atrativo, fora do comum, o que possibilita aos games passarem a ser incorporados à cultura mundial (RODRIGUES; LOPES; MUSTARO, 2007). As qualidades dos games atraem a atenção de estudiosos das mais diversas áreas no intuito de que os games podem e devem ser usados de forma planejada e direcionada, das mais diversas formas e com objetivos definíveis, apresentando vantagens em relação às mídias tradicionais.

A incorporação dos games à cultura mundial e a difusão dos meios de comunicação digital contribuíram para a migração dos games para dispositivos além dos computadores e consoles, atualmente é possível encontrar jogos eletrônicos distribuídos por canais a cabo (ARANHA, 2004), incorporados em aparelhos de televisão e DVD e executados em dispositivos móveis, entre outros.

Dentre as mais recentes plataformas de jogos destacam-se os telefones celulares, sobretudo por sua difusão e integração de mídias. O número de celulares atualmente no mundo gira em torno de 3 bilhões de unidades, um número três vezes maior que o número de computadores (AFP, 2007), com o aumento do poder computacional dos aparelhos suas funções também se tornaram mais numerosas somando à tradicional comunicação por voz, possibilidades de tratamento e transmissão de diversos conteúdos como vídeos, imagens, músicas e textos (LEMOS, 2004), além da possibilidade da execução de aplicativos mais elaborados o que inclui games.

Apesar das limitações dos celulares, os jogos voltados para esses aparelhos já constituem uma indústria lucrativa que conta com plataformas de desenvolvimento especificamente voltadas para eles (NOGUEIRA; LOUREIRO FILHO; ALMEIDA, 2005) e já pressiona os fabricantes a lançarem equipamentos mais potentes que possibilitem jogos com mais recursos e qualidade.

Os jogos eletrônicos constituem objetos de estudo e aplicação de técnicas atrativas para a Computação, sua complexa integração de diversas mídias de forma a construir um todo não só tecnicamente bem construído e conciso, mas também que possa proporcionar diversão ao usuário final, constituindo, assim, um desafio acadêmico fascinante, pois muitas vezes requerem a utilização de técnicas sofisticadas que são o que há de mais recente no conhecimento de computação nas mais diversas áreas tais como, computação gráfica, otimização de algoritmos, inteligência artificial.

O objetivo desse trabalho é mostrar o processo e os aspectos característicos do

desenvolvimento de jogos e como se dá tal desenvolvimento no âmbito dos telefones celulares, bem como apresentar as ferramentas disponíveis para o desenvolvimento nos tais dispositivos. São também apresentados aspectos de mercado dos jogos eletrônicos com ênfase no desenvolvimento de jogos no Brasil.

O presente estudo foi dividido em quatro capítulos. O Segundo capítulo apresenta aspectos do desenvolvimento de jogos eletrônicos, as mídias que os compõe e o processo de desenvolvimento, também são abordados aspectos de mercado e sua classificação, por gênero.

No capítulo 3 é apresentada a produção de jogos para telefones celulares, suas características e limitações, são também apresentadas às principais plataformas para a produção de jogos. O mercado para esse tipo de jogo é discutido em maiores detalhes acompanhado de casos de sucesso da produção brasileira desse tipo de game. No capítulo 4 é apresentado o projeto do demo de jogo desenvolvido e alguns aspectos de seu desenvolvimento. Finalmente no capítulo 5 são apresentadas as considerações finais.

CAPÍTULO 2

DESENVOLVIMENTO DE JOGOS ELETRÔNICOS

O desenvolvimento de um jogo eletrônico envolve diversas atividades e artefatos, num processo complexo não padronizável que pode variar conforme o projeto, no entanto alguns aspectos são comuns à maioria dos projetos de jogos eletrônicos. Este capítulo aborda tais aspectos, sua importância e técnicas comuns utilizadas. São também abordados os aspectos de mercado dos games e classificação dos mesmos.

2.1 – GÊNEROS DE JOGOS

A boa definição e especificação do tipo de jogo a ser desenvolvido irão definir várias de suas características, desde a interface até o motor do mesmo, exercendo influência na técnica de implementação que será usada no projeto. O tipo de jogo também irá definir o método de interação do jogador no universo do jogo (BATTAIOLA et al, 2001).

A seguir são apresentados os principais gêneros de jogos eletrônicos:

- Ação: Jogos que focam a interação de forma a testar a reação do jogador em seu tempo de resposta entre visão e reação. É comum nesse tipo de jogo, que a ação esteja centrada em um personagem que apresenta diversas possibilidades de movimentos para ação e combate, esta categoria inclui os jogos de tiro, luta, plataforma, entre outros (SANTANA, 2006). Um exemplo de jogo de ação é o jogo Uncharted produzido pela Naughty dog.
- Jogos de aventura: Exigem do jogador raciocínio e reflexo, direcionando-o à solução de enigmas e quebra-cabeças (SANTANA, 2006), é também característica dos jogos desse gênero a valorização da exploração dos cenários e a ênfase reduzida nos elementos de ação e combate (LUZ, 2004). Um exemplo de jogo com enfoque em aventura é Tomb Raider da Eidos, o jogo mescla elementos de ação e aventura, mas foca-se na exploração de ambientes, o que é típico dos jogos de aventura.
- RPG: Estilo derivado dos tradicionais jogos de RPG, nesse gênero o jogador assume o papel de um personagem ou grupo de personagens, sendo guiado por um roteiro

pré-determinado pela história no mundo do jogo (VIEIRA FILHO, 2005). É também típico desse gênero de jogo a possibilidade de aprimorar as características do personagem durante o jogo, através de batalhas e aquisição e alteração de armas e equipamentos. O jogo Final Fantasy XIII da Square-Enix é um exemplo de jogo de RPG.

- Simulação: Simulam situações específicas da vida real, tentando apresentar o máximo realismo possível, através da inserção de limitações, características e aspectos físicos conhecidos do mundo real, existem diversos tipos de jogos de simulação, as variações incluem simulação de voo, de corrida, simulação da vida, simulação e gerenciamento de construções (VIEIRA FILHO, 2005). Um exemplo de jogo de simulação é o jogo Flight Simulator X da Microsoft, o jogo aborda simulação de pilotagem de aviões.

- Estratégia: Nesse tipo de jogo o jogador interage através do gerenciamento dos diversos recursos disponíveis, a fim de atingir determinado objetivo, subdividem-se em dois tipos: Os jogos baseados em turno, onde cada jogador faz sua jogada e passa a vez; E jogos de tempo real, onde o gerenciamento deve ser feito em tempo real. Um exemplo de jogo de Estratégia é Age of Empires da Microsoft.

- Esportes: Jogos que simulam os esportes tradicionais, tais como futebol, basquete, tênis, boxe entre outros (SANTANA, 2006). Existem duas ênfases básicas nos jogos de simulação esportiva: O jogador pode atuar sob o ponto de vista do atleta, realizando jogadas, passando a bola, driblando ou o jogador pode assumir o papel de técnico, gerenciando o time e traçando estratégias (VIEIRA FILHO, 2005). Os jogos da série FIFA da ElectronicArts (EA) são exemplos de jogos de esporte.

- Passatempo: Jogos de interface simples e fácil interação tratam-se na maioria dos casos de quebra-cabeças sem nenhuma história relacionada. O principal objetivo do jogador nesse tipo de jogo é atingir uma alta pontuação (BATTAIOLA et al, 2001). Um exemplo desse gênero de jogo é Bejeweled da Astraware.

- Jogos em rede/ Online: Jogos que são jogados por múltiplos jogadores, através da Internet, podem englobar jogos de qualquer dos gêneros anteriormente citados. Podem ser divididos em dois tipos básicos: Os jogos com opção de jogo em rede, onde os jogadores escolhem se preferem jogar sozinhos ou contra um adversário em rede e os MMO, *Massive Multiplayer Online*, cuja única forma de jogar é através da Internet (SANTANA, 2006). Um exemplo de jogo online é Counterstrike da Half-life, um jogo online de tiro.

2.2 – ASPECTOS DO DESENVOLVIMENTO

A seguir são apresentados aspectos comumente envolvidos no desenvolvimento de jogos eletrônicos.

2.2.1 – ARTE GRÁFICA

A interface gráfica é um das partes mais notáveis num jogo, sua qualidade define muito do realismo do jogo e da imersão do jogador no mundo do jogo, conferindo um atrativo de suma importância para o sucesso do jogo (BATTAIOLA et al, 2001).

A construção da interface gráfica de um jogo envolve não só todos os elementos em tempo de jogo, tais como, personagens, cenários, itens e armas. Mas também os elementos de telas auxiliares, como telas de carregamento e menus (CLUA; BITTENCOURT, 2005).

2.2.1.1 – JOGOS 2D

Os jogos 2D são caracterizados pela utilização dos *Sprites* para animação e pela utilização de Tiles , *Layers* e *Bricks* para a construção de cenários e planos de fundo.

A seguir são apresentados os conceitos dessas estruturas e sua utilização na construção da parte gráfica de um jogo bidimensional.

- *Tiles*: São imagens divididas em partes do mesmo tamanho, a Figura 1 apresenta um conjunto de *tiles* baseados no jogo Super Mario Bros. da Nintendo.

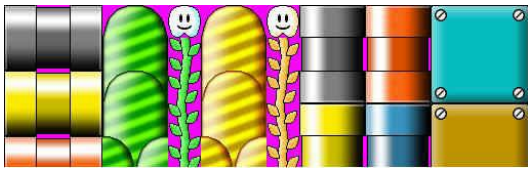


Figura 1: Exemplos de Tiles

- *Bricks:* São blocos de imagens, que podem ser usados para representar uma estrutura no cenário ou parte do cenário, *bricks* são formados por grupos de tiles, a Figura 2 mostra um exemplo de cenário construído com *Tiles* e *bricks* é possível observar a construção de pontes aéreas com *tiles* de tijolos e o céu com *tiles* de cor azul (SANTANA, 2006).

Figura 2: Exemplo de cenário formado com Bricks

- *Layer* ou camada é um objeto gráfico constituído por uma matriz de *tiles* e *bricks*, que podem ser sobrepostas para a construção dos cenários.

Alguns jogos utilizam diversas camadas ou backgrounds. Um exemplo que permite ver essa sobreposição de camadas é dado nas figuras seguintes, onde é possível comparar uma tela completa (Figura 3) a uma onde foi removido um dos planos de fundo, que nesse caso é responsável pela representação dos rios e lagos do cenário (Figura 4), a tela apresentada pertence ao jogo Donkey Kong 3 da Rare.

Figura 3: Exemplo de sobreposição de Camadas

Figura 4: Exemplo de cenário com remoção de um camada

- *Sprites*: Tratam-se de conjuntos de imagens que representam diferentes posições de um mesmo objeto ou ator do jogo que são substituídas em intervalos definidos de tempo, de

forma a simular o movimento do elemento em questão na tela, um exemplo de conjunto de *sprites* é apresentado na Figura 13, que mostra alguns dos *sprites* relativos ao personagem Knuckles do jogo Sonic & Knuckles da Sega(BATTAIOLA et al, 2001).

Figura 5: Exemplo de conjunto de Sprites

Em um jogo 2D todos os elementos que apresentem algum tipo de movimento utilizam a técnica de animação por *sprites* (SANTANA, 2006), as imagens de *sprites* são geralmente retangulares, o que faz necessário o uso de técnicas para remoção dos planos de fundos das imagens dos *sprites* , duas técnicas são utilizadas o uso de cores transparentes e a aplicação de máscaras (BATTAIOLA et al, 2001), o uso de transparência é feito através de uma cor que é ignorada durante a cópia dos *sprites* (PERÚCIA, 2005, p.69 apud SANTANA, 2006, p.83), a aplicação de máscaras utiliza operações de AND e XOR entre máscaras, plano de fundo e *sprites*, primeiro ocorre um AND entre plano de fundo e máscara, em seguida esta área sofrerá uma operação de XOR junto com o *sprite* desejado.

Dados podem ser associados aos *sprites* a fim de conferir-lhes mais funções, tais como posição, tamanho, estado de visibilidade, também são possíveis transformações, tais como rotação, translação e detecção de colisões (BATTAIOLA et al, 2001).

2.2.1.2 – JOGOS 3D

A construção de gráficos tridimensionais começa geralmente com a produção de desenhos 2D, que são convertidos para modelos tridimensionais através de softwares de modelagem 3D, como o Blender (NACKE, 2005).

O processo de construção de um modelo tridimensional passa por dois processos básicos, a modelagem e a renderização. A modelagem é realizada basicamente com a combinação de polígonos mais simples, para gerar o modelo complexo pretendido. Essa etapa

é realizada utilizando a representação de malha poligonal ou fios de arame (BATTAIOLA et al, 2001), essa representação consiste em exibir somente as arestas e vértices dos polígonos.

Uma vez que o objeto tridimensional esteja modelado, ele passará pelo processo de renderização, que consiste no preenchimento e aplicação de simulações de características de objetos reais como sombras, cor, rugosidade e brilho.

Uma das técnicas mais utilizadas para obter realismo é o uso de texturas. Texturas são imagens 2D usadas para representar um material do mundo real, essas imagens são aplicadas nas faces dos polígonos que constituem os modelos (CLUA; BITTENCOURT, 2005).

Jogos eletrônicos 3D utilizam renderização em tempo real, ou seja, os modelos são preenchidos e sofrem efeitos tais como iluminação e sombras em tempo de execução. Alguns jogos 2D, 2,5D e *sprites* 3D utilizam cenas pré-renderizadas: Imagens de cenas produzidas e renderizadas num editor 3D que são armazenas e utilizadas como *sprites* ou *tiles*.

2.2.2 – SONS

Os sons têm um papel fundamental no jogo, confere sentindo ao mundo do jogo, faz com que os personagens pareçam de carne e osso e é parte essencial da cena, pois é capaz de criar uma atmosfera de realismo; podendo guiar as ações do jogador (LIUKKONEN, 2006), portanto o áudio é fundamental para que a experiência de jogar seja ainda mais emocionante (BATTAIOLA et al, 2001).

A inserção de sons em um jogo deve considerar os vários tipos de áudio que devem ser usados para que se obtenha êxito na experiência de interação do jogador, existem basicamente dois tipos de som que podem ser inseridos em um jogo, as trilhas sonoras, que são as músicas do jogo; e os efeitos sonoros que são sons e ruídos que acontecem em determinados momentos ou cenários do jogo, tais como, passos, barulhos de frenagem, sons de animais e colisões (BATTAIOLA et al, 2001).

O formato do arquivo de som a ser utilizado no jogo irá variar conforme o tipo de sons que se pretende utilizar, fatores como qualidade, interatividade e tamanho irão variar entre os formatos disponíveis. A escolha correta do formato de som deve equilibrar qualidade às limitações que o hardware da plataforma escolhida impõe, assim sendo a maioria dos jogos utiliza mais de um formato. Os formatos de armazenamento ou codificação de sons mais utilizados são MIDI, wav, mp3, RedbookAudio e OGG Vorbis (AZEVEDO, 2005).

O som bem colocado pode melhorar a interação do jogador, mas o som inadequado ou inserido de forma ou em momento inadequados pode fazer com que o jogador perca sua atenção (AZEVEDO, 2005), sons fora de contexto devem ser evitados, tais como um som de tiro de revólver para uma arma de um jogo futurista, sons repetitivos e músicas muito curtas e repetitivas (LIUKKONEN, 2006).

2.2.3 – INTELIGÊNCIA ARTIFICIAL

As diversas técnicas e abordagens de IA visam capacitar programas a assumir comportamentos tipicamente humanos tais como tomada de decisões, resolução de programas e aprendizado, tais comportamentos são cada vez mais desejados nos jogos eletrônicos, pois implicam em maior realismo e uma experiência mais profunda para o jogador (AZEVEDO, 2005).

A Inteligência Artificial para jogos pode ser entendida como o conjunto de programas que irá descrever o comportamento dos personagens do jogo não controlados pelo jogador, conhecidos como *Non-Playable Characters* ou NPC (CLUA; BITTENCOURT, 2005), acrescentando características humanas aos elementos do jogo suficientes para tornar o jogo mais divertido (LAIRD; VAN LENT, 2004).

A inserção da IA em um jogo irá variar conforme o gênero do jogo e de jogo para jogo, no entanto algumas aplicações mais comuns podem ser citadas: Parceiros e inimigos táticos, são comuns em jogos de ação, precisam atacar e se defender de ataques e também localizar os inimigos; Personagens de apoio, são comuns em jogos de RPG, tratam-se, por exemplo, dos mercadores das vilas e moradores, têm geralmente comportamento limitado e baixa taxa de interação, cumprindo um script bem definido e repetitivo; Inimigos estratégicos são típicos de jogos de estratégia e de jogos de esportes em grupo, tem como objetivo alocar recursos e comandar a distribuição de unidades, através de uma estratégia pré-definida ou variável; Unidades trata-se dos requisitos necessários às unidades referentes à jogos de estratégia para cumprir os comandos recebidos, como por exemplo reagir a inimigos encontrados; Oponentes em corridas: Precisam ser capazes de conduzir um veículo através de um circuito, levando em consideração as capacidades do veículo, circuito e leis físicas.

Uma das técnicas de inteligência mais utilizadas em jogos é a de máquinas de estados finitos, Essa técnica funciona definindo o comportamento do personagem a partir de estados nos quais ele possa encontra-se, com ações possíveis em cada um deles e fatores externos ou

internos que possam disparar a mudança de um estado para outro, como exemplo, um inimigo em jogo de tiro que esteja em estado de espera, ao avistar o jogador entra em modo (estado) de combate, até que sua munição acabe, tal fato o obriga a entrar em modo de fuga. Outras técnicas de IA também têm sido usadas nos jogos com sucesso tais como, agentes autônomos, possuindo inclusive bibliotecas e ferramentas de apoio como é o caso do DirectIA do pacote DirectX da Microsoft (CLUA; BITTENCOURT, 2005).

É importante ao se escolher alguma técnica de IA que o comportamento humano além de ser custoso, difícil ou mesmo impossível de atingir, pode também ser indesejável em toda a sua extensão, já que pode implicar em diminuição da diversão.

2.3 – PROCESSO DE DESENVOLVIMENTO

O processo de desenvolvimento de jogos eletrônicos não apresenta nenhum padrão formal ou acordo de definição de fases, tarefas ou mesmo nomenclaturas, no entanto, segundo Nacke (2005) existem algumas características que são comuns a todos os projetos de jogos eletrônicos, tais como, interdisciplinaridade, ciclos de desenvolvimento iterativos (com refinamento incremental), desenvolvimento colaborativo e desenvolvimento dirigido por custos.

A produção de um jogo envolve o trabalho em conjunto de uma equipe multidisciplinar, onde o produto de uma equipe serve como objeto de trabalho de outra equipe, num processo de melhoria incremental, que culmina com a produção da versão ouro que é a versão final a para distribuição ao consumidor (BREYER et al, 2006). Durante o processo, a produção passa por várias fases de desenvolvimento, tais fases não são estritamente definidas, podendo sofrer variações, assim como a transição entre elas pode, também não ser tão clara (NACKE, 2005).

De forma geral pode-se afirmar que um projeto de jogo passar por quatro fases básicas: a concepção, pré-produção, prototipação ou primeiro protótipo funcional e a fase de produção (ELECTRONIC ARTS, 2005):

- Concepção: Todo projeto de jogo parte de uma idéia inicial. A concepção envolve não só a exposição e coleta de ideias, mas também uma análise destas levando-se em consideração alguns aspectos mais concretos como: público-alvo, plataforma de desenvolvimento, originalidade e possibilidades de mercado (SANTANA, 2006), a fase de concepção deve terminar com uma descrição simples inicial ou um game-design inicial com a

descrição do jogo e suas características, bem como alguns de seus elementos visuais (NACKE, 2005), esta primeira descrição é a que geralmente é submetida aos publicadores para análise (ELECTRONICARTS, 2005).

- Pré-produção ou game design: É o período que compreende a finalização do game design e planejamento de produção do jogo, já considerando custos e recursos necessários. As ideias descritas no game design inicial são verificadas através da prototipação em ciclos e então são incluídas no documento de desenvolvimento final, os protótipos podem ser implementados ou até mesmo simulados utilizando qualquer recurso material de forma a esclarecer para a equipe algum aspecto ou aspectos do jogo, como por exemplo um desenho ou tabuleiro (NACKE, 2005). A fase de pré-produção ou game design deve terminar com um game design que inclua a maioria das características do jogo, devendo ser tratado de forma a se tornar o documento de consulta da equipe de produção a qualquer momento, portanto toda informação relevante para o projeto deve ser incluída (LARAMEE, 1999).
- Primeiro protótipo funcional: Compreende a produção de uma primeira versão do jogo onde as primeiras características descritas no game design devem estar funcionando, sem necessariamente estar completas ou estáveis (BREYER et al, 2006). É nesta fase que as ferramentas, tais como geradores de mapas e controladores de animação são produzidos, iniciam-se a construção do motor do jogo ou o aprendizado de um *middleware* de terceiros e a construção das artes visuais, tais como modelagem de personagens e criação de texturas, bem como a produção dos sons do jogo. (ELECTRONICARTS, 2005).
- Produção: É alcançada de forma não muito clara, geralmente pode-se afirmar que um jogo está em fase de produção quando é possível observar a integração dos elementos produzidos nos diversos departamentos: motor do jogo, ferramentas, modelos visuais e som (NACKE, 2005).
- Fase Alfa: Antes que um jogo em produção possa ser lançado, ele deve passar por duas fases: A fase alfa e a fase beta (ELECTRONICARTS, 2005). Um jogo atinge a fase alfa quando todas as características descritas para ele estão implementadas, o foco da fase alfa é a realização de testes, por parte da própria equipe de desenvolvedores, a fim de detectar e corrigir erros, tais como balanceamento, problemas de interface entre outros, portanto é importante evitar-se o acréscimo de qualquer nova propriedade (BREYER et al, 2006).
- Fase Beta: A fase alfa termina quando não há mais nenhum erro, conhecido pela equipe, que impeça o jogo de estar pronto para ser distribuído. O jogo passa então para a fase beta, nesta fase o jogo irá passar por testes realizados por pessoas externas à produção, que terão suas opiniões recolhidas e avaliadas pela equipe de produção (BREYER et al, 2006).

Alguns jogos especialmente para computadores que já estão finalizados quando da distribuição, podem precisar de manutenção posterior ao lançamento, tais como adaptações a hardware, software, novas propriedades e correção de erros não detectados anteriormente. As correções são geralmente disponibilizadas em pacotes chamados *patches*, que podem ser baixados gratuitamente nas páginas dos desenvolvedores ou publicadores (NACKE, 2005).

2.4 - MERCADO

O Mercado de jogos eletrônicos é principalmente segmentado em função da plataforma de execução dos mesmos. As principais delas são os consoles de videogame e os computadores, nos últimos anos o aumento das capacidades de processamento e armazenamento dos dispositivos móveis, tem apontado estes como uma nova e lucrativa plataforma de distribuição, principalmente no que diz respeito aos telefones celulares. (RAHAL, 2006).

2.4.1 - JOGOS PARA CONSOLES

Os principais consoles atuais são o Wii da Nintendo, o Xbox 360 da Microsoft e o Playstation 3 da Sony. A sétima geração de consoles foi iniciada com o lançamento do Xbox 360 em 2005, seguido pelos demais consoles em 2006. Os consoles da Sony e Microsoft focaram-se em poderoso hardware capaz de exibir gráficos em alta definição, enquanto que o console Nintendo apresentava gráficos mais simples mas trazia uma nova forma de interação com controles com sensores de movimento (WIKIPEDIA, 2011)

Uma das características da geração atual são os controles com sensores de movimento, o sucesso do sistema de controle inicialmente proposto no Wii fez com que Sony e Microsoft também apresentassem controles sensíveis a movimentos para seus aparelhos (MORRIS, 2010). Dois outros pontos marcantes nessa geração são aumento das capacidades de armazenamento com inclusão de Discos rígidos e memória flash e o aprimoramento das redes proprietárias que além de jogos on-line, oferecem outros serviços, como atualizações de sistema e download de jogos, incluindo os de consoles anteriores (NINTENDO, 2010; SCE, 2010; MICROSOFT, 2010). Os consoles representam a parcela mais lucrativa e concorrida do mercado de games. (SPOHN, 2002). O desenvolvedor de jogos para consoles deve possuir

uma licença para a produção do jogo voltado ao dispositivo escolhido e ainda precisa adquirir um kit de desenvolvimento para o mesmo, junto ao fabricante do console (CARNOY, 2005).

O setor representado pelos consoles é o que representa o maior desafio aos desenvolvedores do Brasil. Poucos fabricantes de consoles se arriscam a penetrar oficialmente no mercado brasileiro, os motivos principais são a carga tributária nacional, e o alto índice de pirataria. Cerca de 90% dos jogos nacionais são ilegais, bem como grande parte dos consoles atualmente no Brasil são importados de forma ilegal (WHARTON SCHOOL, 2004).

Apesar das dificuldades, algumas iniciativas, ainda que isoladas foram tomadas em direção à conquista deste mercado, como a aquisição de licença de produção de games para um console, por parte de uma produtora nacional (LUZ, 2004), e o lançamento oficial dos consoles Xbox 360 em 2006(UOL JOGOS, 2006) e Playstation 3 em 2010 (SOUZA FILHO, 2010) no Brasil.

2.4.2 – JOGOS PARA COMPUTADORES

O desenvolvimento de jogos para computadores é feito de forma aberta, ou seja, qualquer desenvolvedor que disponha das ferramentas necessárias, conhecimento e equipamento adequado pode desenvolver um jogo para computador. Há ainda a possibilidade de utilizar ferramentas gratuitas de boa qualidade, que abrangem desde a criação gráfica até o motor básico de jogo (COELHO, IORIO, 2005).

A maioria dos jogos produzidos atualmente é dirigida à plataforma Windows, sobretudo pelas facilidades e constantes atualizações oferecidas pelo DirectX da Microsoft, exclusivo desta plataforma (SANTEE, 2005).

O computador oferece muito mais do que uma plataforma de jogo, sua flexibilidade também pode oferecer aos distribuidores e consumidores modelos de negócio bastante diferenciados. Muito além do processo tradicional de vendas possível para os jogos de consoles, um exemplo disto é o modelo adotado por algumas distribuidoras que oferecem o download gratuito de seus jogos, multiusuário via Internet, e cobram pelo acesso aos servidores ou pela venda de itens do jogo (LONGO, 2006).

Como plataforma o computador oferece mais flexibilidade, suas funções online não são dependentes ou limitadas a redes proprietárias, a evolução de hardware dos computadores é muito mais rápida, o que os coloca sempre à frente dos consoles, além disso, a atualização de hardware no PC é mais simples, pois não é necessária a troca total do equipamento,

bastando a substituição de componentes isolados (SPOHN, 2010).

A evolução dos consoles de videogame têm abalado o mercado de jogos para computadores, chegando a levantar questões a cerca do futuro dos jogos eletrônicos nesta plataforma, principalmente após a incorporação das tecnologias de rede nos consoles domésticos (SPOHN, 2002).

2.4.3 – JOGOS PARA DISPOSITIVOS MÓVEIS

O aumento das capacidades de processamento, armazenamento e conectividade apresentado pelos aparelhos móveis, incluindo os telefones celulares, possibilitou-lhes a execução de softwares cada vez mais complexos e funcionais, atraindo a atenção da indústria de software para essa plataforma. (NOGUEIRA, LOUREIRO FILHO, ALMEIDA, 2005).

Alguns dos principais desafios desse mercado são as limitações de hardware e a cultura do consumidor. A maioria dos consumidores não leva em consideração a possibilidade de jogar no celular no momento da compra, que é geralmente direcionada pelo preço. O perfil de jogo para o celular também precisa ser diferenciado, pois tais consumidores não estão dispostos a passar horas em frente ao aparelho para jogar. Os celulares possuem, contudo uma vantagem sobre as demais plataformas, mesmo os consoles portáteis: os celulares são plataformas sempre à mão (AMARO, 2003).

A atuação do desenvolvedor no mercado dos aplicativos e jogos para celulares é na maioria das vezes feita em acordos com as operadoras de telefonia móvel ou os fabricantes dos aparelhos. O desenvolvimento junto ao fabricante é geralmente feito com plataformas exclusivas e voltadas a aparelhos específicos que incluirão o jogo ou aplicativo em sua configuração padrão, no modelo de distribuição com apoio das operadoras, os jogos são voltados a um número maior de dispositivos e são disponibilizados aos usuários por venda via download. (NOGUEIRA, LOUREIRO FILHO, ALMEIDA, 2005).

O mercado de jogos para dispositivos móveis tem experimentado um crescimento bastante significativo nos últimos anos, em 2006 alcançou a marca dos 3 bilhões de dólares (PRADO, 2007) e com o crescimento das vendas de aparelhos celulares o mercado tem um grande potencial ainda a ser explorado (SCHNOOR, 2007).

Grandes produtoras e distribuidoras de jogos nas demais plataformas voltam-se também para a produção de jogos para esses dispositivos, tentando alcançar o público com versões móveis de seus sucessos atuais e antigos (REUTERS, 2005), franquias de sucesso

como Sonic the hedgehog da SEGA (SEGA MOBILE, 2007) e Tomb Raider da Eidos (EIDOS MOBILE UK, 2006), entre outros já têm suas versões móveis, além disso essas empresas têm formado divisões específicas para este nicho de mercado, com produção própria e associações com produtoras de sucesso no mercado móvel (REUTERS, 2005).

Empresas que produzem conteúdo exclusiva ou principalmente para dispositivos móveis têm crescido e se expandido, como por exemplo, a francesa In-Fusio que se tornou líder europeia de jogos móveis e hoje já estende sua atuação aos dois principais mercados de jogos móveis do mundo, a Europa e a China (IN-FUSIO, 2002). A empresa Astraware do Reino unido recebeu diversas críticas positivas da imprensa pelo lançamento de um pacote de jogos móveis: Astraware Board Games, que reúne versões digitais de jogos populares de tabuleiro, como Paciência, Dama, Xadrez e Ludo, entre outros (AGOT, 2007), outro sucesso da empresa é o jogo Bejeweled.

A divulgação de marcas através dos *advergames* cada vez mais populares nos computadores, especialmente em suas versões online, para serem jogados nas páginas das empresas (PEREIRA JUNIOR, 2003), também alcançam os dispositivos móveis. As empresas investem na produção de jogos e conteúdos diversos que possam prender a atenção de seus consumidores, um exemplo é a MTV, que anunciou um investimento de 500 milhões de dólares em jogos eletrônicos, incluindo jogos para celulares (LI, 2007), além dela o gigante do *fastfood*, Burger King uniu-se à provedora de jogos e conteúdos móveis Mobliss, a fim de produzir jogos com seu mascote para dispositivos móveis (SEAN, 2007).

Os líderes desse mercado desde o início dos jogos móveis são os países da região Ásia-Pacífico, tendo como principais representantes Japão e Coréia do Sul, em segundo lugar vem a Europa, seguida pelos Estados Unidos (GIBSON, 2006), no Brasil uma pesquisa divulgada pela TecToy em 2006, estimava para o fim daquele ano um faturamento de 30 milhões de dólares em jogos para celulares (FOLHA ONLINE, 2006). A base instalada no país ultrapassa os 100 milhões de aparelhos (MORAIS, 2007).

As limitações de hardware dos celulares, aliadas ao grande número de ferramentas livres para o desenvolvimento criam uma situação de mercado bastante peculiar, onde os grandes estúdios e os pequenos desenvolvedores e iniciantes competem igualmente (AMARO, 2003), o estado atual do desenvolvimento de jogos para celulares pode ser comparado ao desenvolvimento de jogos na década de 1980, quando somente um desenvolvedor poderia se trancar por um mês e produzir um novo sucesso de vendas (ARTHUR, 2007).

As maiores facilidades para ingressar no desenvolvimento de jogos para celulares

favorecem, sobretudo a mercados com pouca ou nenhuma tradição no desenvolvimento de jogos como o brasileiro, já que o mercado das demais plataformas já se encontra dominado pelas grandes produtoras com suas superproduções milionárias (LACERDA, 2006).

2.4.4 – DESENVOLVEDORES BRASILEIROS E O MERCADO NACIONAL

O desenvolvimento de jogos em território nacional, segundo a ABRAGAMES, iniciou-se timidamente nos anos 1980, sofrendo um crescimento mais significativo a partir de 1997 até 1999, quando 21% das 55 empresas existentes até o ano de 2004 foram fundadas (ABRAGAMES, 2005).

A maior parte da produção nacional concentra-se nos jogos para PC, 63% seguido pelos jogos produzidos para aparelhos celulares (ABRAGAMES, 2005). A produção de jogos para consoles tem pouca representatividade, já que a maioria dos fabricantes não lança seus produtos oficialmente no Brasil, desmotivados pela pirataria e os altos impostos (LACERDA, 2006). Mais de 90% dos consoles existentes no país são ilegalmente importados, já que a carga de impostos chega a aumentar o preço dos aparelhos em mais de 200% (AZEVEDO, 2006), além disso, cerca de 60% de todo software nacional é pirata (FUSCO, 2007).

Dois setores têm tido especial destaque no cenário de desenvolvimento de jogos brasileiro, são os jogos para celular, que com seu baixo custo de produção tornaram-se produto de exportação, além de serem acessíveis ao mercado interno e os *Advergames*, jogos para divulgação de marcas (CORTEZ, 2003).

Os *advergames* têm ganhado espaço no Brasil e muitas empresas brasileiras já divulgam suas marcas e produtos dessa forma, especialmente em páginas da web, alguns exemplos são Nívea (NIVEA, 2007), Procter & Gamble Brasil, com a divulgação do sabão em pó Ariel (P&G BRASIL, 2007) e Tilibra que disponibiliza jogos num hotsite de divulgação de seus produtos, no caso da Tilibra a campanha inclui o uso de conteúdo móvel, mas apenas para divulgação do hotsite (DE FALCO, 2007). Apesar de os advergames brasileiros ainda concentrarem-se em páginas de Internet, há alguns casos de advergames feitos diretamente para dispositivos móveis a produtora nacional Overplay de Campinas, já produziu três jogos móveis voltados para campanhas de marketing de empresas, tais como Vivo, Visa e Volkswagen (OVERPLAY, 2007).

Os jogos para celulares produzidos por produtoras nacionais têm boa aceitação também no que diz respeito aos jogos de entretenimento puro. Os mercados nacional e

internacional já vêm oportunidades e parcerias com os desenvolvedores brasileiros, alguns exemplos são: A parceria entre Samsung e CESAR, Centro de Estudos e Sistemas Avançados do Recife, que resultou na produção dos jogos Dragon Wrath, Mobies e Route 66 que são disponibilizados para download na página da Samsung, além dessa parceria o CESAR também já produziu mais de 40 jogos para outras fabricantes de celulares entre elas, LG, Motorola e Nokia (ARAUJO, 2005).

O sucesso dos jogos móveis do Centro Tecnológico de Recife levou à criação da Meantime Mobile Creations, empresa encubada pelo CESAR que produz jogos e outros conteúdos para dispositivos móveis (ARAUJO, 2005). Os jogos Gold Hunter e SeaHunter da Meantime foram distribuídos em 40 operadoras na Europa e Ásia (CORTEZ, 2003), após terem sido classificados entre os 20 melhores no Ásia Java Mobile Challenge em 2003, e um deles, Sea Hunter (Figura 6) classificou-se entre os top 5 (WEBINSIDER, 2003).

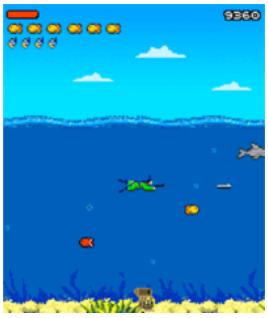


Figura 6: Tela do jogo Sea Hunter

Outra produtora nacional que também se destaca no mercado de jogos móveis é a Overplay, anteriormente chamada Delírius. A empresa tem jogos distribuídos por toda a América latina, Suíça, Alemanha, França, Turquia e Índia (STÄHELIN, 2006), um de seus jogos: Ice post foi sucesso na suíça, chegando a vender 500 cópias em apenas um mês, a Overplay tem também parcerias com operadoras nacionais como Telemig Celular, a Amazônia Celular, Claro e Brasil Telecom (PAGEL, 2005).

No mercado brasileiro é ainda importante citar a TecToy Mobile, braço de entretenimento móvel da TecToy, empresa já tradicional no mercado de games nacional, responsável pela representação dos produtos da SEGA, em território brasileiro. A TecToy mobile foi fundada em Junho de 2005 e tem parcerias com grandes publicadoras e desenvolvedoras internacionais como Sega Mobile, In-Fusio, Pulse Interactive, Bandai e etc. A empresa atua também publicando jogos nacionais e na produção de jogos com um estúdio

próprio (TEC TOY MOBILE, 2007).

As produtoras de games brasileiras concentram-se principalmente nas regiões sul e Sudeste do país, com destaque para os estados do Paraná, com 33% do total de empresas e São Paulo, com 30%.A maioria delas, 72% concentra-se na produção de games de entretenimento puro, seguidas pelas produtoras de Advergames, 14%. Os *middlewares*, ferramentas para produção e manutenção de jogos e Os Business games, simulações de negócio com foco em aprendizado, ficam em último lugar com 8% e 6% respectivamente (ABRAGAMES, 2005).

As desenvolvedoras nacionais faturam cerca de 20 milhões de reais, numa taxa de crescimento de 30% ao ano (LACERDA, 2007). Alguns passos importantes já foram dados em direção ao crescimento da produção nacional de jogos eletrônicos, tais como a criação da Associação Brasileira das Desenvolvedoras de Jogos Eletrônicos, ABRAGAMES, em 2004 (ABRAGAMES, 2004), e a realização do jogosBr, última edição em 2006, concurso promovido pelo Ministério da Educação e Cultura, para fomentar o setor, oferecendo recursos aos vencedores.

CAPÍTULO 3

JOGOS PARA TELEFONES CELULARES

Os aparelhos celulares sofreram uma grande evolução desde o primeiro jogo Snake (Figura 7), lançado pela Nokia em 1997 e embutido no modelo 6110 em tela monocromática. Os modelos atuais são capazes de exibir centenas de milhares de cores e em alguns casos disporem de alguns Gigas de memória (MCLACHLAN, 2006),possibilitando a produção de jogos cada vez mais complexos, mas mesmo com a considerável evolução das capacidades dos aparelhos, os telefones celulares como plataformas para jogos ainda sofrem muitas restrições quando comparados aos consoles e computadores.

Figura 7: Tela do jogo Snake

A capacidade de armazenamento dos celulares, em sua maioria, ainda está limitada a Kilobytes, enquanto computadores e consoles apresentam, centenas de megabytes de memória RAM e outras possibilidades de armazenamento permanente da ordem de Gigabytes de capacidade, além disso, consoles e computadores apresentam processadores de alta capacidade (NASCIMENTO, 2003), numa tendência geral a apresentar processadores dedicados para cada mídia de interação, como por exemplo, placas de vídeo, de som com processamento e memórias próprios (SPOHN , 2006), os celulares por outro lado, têm capacidade muito limitada de processamento, restrita ao processamento mono-tarefa (NASCIMENTO, 2003).

Outro problema relacionado ao desenvolvimento para telefones celulares é a questão da portabilidade, os fabricantes lançam aparelhos com características muito diferentes voltados para alcançar públicos diferenciados, desta forma ocorre uma variação muito grande

dos recursos disponíveis em cada aparelho, mesmo para aplicações desenvolvidas e executadas em plataformas comuns em diversos aparelhos (LAU, 2004), o que muitas vezes inviabiliza a garantia de que o jogo irá funcionar corretamente em determinado aparelho (FERNANDES; DIAS, 2006) .

3.1 - PLATAFORMAS DE DESENVOLVIMENTO

As diferenças entre os aparelhos, até mesmos entre aqueles que eram produzidos pelos mesmos fabricantes dificultavam o desenvolvimento de aplicativos para essa plataforma, deste modo algumas soluções foram propostas no sentido de fornecer aos desenvolvedores condições de atuar em tal mercado sem ter de se preocupar com as minúcias de cada aparelho.

As soluções, plataformas de desenvolvimento, visam acrescentar uma camada de software adicional ao aparelho de forma a ocultar as diferenças entre os mesmos, para isso as plataformas oferecem ambientes de execução para aplicativos desenvolvidos, uma ou mais linguagens de programação para o desenvolvimento e bibliotecas de apoio. Os proprietários das plataformas também disponibilizam ferramentas para os desenvolvedores, e em alguns casos até mesmo programas para a publicação dos aplicativos desenvolvidos (NOGUEIRA; LOUREIRO FILHO; ALMEIDA, 2005).

O sucesso de um aplicativo depende bastante da plataforma escolhida, a análise dos pontos fortes e fracos de cada plataforma comparados com os pontos preferenciais do projeto é fundamental (AMARO, 2003),

As seções seguintes apresentam as características gerais das principais plataformas de desenvolvimento disponíveis.

3.1.1 – BREW MP

A plataforma BREW, *Binary Runtime Environment* for Wireless (Ambiente binário de execução para dispositivos móveis), foi lançada pela Qualcomm em 2001 e oferecia ambiente de execução e download com assinatura digital (NOGUEIRA; LOUREIRO FILHO; ALMEIDA, 2005).Em 2010 a plataforma tornou-se um sistema operacional e passou a adotar o nome BrewMP, *Mobile Platform*(KENT, 2010).

O BREW original oferecia uma API que agregava diversos recursos úteis ao

desenvolvimento de jogos, como mecanismos de exibição de formas, gráficos vetoriais e Bitmaps, com recursos para animação de bitmaps (BARBAGALLO, 2002), (NOGUEIRA; LOUREIRO FILHO; ALMEIDA, 2005), o BREW MP é compatível com a plataforma antiga e oferece além das linguagens nativas C e C++, oferece suporte integrado para Flash e Java, além de apresentar novas e melhoradas API's (QUALCOMM, 2011)

O modelo de negócios da Qualcomm é baseado nas assinaturas digitais atribuídas relativas aos aplicativos desenvolvidos para a sua plataforma, além das assinaturas por download, cada aplicativo BREW recebe uma assinatura própria, desta forma a Qualcomm oferece diversos níveis de associação aos quais são atrelados diversos benefícios e valores cobrados com base no número de aplicativos que podem ser distribuídos, relativo às assinaturas compradas pelo desenvolvedor (BARBAGALLO, 2002).

A Qualcomm oferece dois níveis básicos de associação um gratuito e outro pago, em diversas modalidades, para associar-se é necessário um cadastro no site da BREW, tendo feito o cadastro gratuito o desenvolvedor tem acesso às ferramentas básicas para o desenvolvimento e um série de documentações, a associação paga garante ao associado a distribuição do conteúdo produzido, por meio do ambiente BREW, frente às operadoras e fabricantes, após testes que são feitos pela Qualcomm (QUALCOMM, 2007).

O custo de produção e os métodos de publicação da Qualcomm são benéficos para o consumidor que pode realizar transações de compra num ambiente seguro (NOGUEIRA; LOUREIRO FILHO; ALMEIDA, 2005), são também bastante interessantes para o desenvolvedores já estabelecidos no mercado, pois garantem a segurança do seu produto quanto à pirataria, no entanto é pouco atrativo para os pequenos e novos desenvolvedores (NOGUEIRA; LOUREIRO FILHO; ALMEIDA, 2005).

3.1.2 – JAVA ME

A plataforma Java ME, Java *Micro Edition*, é uma versão reduzida da plataforma Java, lançada em 1999 com o intuito de atender aos dispositivos com maiores limitações de poder de armazenagem e processamento (NOGUEIRA; FILHO; ALMEIDA, 2005) , já que com o crescimento da plataforma Java esta se tornou demasiadamente grande para os dispositivos para os quais teria sido originalmente desenvolvida (AMARO, 2003).

São definidos para a plataforma dois módulos: Configuração e Perfil. O módulo de configuração é composto de um conjunto de bibliotecas e uma máquina virtual com

funcionalidades comuns para um determinado conjunto de dispositivos com capacidades de processamento e memória semelhantes, são definidas duas configurações para Java ME, CLDC e CDC (SUN, 2005).

A configuração CLDC (*Connected Limited Device Configuration*), contempla os dispositivos com nas capacidades gráfica, de armazenamento e de processamento reduzidas, tais como os telefones celulares e *pagers*. A configuração inclui uma versão limitada da máquina virtual Java, que omite algumas das capacidades da máquina virtual Java padrão. A configuração CDC (*Connected device Configuration*) contempla dispositivos com maiores capacidades e com conexão constante de rede (SUN MICROSYSTEMS, 2007), como *Smartphones*, impressoras de rede e PDAs (*Personal Digital Assistant*). A configuração CDC objetiva oferecer uma máquina virtual Java adaptada e APIs que possam oferecer todas as capacidades da Máquina virtual padrão Java com um consumo mínimo de espaço, 2 megabytes de ROM e 2 Megabytes de RAM (SUN, 2005).

Os perfis definidos para as configurações Java 2 ME são conjuntos de APIs com o objetivo de alcançar subconjuntos de dispositivos dentro do conjunto de dispositivos contemplados pelas configurações da plataforma, como por exemplo, telefones celulares, dentro da configuração CLDC (NOGUEIRA; FILHO; ALMEIDA, 2005).

A configuração CDC dispõe de três perfis, *Foundation Profile, Personal Basis Profile* e *Personal Profile*. O *Foundation Profile*, JSR 219 oferece serviços básicos como entrada e saída de dados, serviços de rede sem inclusão de recursos de interface do usuário. O *Personal Basis Profile*, JSR 217, inclui os serviços do primeiro e acrescenta outros como a possibilidade de construção de estruturas básicas de interface do usuário. O *Personal Profile* agrega as funções dos perfis anteriores e oferece suporte a *applets* entre outros serviços (SUN, 2005).

A configuração CLDC oferece três perfis: MIDP – *Mobile In formation Device Profile* importantes, MIDP1.0 e MIDP 2.0 e IMP, O perfil MIDP 1.0 oferece APIs de controle do ciclo de vida de aplicação, conexão de rede, interface de usuário e armazenamento persistente, a versão 2.0 deste perfil inclui melhorias principalmente no que diz respeito à interface do usuário, além de oferecer, uma API voltada para jogos (KNUDSEN, 2002). O perfil IMP contempla dispositivos que não dispõe de exigências padrão do MIDP para interação, como teclado ou tela, o perfil oferece todos os outros serviços do MIDP 1.0, exceto os serviços de Interface do usuário (GIGUERE, 2004).

Os jogos voltados para telefones celulares desenvolvidos para a plataforma Java ME, utilizam a configuração CLDC (SUN, 2007). O suporte a jogos dado pela plataforma depende

do perfil utilizado, MIDP 1.0 ou 2.0, já que o IMP não oferece opções de interface do usuário (GIGUERE, 2004), a versão 1.0 não oferece suporte específico para a produção de jogos eletrônicos, e tem suporte limitado aos serviços de interface do usuário (AMARO, 2003). A versão MIDP 2.0 dispõe de uma API para jogos que inclui suporte direto a criação e manipulação de *layers, tiles e sprites* (KNUDSEN, 2002), reprodução de sons e manipulação de imagens entre outros (NOGUEIRA; FILHO; ALMEIDA, 2005).

Os jogos como todos os outros aplicativos desenvolvidos para a plataforma Java ME são executados na máquina virtual Java que roda sobre o sistema operacional do aparelho, esse fato implica no comprometimento do desempenho da aplicação, em contrapartida possibilita maior alcance de mercado da aplicação produzida já que para um mesmo perfil e configuração o funcionamento é praticamente garantido (NOGUEIRA; FILHO; ALMEIDA, 2005).

O custo de desenvolvimento para a plataforma é bastante baixo, já que as ferramentas para o desenvolvimento são totalmente gratuitas, disponíveis para sistemas de código aberto como o Linux (SUN, 2007), não é necessária a publicação por nenhum canal específico, e não há nenhum custo para a publicação relacionado à plataforma (NOGUEIRA; FILHO; ALMEIDA, 2005), o modelo gratuito é, no entanto desvantajoso na medida em que torna o produto suscetível à pirataria e também coloca a cargo do desenvolvedor a negociação para a publicação (AMARO, 2003).

A principal vantagem da plataforma Java é sua disseminação, grande parte dos telefones celulares dispõe da tecnologia Java o que a torna bastante atraente quanto ao número de potenciais consumidores (AMARO, 2003)

3.1.3 – EXEN

A ExEn, *Executivo Engine*, foi a primeira plataforma de desenvolvimento para celulares voltada diretamente para a produção de jogos (IN-FUSIO, 2002), foi desenvolvida pela In-Fusio, líder européia de jogos para dispositivos móveis (NOGUEIRA; FILHO; ALMEIDA, 2005).

A linguagem da plataforma é Java, que é executada numa máquina virtual própria que segundo a desenvolvedora pode ser até 30% mais rápida que uma máquina virtual genérica (AMARO, 2003). A ExEn oferece compatibilidade com a configuração CLDC do Java ME e as APIs de jogos dos perfis MIDP (IN-FUSIO, 2002), combinando a especificação CLDC do

Java com funções nativas do sistema operacional para acelerar o processamento gráfico da plataforma (NOGUEIRA; FILHO; ALMEIDA, 2005).

A plataforma ExEn oferece mecanismos para construção de jogos 2D como manipulação de Sprites com operações de *zoom, flipping e mirroring, Scroll Parallax* e possibilidade de áreas com transparência total; Inclui também mecanismos de manipulação de gráficos 3D, tais como, cálculos de matrizes tridimensionais, primitivas de renderização e *raycasting*. O download e execução dos jogos da plataforma é feita dentro do ambiente da máquina virtual ExEn o que protege o usuário de softwares maliciosos e travamentos por invasões de áreas indevidas de memória(IN-FUSIO, 2002).

O modelo de negócios da In-Fusio divide os desenvolvedores em dois níveis o nível Standard e o nível Premium. Os desenvolvedores do nível Standard, cadastram-se gratuitamente junto à In-Fusio e têm acesso ao kit de desenvolvimento gratuito, um emulador e suporte técnico online, além da possibilidade de poder ascender ao nível Premium, os desenvolvedores que alcançam o nível Premium têm seus jogos publicados pela In-Fusio junto às operadoras de telefonia móvel (AMARO, 2003), para atingir o nível Premium os desenvolvedores passam por um processo de seleção que envolve critérios de avaliação relacionados à experiência na área de jogos eletrônicos e de tecnologia móvel (NOGUEIRA; FILHO; ALMEIDA, 2005).

A plataforma da In-Fusio abrange sete países da Europa e a China, somando um total de 220 milhões de potenciais jogadores.

3.1.4 – MOPHUN

A plataforma Mophun foi desenvolvida pela synergenix e lançada no mercado em 2002 (AMARO, 2003). O objetivo da plataforma é aproximar o desenvolvimento de jogos para dispositivos móveis do desenvolvimento tradicional para consoles, oferecendo aos desenvolvedores interessados uma API centrada na produção desse tipo de aplicação, a fim de acelerar o processo de desenvolvimento (SYNERGENIX, 2004).

As linguagens nativas Mophun são C e C++, também havendo a possibilidade de utilizar a programação *assembler* (SYNERGENIX, 2004). A plataforma é composta por quatro partes: O módulo de gerência e execução, *Run Time Engine* (RTE), onde os jogos são executados em esquema de *sandbox*, a fim de não realizem ações maliciosas; A Mophun API (*Application Programming Interface*) que reúne um conjunto de funções suportadas pela

plataforma e disponibilizadas para o desenvolvedor; O kit de desenvolvimento distribuído gratuitamente que possibilita o uso das APIs da plataforma entre outras facilidades para o desenvolvedor e o Mophun VST que é uma ferramenta de distribuição do jogo produzido (NOGUEIRA; FILHO; ALMEIDA, 2005).

A API Mophun é otimizada para a produção de jogos e oferece motor para a manipulação de sprites e tiles, sistema de detecção de colisões, suporte a som e a conexões, além da possibilidade de acesso a itens de hardware como vibração do aparelho, luzes de fundo e telas sensíveis ao toque, a plataforma também oferece serviços voltados para a produção de jogos 3D, entre eles encontram-se cálculos tridimensionais, manipulação de materiais, iluminação, sombreamento, neblina, correção de perspectiva e a possibilidade de utilização de diferentes métodos de renderização (SYNERGENIX, 2004).

O modelo de negócios da Synergenix segue quatro passos básicos, o primeiro deles é a submissão da idéia do jogo para a empresa que irá analisar as possibilidades comerciais do jogo. O segundo passo é a submissão da versão final do jogo para a Synergenix, que irá analisar seu conteúdo e realizar testes, a fim de evitar a publicação de conteúdos ofensivos e certificar-se de que os jogos apresentam os padrões exigidos pela empresa, os jogos aprovados são certificados e recebem uma assinatura digital, seguindo então para os passos três e quatro, que são respectivamente publicação e distribuição (NOGUEIRA; FILHO; ALMEIDA, 2005).

A plataforma Mophun apresenta muitas facilidades para o desenvolvedor que diminuem consideravelmente o tempo de execução, além disso o desempenho da máquina Virtual Mophun é bastante satisfatório chegando a um índice de desempenho 150 vezes maior que o da máquina virtual Java ME, e a implementação de partes dessa máquina virtual em linguagem de máquina torna possível o aproveitamento de até 90% da capacidade total do dispositivo (AMARO, 2003).

A principal desvantagem da plataforma desenvolvida pela Synergenix é o número limitado de dispositivos que portam sua tecnologia (NOGUEIRA; LOUREIRO FILHO; ALMEIDA, 2005).

3.1.5 - ANDROID

Android é um conjunto de software para dispositivos móveis, que engloba sistema operacional, *midllewar*e aplicações chave que incluem clientes de e-mail, navegador, calendário, mapas e etc. É uma plataforma de código aberto onde os aplicativos de terceiros têm acesso às mesmas API's dos aplicativos nativos executando com a mesma prioridade. Os aplicativos Android são desenvolvidos na linguagem Java (ANDROID DEVELORES, 2011a).

A plataforma Android foi proposta em 2007 pela *Open Handset* Alliance, (ROGERS et al., 2009). A aliança envolve empresas ligadas a tecnologia e telefonia móvel, seu objetivo é fomentar a inovação e melhorar a experiência do usuário com as plataformas móveis (OPEN HEADSET ALLIANCE, 2011), para isso uma nova plataforma foi proposta, não proprietária e baseada em padrões de código aberto, a fim de maximizar os lucros e diminuir os custos de produção. O primeiro aparelho Android, T-mobile G1, foi lançado no final de 2008 (DARCEY; CONDER, 2010).

A arquitetura Android é organizada numa pilha, cuja base é o *kernel* Linux 2.6 que realiza serviços de núcleo de sistema, como gerenciamento de memória e de processos e segurança, a próxima camada é formada pelas bibliotecas do sistema e pelo tempo de execução do Android, as bibliotecas são um conjunto de bibliotecas C/C++ que fornecem funcionalidades como gravações de áudio em diversos formatos e bibliotecas gráficas baseadas no Open GL (ANDROID DEVELOPERS, 2011a).

O tempo de execução do Android inclui uma máquina virtual e bibliotecas para a linguagem Java. As bibliotecas Java inclusas não são as mesmas da linguagem padrão (STEELE; TO, 2011);A maquina virtual inclusa, Dalvik *Virtual Machine*, é otimizada para dispositivos móveis, de forma que um dispositivo possa executar diversas instâncias dela, cada aplicativo Android é executado em um processo próprio com uma instância própria da máquina virtual(ANDROID DEVELOPERS, 2011a).

A próxima camada da arquitetura Android é formada pelos Frameworks de aplicação, nela estão disponíveis para o desenvolvedor ,através de API's Java, todos os recursos definidos nas bibliotecas da camada anterior, é também possível para o desenvolvedor disponibilizar algumas das capacidades de seu aplicativo para que outros aplicativos possam utilizá-las. A última camada é formada pelos aplicativos (ANDROID DEVELORES, 2011a).

A plataforma não apresenta ferramentas diretamente voltadas para facilitar o desenvolvimento de jogos, mas oferece o NDK – *Native Development Kit*, kit de

desenvolvimento que permite ao desenvolvedor implementar funções críticas ou todo ciclo de vida de seus aplicativos diretamente em linguagem nativa: C/C++, permitindo acesso direto a funções como manipulação de comandos, áudio e manipulação de janelas, com bibliotecas baseadas no padrão OpenGL (ANDROID DEVELORES, 2011b).

As aplicações Android podem ser distribuídas em diversos sites de venda de aplicativos para dispositivos móveis, dada a natureza aberta da plataforma, podem até mesmo ser distribuídas pelo próprio desenvolvedor desde que esse possua um host onde o pacote da aplicação possa ser armazenado e baixado (DARCEY; CONDER, 2010). O principal canal de publicação de aplicativos da plataforma é o Android *Market*, serviço do Google onde se pode publicar e fazer downloads de aplicações(ROGERS et al., 2009).

A publicação de uma aplicação no Android *Market* requer que o desenvolvedor mantenha uma conta, para inscrever-se é necessário o pagamento de uma taxa única e através dessa conta é possível fazer *upload* de aplicativos para serem vendidos ou distribuídos gratuitamente , o desenvolvedor pode ainda escolher quando seus aplicativos deverão estar disponíveis para *download*. O Android Market cobra uma taxa de 30% a cada venda sobre os valor de venda do aplicativo (ANDROID MARKET, 2011).

3.1.6-FLASH

A plataforma Flash foi lançada pela Macromedia, empresa posteriormente anexada à Adobe, em 1996 e inicialmente foi concebida para a produção e execução de animações 2D em páginas da web (WALDRON, 2000). O crescimento da Internet e a necessidade de tornar as páginas mais interativas e atraentes para os usuários popularizaram o flash, que por sua vez passou a receber cada vez mais recursos que o fizeram evoluir de um único programa para uma importante plataforma para aplicações entregues via Internet (LABRIOLA; TAPPER; BOLES, 2010).

Além do Flash Player, que permite a execução de conteúdos e aplicativos flash a partir de um navegador de internet, a plataforma oferece dois outros tempos de execução: Adobe AIR e Flash Lite. O Adobe AIR permite a execução dos aplicativos sem a necessidade de navegador (ADOBE, 2011). O Flash Lite é um tempo de execução para dispositivos móveis, adaptado à baixa capacidade computacional desses dispositivos e a algumas de suas características típicas como telas pequenas e limitação de carga das baterias (ELROM; JANOUSEK; JOOS, 2009).

A Adobe oferece um série de produtos para a produção o de conteúdo flash, o que inclui tanto ferramentas pagas quando uma SDK gratuita (LABRIOLA; TAPPER; BOLES, 2010), além disso há várias ferramentas gratuitas mantidas por comunidades disponíveis pela Internet (ALLEN et al, 2008) e algumas apoiadas pela própria Adobe (ADOBE OPEN SOURCE, 2011) .

A plataforma não apresenta bibliotecas nativas com recursos específicos para jogos e não há recursos para melhoria do desempenho. Mesmo que o funcionamento possa ser garantido em diversas plataformas, uma medida mais precisa dos recursos mínimos para o jogo, e de sua performance demanda testes diretos nas plataformas escolhidas (ELROM, JANOUSEK, 2009; GRIFFITH, 2010).

A principal vantagem da plataforma flash é sua penetração, Mais de 90% dos usuários de internet tem uma versão do Flash Player instalado em navegadores que operam em todos os principais sistemas operacionais atualmente em uso (GRIFFITH, 2010). Outra vantagem da plataforma é sua flexibilidade, tendo um tempo de execução Flash é possível desenvolver não só aplicativos e jogos, mas também banners, páginas web, apresentações e animações sem utilização de linhas de código (GRIFFITH, 2010).

O desenvolvedor interessado nos telefones celulares pode, no entanto, ter que lidar com diversas versões do tempo de execução flash lite, pois nem todos os celulares são capazes de executar, o Flash player 10 e o adobe AIR, além disso, a substituição desses aparelhos costuma ser mais demorada, quando comparada a dos computadores, o que acrescenta uma maior sobrevida às versões. Diferentes versões do Flash lite oferecem diferentes recursos que dependem da versão do tempo de execução, fabricantes e modelos dos aparelhos (ANDERSON, 2010; ELROM, JANOUSEK, 2009).

A publicação dos aplicativos flash fica a cargo do desenvolvedor, não há encargos devidos à Adobe que não oferece serviço de distribuição. Jogos para os dispositivos que executam as versões completas dos tempos de execução podem ser diretamente distribuídos pelo próprio desenvolvedor, através de downloads serem jogados em páginas da web. A distribuição para as plataformas móveis depende dos mercados fechados de distribuição não só do fabricante como também das empresas de telefonia. (RIVELLO, 2011).

CAPÍTULO 4

ESTUDO DE CASO

Neste capítulo é apresentado como estudo de caso o demo do jogo “Eddy”, desenvolvido na plataforma Java ME, essa plataforma foi escolhida por atender aos quatro quesitos desejados para o projeto: A existência de suporte ao desenvolvimento de jogos; Acesso as ferramentas de desenvolvimento; Linguagem utilizada e popularidade, pois o número de aparelhos que dispõem da tecnologia, afeta diretamente os possíveis aparelhos para a realização de testes.

Ferramentas de desenvolvimento para Java ME são disponibilizadas gratuitamente para download, a plataforma está presente em um grande número de aparelhos e ainda é a mais popular entre os dispositivos (AMARO, 2003), além disso, a documentação e materiais de apoio sobre a plataforma estão disponíveis de forma gratuita na página da mesma.

A linguagem utilizada na plataforma é muito semelhante à Linguagem Java padrão, diferenciando-se basicamente pelas API’s e limitações. Como a linguagem Java já é conhecida pelo desenvolvedor, o aprendizado consistiu basicamente em aprender a utilizar essas API’s.

Finalmente um dos perfis da plataforma apresenta facilidades bastante interessantes para o desenvolvimento de jogos, utilizando o MIDP 2.0 o desenvolvedor pode contar com gerenciamento de camadas gráficas, incluindo suporte a tiles; Manipulação de Sprites e detecção de colisões (WELLS, 2004).

4.1 APRESENTAÇÃO

Eddy é um jogo de ação do tipo plataforma com elementos de tiro. Apresenta gráficos 2D em progressão lateral, nele o jogador controla o robô Eddy que deve atravessar as paisagens em busca de um artefato escondido em cada uma delas. O *gameplay* se concentra na travessia dos cenários e enfrentamento de inimigos.

4.2– PRINCIPAIS DEFINIÇÕES DO GAME DESIGN

Essa seção apresenta os pontos principais do Game design, maiores detalhes podem ser encontrados no documento de Game design no Apêndice A.

Eddy é um jogo para telefones celulares que roda sobre a plataforma Java Me, CLDC 1.0 e perfil MIDP 2.0 e com resolução de tela igual a 320x240 pixels, a quantidade de memória não foi definida para a versão demo.

A versão demo do jogo é formada pelo primeiro cenário do jogo, constituído pelos dois primeiros níveis do jogo.

Devem estar presentes pelo menos duas telas de apoio para a versão demo, a primeira tela é a do menu principal e deverá oferecer ao usuário as opções de começar um novo jogo e de sair da aplicação, a segunda tela, menu de pausa deve oferecer as opções de retornar ao jogo ou voltar ao menu principal.

O personagem principal Eddy é controlado pelo jogador, um robô que teve de ser ativado antes que sua construção estivesse completa a fim de alcançar os artefatos antes do inimigo, logo apenas algumas de suas capacidades estão disponíveis no inicio do jogo, portanto o jogador deverá buscar os itens que ativem os poderes do personagem. Na versão demo as ações disponíveis para o jogador são: Correr, saltar, agachar e atirar. O jogador contará com três pontos de vida indicados no lado esquerdo da tela, caso perca todos esses pontos perderá uma vida. A figura 8 mostra uma pose do personagem principal.

Figura 8: Eddy, Personagem principal do jogo

O primeiro nível do jogo ocorre no subsolo de um pântano de águas ácidas, a paisagem é pouco variada e formada por fungos, rochas e poças de ácido, as armadilhas de ácido são fatais para o personagem, o objetivo principal desse nível será subir até a superfície

na direção do segundo nível, onde se encontra o item buscado. O segundo nível acontece na superfície do mesmo pântano, o nível se passa sobre as plataformas suspensas acima do lago de ácido, e pequenas cavernas distribuídas pela região, além do solo rochoso, aqui os fungos são mais coloridos e podem alcançar diversos tamanhos, parecendo-se com árvores, há também pequenas plantas. O objetivo é atravessar a paisagem destruindo os inimigos até chegar ao ponto onde o artefato está.

Os inimigos do jogo (Figura 9) têm comportamento reativo ou repetitivo. Os inimigos repetitivos, SpykeBall e MechaBee executam seus movimentos independentemente das ações do jogador parando somente quando destruídos pelo mesmo. O inimigo reativo, FastPunch executa ações padrão até que uma determinada ação do jogador seja detectada, a partir de então muda seu comportamento, para um novo conjunto de ações.

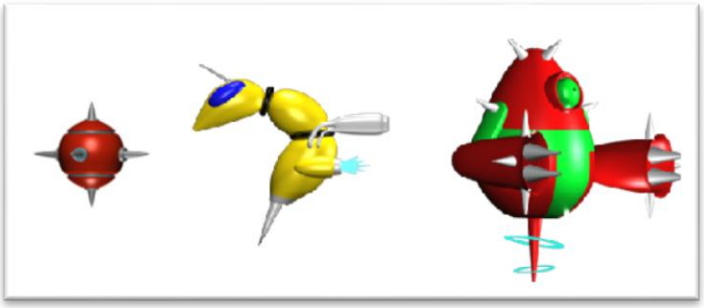


Figura 9: Inimigos SpykeBall, MechaBee e FastPunch respectivamente

O jogador conta com os itens *Checkpoint e Cog* (Figura 10). Os itens *Cog* constituem itens de bônus, cada objeto *Com* tem um valor aleatório, chamado *Cog points*, a cada cem *Cog points* coletados o jogador recebe uma vida extra. O Checkpoint marca um ponto para o qual o jogador pode voltar depois que perde uma vida, servindo como gravação temporária do avanço do jogador no nível.

Figura 10: Itens Checkpoint e Cog respectivamente

A tela deve ser apresentar os seguintes mostradores: *Health Meter*, *Lives panel* e *Cogs panel*. O jogador terá o total de pontos de vida que poderá utilizar, indicado no *Health meter*, cada dano implicará na perda de um ponto, a perda de todos os pontos implicará na perda de uma vida. O indicador *Lives panel* mostrará quantas vidas ainda restam ao jogador, caso o jogador morra e o indicador vida marque zero, ele sofrerá *Game Over*. O indicador *Cogs panel* irá indicar quantos *Cog Points* o jogador possui, quando esse valor superar 99, ele será novamente zerado e uma nova vida será acrescida no *Lives panel*, os mostradores são exibidos na Figura 9.

Figura 11: Mostradores do jogo

Eddy deve utilizar gráficos 2D criados a partir da renderização de modelos tridimensionais, seu aspecto gráfico deverá lembrar animações 3D sem necessidade de fotorrealismo e com iluminação simples, sem a preocupação com a definição de sombras.

4.3-ARTE GRÁFICA

Toda a arte gráfica produzida para Eddy foi modelada em um criador de gráficos 3D, e renderizado em imagens 2D, para criação dos Sprites ou Tiles e então reunidos em arquivos de Imagem.

A criação dos Modelos 3D e arquivos de imagens forma feitos utilizando o software Blender (BLENDER, 2011) que permite a criação de modelos 3D e sua posterior renderização em arquivos de imagem PNG, onde cada arquivo é equivalente a um dos quadros da animação. Um dos itens de cenário na tela de edição do Blender é mostrado na figura 12.

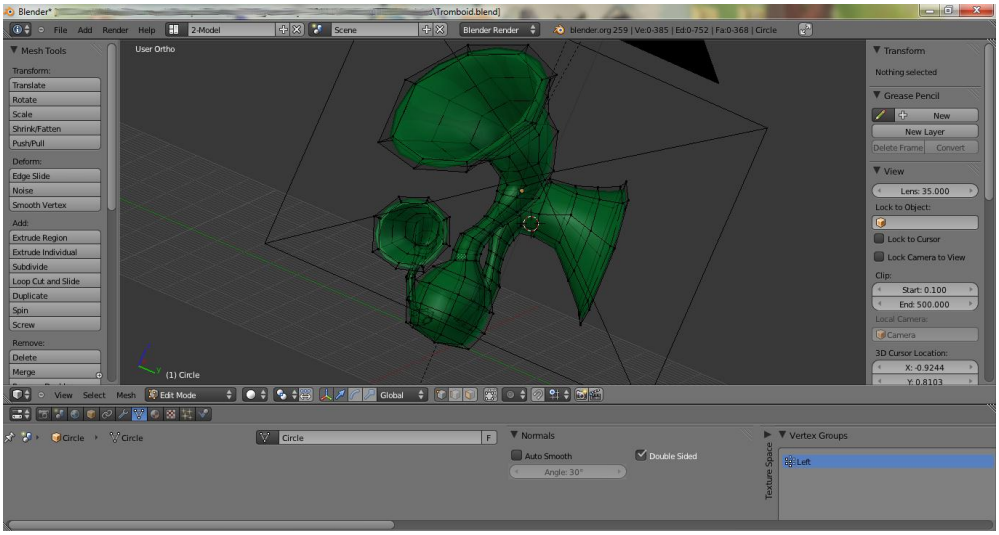


Figura 12: Item do jogo na tela de edição do Blender

O Java Me utiliza para cada objeto do tipo *Sprite* ou *Tile* um arquivo no formato PNG, como fonte das imagens, que devem estar enfileiradas, não necessariamente na ordem de utilização (HAMMER, 2007), dessa forma fez-se necessária a utilização de um editor de imagens, não só para reunir os quadros num arquivo, mas também para converter o formato final, o editor utilizado foi o GIMP (GIMP, 2011). A figura 13 mostra o arquivo de sprites com alguns dos itens e inimigos dos dois primeiros níveis.

Figura 13: Arquivo de sprites incluindo itens e inimigos do jogo

O arquivo criado pode ser diretamente referenciado pelo Java ME mencionando-se apenas a posição do Sprite no arquivo, tomando-o como uma fila única de imagens de mesmo tamanho. A criação de cada animação foi feita criando-se *arrays* com as posições dos Sprites, relativos a animação desejada, no arquivos e controlando sua duração e repetição.

A criação dos cenários exige um passo adicional que equivale à produção do mapa para a sua montagem, para isso uma ferramenta visual foi utilizada, o aplicativo Mappy

(MAPPY EDITOR, 2011). O cenário é criado visualmente através do arrastar dos tiles para a posição desejada e em seguida convertido em um *array* de inteiros que é utilizado para recriar o cenário no jogo. Uma tela do Editor Mappy é mostrada na Figura 14.

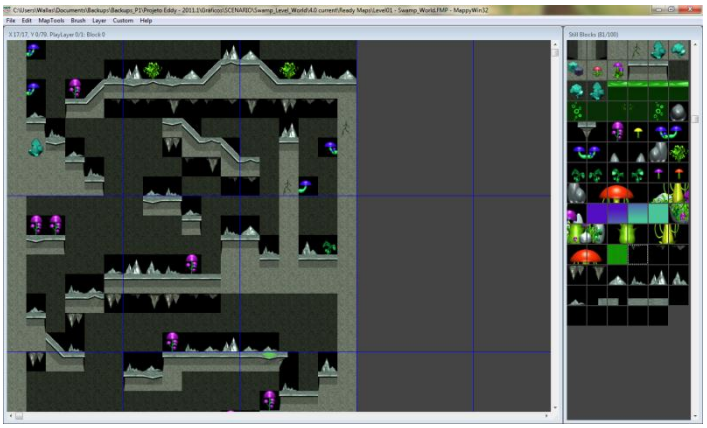


Figura 14: Tela do editor de tiles Mappy

4.4 –IMPLEMENTAÇÃO

A implementação de aplicativos Java Me inicia-se com a definição de um MIDlet, Todo aplicativo Java ME deve definir um MIDlet, através da extensão da classe de mesmo nome, ela é usada para controlar o ciclo de vida do aplicativo, parada, Início, Pausa e também fornece acesso ao display permitindo controlar o que é mostrado nele, a classe MIDlet possui métodos padrão, que devem ser incluídos no código e que podem ser sobrescritos incluindo operações adicionais como salvar antes de encerrar a aplicação (HAMER, 2007). A classe MIDlet definida para o projeto é a classe EddyMID, (Figura 21).

A tela do jogo: EddyCanvas (Figura 15) foi definida a partir da derivação da classe GameCanvas, que permite o desenho e a alocação de elementos gráficos na tela em baixo nível com bufferização automática e, detecção de entradas no teclado do dispositivo (HAMMER, 2007).

Figura 15: Diagramada classe EddyMid

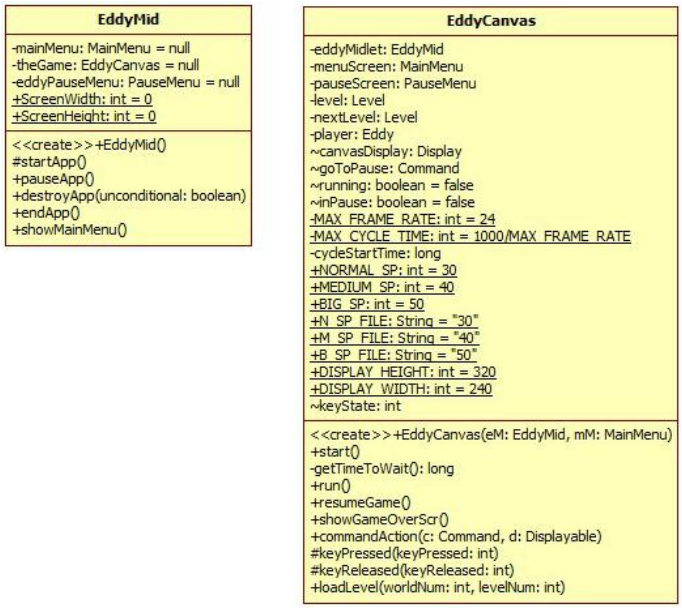


Figura 16: Diagramas das classes EddyMid (à esquerda) e EddyCanvas (à direita)

A classe EddyCanvas é definida como *runnable* o que permite a definição de um thread separado para a execução do jogo isso possibilita não só a interrupção do jogo sem a interrupção do aplicativo, tornando possíveis operações como pausar o jogo, salvá-lo ou carregar outros níveis, mas também permite controlar a velocidade da execução do jogo, de forma que os elementos da tela de jogo pareçam ser sempre atualizados numa mesma velocidade, mantendo a animação sempre suave e agradável.

Para o controle da velocidade do jogo foi inserindo um período de espera após cada atualização da tela do jogo, durante a execução do thread do jogo, caso toda a animação seja terminada antes do tempo determinado, um tempo de espera é calculado, com base no tempo usado para atualizar a tela e na velocidade esperada para os frames.

4.4.1– MENUS

Todos os menus do aplicativo são exibidos em tela inteira, cada um deles é também definido em sua própria classe. As classes utilizadas para os menus derivam a classe Form presente no Java ME que apresenta diversas funcionalidades de seleção de opções na tela, tais como tabelas de opções, listas de opções e caixas de seleção.

A primeira tela exibida quando o MIDlet do jogo Eddy se inicia é o Menu Principal (figura 17), sua função é guiar o usuário para o jogo ou para fora da aplicação, esse menu pode ser acessado a partir de todos os demais menus, por isso foi o primeiro a ser implementado. O Menu principal na versão demo apresenta em sua tela o logo provisório do jogo e as opções: “Start demo” e “Quit”, a primeira opção inicia o jogo no nível definido para a versão demo e a segunda encerra a aplicação.

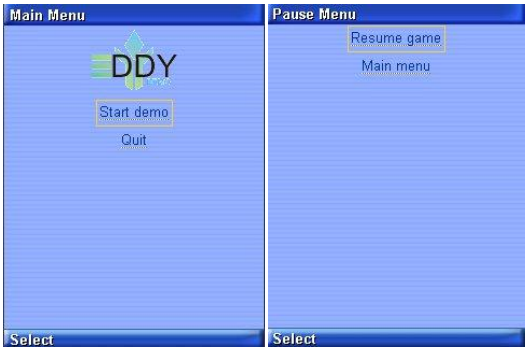


Figura 17: Menus do jogo

O Segundo Menu presente nessa versão do jogo é o Menu de pausa. Ele é acionado a partir da tela do jogo, através do comando Pausar e permite interromper o jogo temporariamente, mantendo o estado atual do jogo, na memória. Ao acionar o comando Pausar o MIDlet interrompe o thread do jogo e exibe uma nova tela com as opções de retornar ao jogo: “Resume game” ou de retornar ao Menu principal.

4.4.2 –SPRITES E MOSTRADORES

A classe base para os itens e personagens animados do jogo foi a classe EddySprite (Figura 18). Ela deriva a classe Sprite do Java Me, adicionando à essa as funcionalidades de direção e verificação de colisões com zona transparente.

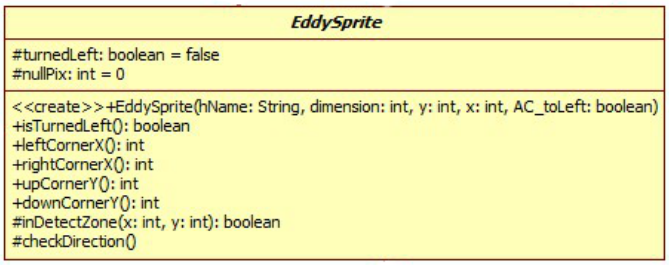


Figura 18: Diagrama da classe EddySprite

Cada EddySprite tem um valor booleano, turnedLef que indica se o Sprite está ou não voltado para a esquerda, caso seja verdadeiro, o método checkDirection transforma as imagens relativas ao Sprite para que sua parte frontal esteja voltada para a direção correta.

A detecção de colisões do Java Me mostrou-se inadequada ao propósito do jogo, como alguns atores não ocupam todo o espaço do Sprite, a colisão simples, detectava colisões em que os personagens estavam relativamente distantes, por outro lado a detecção com verificação de transparência de pixels mostrou-se muito lenta o por diversas vezes os atores já se encontravam entrelaçados quando a colisão era detectada.

A solução encontrada foi atribuir a cada ator uma zona de pixels considerados transparentes que é ignorada quando da detecção, reduzindo a parte opaca dos sprites, passível de sofrer colisões, para uma área dentro deles. A construção da zona de transparência para a detecção de colisões é feita com base no valor inteiro nullPix que define o número de pixels a partir dos vértices reais do Sprite que são ignorados quando os métodos que retornam a posição dos seus vértices são invocados.

A verificação de colisões é feita utilizando os métodos de localização dos vértices pela verificação de intersecção das áreas opacas dos sprites envolvidos.

O primeiro personagem implementado foi o personagem principal: Eddy. Um dos principais pontos da classe de mesmo nome (Figura 25) é o acréscimo de uma máquina de estados que podem ser alterados por fatores do ambiente (nível) e pelos comandos do jogador.

O controle do jogador utiliza os métodos de controle presentes na classe Canvas padrão no MIDP 1.0, que apesar de apresentar a desvantagem de detectar somente um botão pressionado por vez, mostrou-se mais adequado às ações do jogo tais como correr. No esquema utilizado, basta que o jogador mantenha pressionado o botão na direção desejada. Utilizando o esquema padrão GameCanvas era necessário pressionar o botão várias vezes, ou utilizando a repetida detecção do botão a velocidade do personagem aumentava.

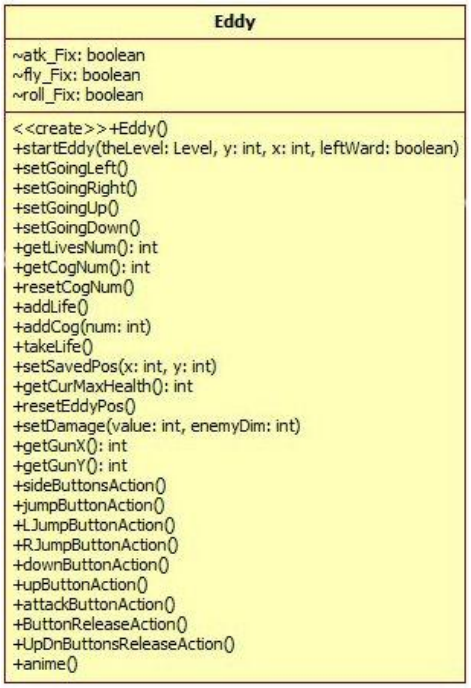


Figura 19: Diagrama da classe Eddy

A classe do personagem principal também acumula a verificação de gravidade sobre o personagem, apesar de a gravidade afetar outros atores do jogo, a análise diferenciada foi incluída em Eddy para permitir flexibilidade diante de alguns aspectos restritos à animação desse personagem como subir e descer planos inclinados, e cair sobre *tiles* especiais do cenário, como armadilhas e passagens secretas.

A implementação dos inimigos partiu da criação da classe Enemy (Figura 26), que derivada da classe EddySprite. Cada inimigo apresenta valores de *gameplay* como vitalidade, pontos de ataque e defesa e um valor booleano, *active* para indicar se o inimigo está ativo ou não. O valor *active* é utilizado para facilitar a detecção de colisões, evitando a verificação de inimigos que já foram abatidos e possibilitando a reutilização dos inimigos.

A classe Enemy não inclui verificações de interação entre o inimigo e o cenário, dessa forma, para verificações como gravidade uma nova classe dela derivada foi definida, a classe EnemyGround, os inimigos derivados dessa classe, SpykeBallGround e fastPunch são capazes de se mover sobre o chão e de verificar se existem paredes à sua frente, podendo alternar para estado de queda (FALL_ST) ou padrão (STD_ST), quando está sobre o chão. A classe Enemy e a classe EnemyGround são mostradas na figura 20.

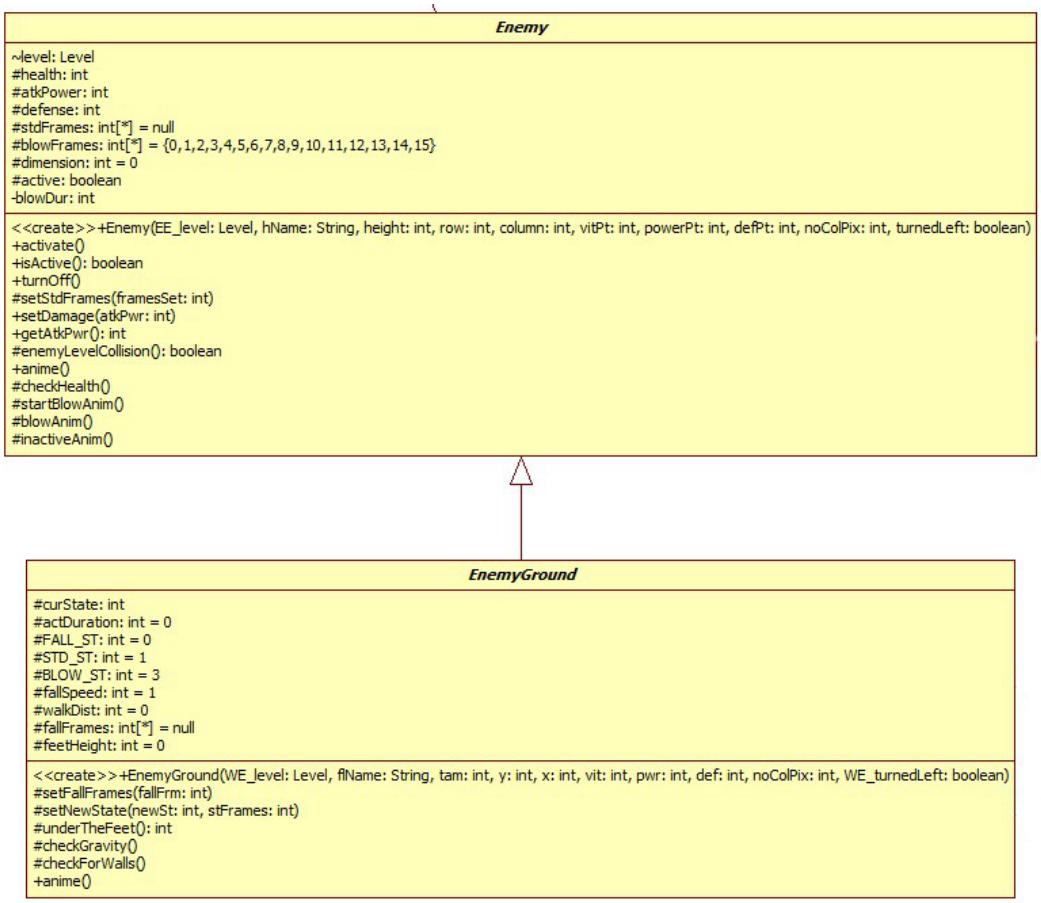


Figura 20: A classe Enemy e a classe EnemyGround

Os itens do jogo, Cogs, Checkpoints e EnergyShots (figura 21) são também derivados da classe EddySprite e devido às suas diferentes naturezas e comportamentos eles foram diretamente derivados dela, não sendo, portanto agrupados numa classe comum de itens, a exemplo do que foi feito com os inimigos.

A primeira classe de itens e também primeiro Sprite animado acrescentado ao jogo foi o item Cog. Cada *cog* executa sua animação padrão de rotação, sem influência da gravidade, até que tenha contato com Eddy, a partir daí o nível o comanda a tornar-se invisível; o marca como *used* e envia a Eddy a mensagem para que este incremente o número desses itens que já foram coletados, na classe Eddy por sua vez um método de verificação é acionado sempre que um Cog é coletado, a fim de incrementar o número de vidas sempre que o número deles atingir 100.

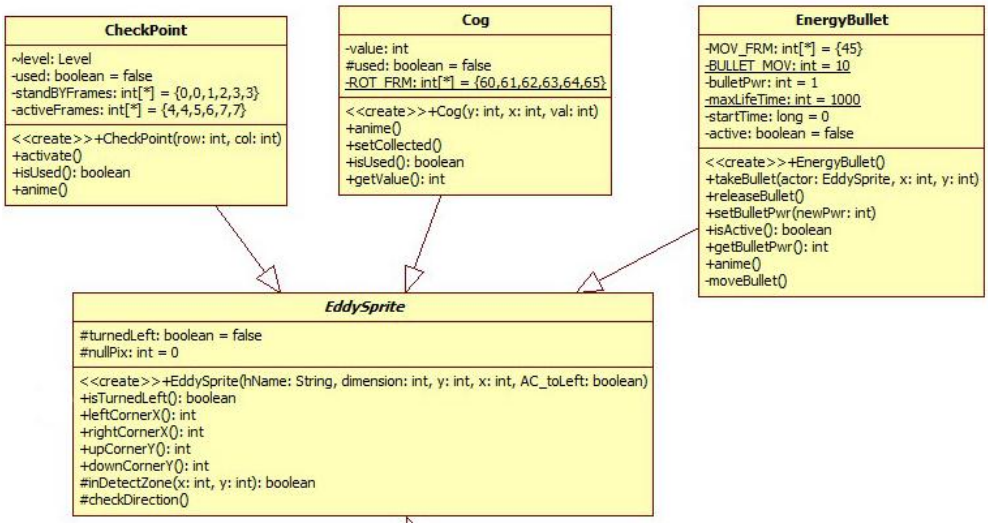


Figura 21 : Diagrama com a derivação que origina os itens do jogo

Os objetos da classe Cog apresentam um campo *used* que informa se o item já foi coletado e um campo *value* de modo a permitir que um objeto Cog possa representar diferentes números de itens Cog, ou seja um objeto coletado pode representar um ou mais objetos acrescentados ao contador do jogador. O intuito dessa implementação é permitir um número grande de itens cog, com um número pequeno de objetos na tela, o que economiza recursos, além de acrescentar surpresa ao jogo, já que o jogador não saberá exatamente quantos itens irá coletar, sem que tenha jogado o nível anteriormente.

O checkpoint marca pontos de retorno dentro do nível, nenhum dado é salvo nesse objeto. Os dados para retorno, posição do jogador e posição da câmera, são guardados em variáveis presentes nos objetos das classes Eddy e Level. Após detectar a colisão entre Eddy e um checkpoint, o Level guarda a posição atual da câmera e envia a Eddy a mensagem para armazenar como posição de retorno a posição do checkpoint.

O item checkpoint apresenta duas sequências de animação, a animação de espera e a animação de ativado, A animação de espera é executada até que Eddy entre em contato com um desses, objetos, após o contato, o checkpoint é definido como usado (*used*) e a animação de ativado é acionada.

Os projéteis de tiros do jogo são representados por objetos da classe EnergyBullet, cada projétil é um objeto dessa classe, também derivada de EddySprite. Cada projétil tem um marcador de atividade, *active*, que indica se o projétil está em uso e um valor de tempo de vida máximo que define o tempo durante o qual o projétil ficará ativo, além desses há também os valores de velocidade (espaço a percorrer) e poder, pontos removidos da vida do personagem atingido.

Os projéteis não estão armazenados nos personagens e sim no nível, quando um personagem atira ele solicita um objeto *EnergyBullet* à classe Level, através do método showShot. A cada solicitação, o nível verifica se existe um projétil livre e seu *array* de projéteis e em caso positivo associa ao projétil os dados necessários vindos do jogador: Ponto de saída e direção do tiro e exibe o tiro na tela.

Os mostradores do jogo foram implementados por semelhança de comportamento, o mostrador de vitalidade do jogador, que tem comportamento único, foi implementado numa classe própria com um Array de Sprites com o mesmo e único quadro de animação. Os Sprites tornam-se visíveis de acordo com o número de pontos de vida que o jogador possui. Os dados sobre os pontos de vida são armazenados em Eddy e são disponibilizados ao HealthMeter através da classeLevel, pelo método getPlayerHealth(), a cada dano o nível também indica ao mostrador que deve apagar uma de suas barras.

O mostrador *Health meter* posiciona-se no canto superior esquerdo da tela e possui algumas variáveis e uma constante cuja função principal é manter sua posição e aparência na tela mesmo quando a câmera se move. Os *sprites* que compõe mostrador são enfileirados a uma distância fixa determinada pela constante BAR_SPACE, a posição da primeira barra é determinada pelas variáveis anchorX e anchorY, cuja posição é relativa às coordenadas da câmera, por essa razão o método anime dessa classe recebe dois parâmetros, que são as coordenadas da câmera. A posição das demais barras é calculada com auxilio da variável newX e da constante de espaço entre as barras.

Os outros dois mostradores do jogo *Life panel* e *cog panel* são contadores, sua função é mostrar na tela o número de vidas e *cós*, respectivamente que o jogador possui, cada contador possui um ícone que indica o item contado em seguida o número deles. O comportamento desses mostradores é muito semelhante: Coletar dados de um item específico com Eddy e exibir na tela. A diferença básica entre esses mostradores é o item a ser contado e consequentemente o ícone a ser exibido, diante disso esses mostradores foram definidos tendo como base a mesma classe, a classe GamePanel.

A classe GamePanel é uma classe abstrata que define os elementos necessários para exibir na tela um contador de itens oriundos da classe Eddy, e seu respectivo ícone. A classe utiliza um objeto Sprite como ícone e dois objetos da classe IconNumber para representar a quantidade, além dos campos relativos ao posicionamento na tela como os apresentados na classe HealthMeter.

A classe IconNumber, define um tipo derivado da classe Sprite, que exibe na tela a imagem de um número de um único digito. Cada Objeto da classe IconNumber possui um

valor numérico associado que corresponde ao valor a ser exibido. Um objeto dessa classe exibe o mesmo frame até que uma solicitação seja feita através do método setValue( ). As figuras 22 e 23 mostram os diagramas das classes utilizadas na implementação dos mostradores.

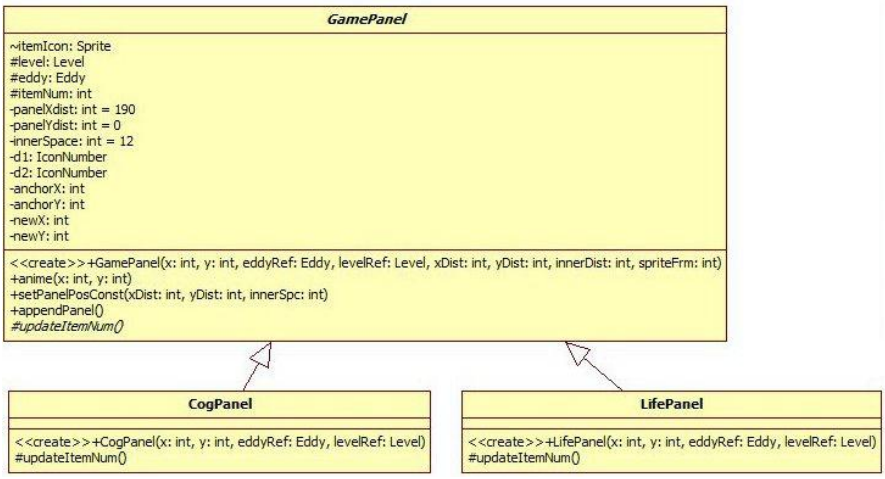


Figura 22 : Diagrama das classes que definem os mostradores

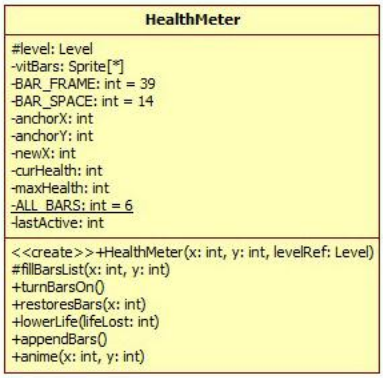


Figura 23: Diagrama das classes que definem os mostradores

4.4.3 – NÍVEIS

A classe Level foi criada pra definir os níveis do jogo e deriva a classe LayerManager, incluída no perfil MIDP 2.0, que apresenta importantes ferramentas para a criação de jogos como a criação e gerenciamento de camadas gráficas bidimensionais, incluindo a inserção de *sprites* e camadas compostas por tiles.

Cada objeto Level deve definir um cenário composto por pelo menos duas camadas, uma camada de fundo, backLayer, decorativa que não é detectada pelos *sprites* e uma camada de interação, actionLayer, onde são verificadas todas as interações com o cenário como colisões com paredes e gravidade. A figura 24 mostra o diagrama da classe Level.

Figura 24: Diagrama da classe Level

As camadas constituintes do nível são definidas na classe EddyTiledLayer (figura 25) que deriva a classe TiledLayer, também presente na API de jogos. A classe do Java ME implementa uma camada de tiles a partir de um arquivo fonte de imagens, seguindo instruções de quais tiles ocupam quais células da camada, através do método setCell( ). A classe EddyTiledLayer especializa a classe original definindo camadas com tiles de dimensões constantes, construídos a partir de um *array* unidimensional, exatamente como é gerado pelo aplicativo gerador de mapas.

Os tiles que constituem o cenário são divididos pelos níveis em tipos. A classificação é feita utilizando as posições dos tiles no arquivo de imagem e permite a criação de áreas especiais no cenário sem a utilização de camadas adicionais, como armadilhas e elementos decorativos. Ao detectar colisões com algum *tile* da camada de interação, o Sprite solicita ao nível qual o tipo desse tile e responde da forma adequada.

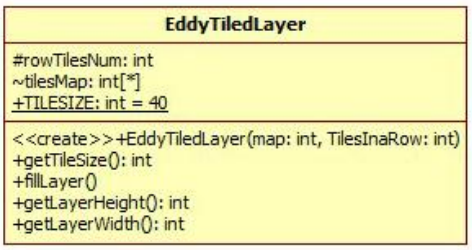


Figura 25:Diagrama da classe EddyTiledLayer

Os níveis do jogo controlam não somente a construção do cenário, mas também sua correta exibição. A área exibida do nível mantém o jogador sempre visível e no centro da tela, exceto quando o fim do cenário é atingido. A área de visualização que pode ser movimentada é implementada no código como câmera e é definida com base no método setViewWindow que é nativo da classe LayerManager, e permite definir qual parte da área gráfica do LayerManager estará visível (HAMMER, 2007).

O método setViewWindow define a área de visualização através da delimitação de um retângulo construído a partir de seu vértice superior esquerdo, num ponto definido, e se estendendo até as largura e altura desejadas (HAMMER, 2007).A câmera do jogo foi definida como uma área de visualização de dimensões iguais a da tela e com origem baseada em duas variáveis cameraX e cameraY que podem ser convenientemente modificadas de forma a proporcionar a movimentação da área visível sobre cenário.

A movimentação da câmera é feita de forma combinada, entre as classes Level e Eddy, a cada movimentação de Eddy uma chamada ao método movCam( ) é feita a fim alterar a posição da câmera, a classe Level por sua vez verifica se o limite do cenário já foi atingido e em caso contrário movimenta a câmera.

Além dos cenários a classe Level também é responsável pelo gerenciamento dos *sprites*. Cada nível é constituído por *arrays* de itens, inimigos e projéteis, além de Eddy, do checkpoint e dos mostradores. O nível é responsável por ativar e desativar inimigos e itens e pelas interações entre os *sprites*, como exibição de tiros quando requisitado e verificação e tratamento de colisões.

As detecções de colisões entre os *sprites* são invocadas e tratadas pelo nível e são realizadas por quatro métodos diferentes, segundo os tipos de *sprites* nelas envolvidos, checkBulletsCollision(), que verifica se algum projétil colidiu com algum inimigo ou com Eddy, checkPlayerCogsCollision(), verifica se o jogador colidiu com algum cog; checkPlayerEnemiesCollision() que verifica colisões entre o personagem do jogador e os inimigos ativos do nível e checkPlayercPointColl() que verifica se o jogador entrou em

contato com o checkpoint do nível, colisões entre inimigos e entre inimigos e itens não são verificadas.

As colisões são verificadas através do método inDetectZone(), presente na classe EddySprite, esse método verifica se um ponto dado está na área opaca do Sprite, a função de verificação do nível consulta o método inDetectZone de um dos sprites, para cada vértice do outro Sprite.

A checagem dos arrays de inimigos, projéteis e itens é feita apenas para os elementos ativados, ou seja, para os inimigos e projéteis são verificadas as colisões apenas para os sprites cujo valor active seja verdadeiro e para os *cogs* são apenas verificados os itens cujo valor *used* seja falso. A checagem de colisões é o primeiro passo executado pelo nível a cada volta no laço do jogo.

As colisões entre sprites e cenários são verificadas e tratadas pelos sprites, já que apenas alguns deles é sensível a esse tipo de colisão, além disso, algumas mudanças no cenário podem acarretar mudanças nos seus estados. O papel do nível é informar ao sprite qual tipo de tile se encontra em um ou mais pontos definidos pelo próprio sprite (método tileTypeInPoint),baseado nessa informação o sprite definirá seu curso de ação podendo alterar ou não seu estado.

A verificação de colisões com o cenário é principalmente usada para a simulação de gravidade e detecção de paredes e obstáculos e é executada pelo personagem sempre que o seu método de animação é invocado.

4.5–INTELIGÊNCIA ARTIFICIAL

A aplicação de comportamento independente no jogo limitou-se aos inimigos, em sua maioria com comportamentos repetitivos, com exceção do inimigo FastPunch que apresenta detecção de distância do usuário.

Todas as instruções de comportamento automático foram inseridas no código da função de animação dos personagens, para todos os inimigos da versão demo.

O comportamento de detecção do inimigo FastPunch foi simulado através da alteração da função de animação para uma máquina de estados que altera seu comportamento a partir da detecção da presença do jogador numa distância horizontal.

4.6 – SONS

Os sons do jogo deverão ser executados em dois modos: Efeitos sonoros e o modo trilha sonora, o primeiro executará sons relativos às ações do jogador tais como sons de tiro e explosões, no modo trilha sonora a música relativa a cada fase será executada durante o jogo, por questões de limitação de hardware os modos não poderão ser ativados simultaneamente, e devendo ser escolhidos pelo usuário antes de cada partida ou no menu pausa. A versão demo do jogo não contará com sons.

4.7-TESTES

Foram executados testes de unidade, testes de integração e de Versão, para os quais não foram definidos formulários, sua execução consistiu somente na execução exaustiva do jogo.

Os testes de unidade ocorreram ao fim de cada funcionalidade acrescentada ao jogo, como por exemplo, um novo movimento da personagem. O objetivo do teste de unidade é a verificação da consistência da nova funcionalidade com o proposto pelo Game design. Uma vez verificada tal funcionalidade, eram realizados os testes de integração cuja função é verificar se nenhum comportamento indesejável foi acrescentado junto com a nova funcionalidade. Após o acréscimo ou exclusão de funcionalidades diretamente relacionadas ou de funcionalidades importantes definia-se uma nova versão a ser completamente testada.

Os testes de versão aconteceram em dois diferentes momentos, primeiramente um teste completo com o jogo em execução nos emuladores do kit de desenvolvimento, em seguida o teste era repetido com o jogo em execução em um telefone celular. A figura 26 mostra telas do jogo em execução no emulador do kit de desenvolvimento.

Figura 26: Jogo em Execução no Emulador do kit de desenvolvimento Java ME

CAPITULO 5

CONSIDERAÇÕES FINAIS

Apesar de um jogo para telefone celular ser um projeto de dimensões consideravelmente menores e das facilidades apresentadas pela plataforma escolhida, algumas dificuldades foram marcantes no processo de desenvolvimento, dentre elas, cita-se a falta de aplicações que englobem todos os pontos da produção, especialmente no que tange aos objetos artísticos utilizados.

A diversidade de mídias envolvidas em um game faz com que um projeto de jogo envolva a utilização de diversas ferramentas, não só para a manipulação de diferentes tipos de mídia como sons e gráficos, mas também na manipulação das diversas etapas necessárias a um mesmo tipo de objeto de mídia. A principal implicação dessa gama de aplicações para o projeto foi o aumento do tempo inicialmente planejado. A produção realizada por um único desenvolvedor precisa ser pausada a cada nova etapa para a realização de treinamento numa nova ferramenta.

A quantidade de tempo gasto no domínio das ferramentas utilizadas e na própria produção das mídias e a constante e necessária busca de soluções criativas, mostraram que um menor tempo de produção seria mais facilmente alcançado se o projeto tivesse sido desenvolvido em equipe.

A utilização da plataforma para desenvolvimento móvel reduz as preocupações com as minúcias existentes entre os diversos aparelhos, também conta com estruturas úteis para a criação de jogos, incluindo simuladores dos aparelhos que facilitam a realização de testes. No entanto, a abstração do nível de hardware limita o acesso os recursos dos mesmos, além de impedir algumas melhorias, como o uso de memória, condicionando o desenvolvedor aos padrões da plataforma que nem sempre são os melhores para as especificações ou o desempenho do jogo.

O impacto mais significativo dessa desvantagem para o projeto apresentou-se na detecção das entradas do teclado do aparelho a fim de manter a consistência da interação. Foi necessário ao jogo combinar as tecnologias gráficas da versão MIDP 2 ao método de detecção de entradas da versão anterior, obrigando o usuário a dispor da interação de apenas um botão por vez, e a aplicação a utilizar mais botões em substituição às ações inicialmente concebidas

para execução por combinação de botões, tais como salto direcionado.

Outro ponto importante observado é que a manutenção das boas práticas de codificação muitas vezes esbarra nas limitações do hardware ou nas especificações do jogo, o que levou não só à eliminação de algumas características inicialmente definidas para o jogo, (inicialmente um das ações disponíveis para Eddy era escalar paredes), mas também na construção de algumas classes com código muito similar ou muito grande. As classes de código similar surgiram para evitar o desperdício de memória com funções não comuns ou variáveis não úteis para elementos de natureza bastante parecida, como por exemplo, os inimigos. As classes grandes, como a classe do personagem principal, foram preferidas de forma a evitar o máximo a comunicação entre classes, o que poderia comprometer o desempenho.

O aspecto mais interessante e atraente na criação de um jogo é a possibilidade e constante necessidade de integração de criatividade ao processo de desenvolvimento de um aplicativo de software, essa integração não só é necessária nos elementos de mídia da aplicação, como gráficos, sons e enredo do jogo, mas também se fazem necessárias aplicações criativas de conceitos acadêmicos e práticas de desenvolvimento. Dessa forma, a produção de um jogo pode ser incluída em projetos pedagógicos como um meio excelente de aplicação dos conteúdos de computação através de projetos pedagógicos que possibilitem a apresentação ou prática de conteúdos diversos da área.

Diante das possibilidades dos games, a área é ainda bastante modesta no país e faltam materiais de estudo em língua portuguesa: As publicações oriundas dos ambientes acadêmicos, na sua grande maioria, não abordam aspectos práticos do desenvolvimento e há poucos livros sobre eles. Os principais pontos de apoio aos desenvolvedores nacionais são as comunidades de desenvolvedores de jogos, que apresentam tutoriais e fóruns de discussão.

A pouca tradição nacional na produção de games faz com que os investimentos nessa área sejam insuficientes para permitir que os produtores nacionais possam competir com as superproduções dos grandes estúdios internacionais. Os dispositivos móveis podem representar uma ótima opção para novos desenvolvedores que dispõe de poucos recursos ou de pouca experiência, já que, além apresentar custos de produção bastante reduzidos, condiciona todos os desenvolvedores às mesmas limitações, colocando-os em igualdade competitiva.

As plataformas de desenvolvimento para dispositivos móveis são ótimas opções para o estudo do desenvolvimento dos jogos, ou a utilização da produção de jogos com fins educacionais. Como muitas delas oferecem recursos específicos para o desenvolvimento de

jogos, o desenvolvedor pode focar-se nos aspectos do jogo, deixando os detalhes técnicos da plataforma e diminuindo o tempo de produção.

A escolha da plataforma do projeto não englobou medições ou comparações detalhadas entre as plataformas, dessa forma o trabalho poderia ser estendido a um estudo comparativo do desenvolvimento entre as plataformas disponíveis, verificando qual o real impacto das diferenças entre elas para um projeto de jogo, tais como tempo de desenvolvimento e desempenho.

Durante a produção dos mapas de *tiles* dos níveis do jogo foram encontrados, nas diversas ferramentas de construção, recursos para a exportação dos mapas e geração de código para diversas linguagens e bibliotecas para jogos de computador. Nenhum delas, no entanto, contempla as plataformas móveis, desse modo o desenvolvimento de uma ferramenta que traduza os mapas de *tiles* em códigos utilizáveis em plataformas para dispositivos móveis seria uma importante contribuição para o desenvolvimentos de jogos móveis.

REFERÊNCIAS:

ABRAGAMES. Associação Brasileira das Desenvolvedoras de Jogos Eletrônicos – Estatuto social. 2004. Disponível em: < http://www.abragames.com.br/docs/estatuto_abragames.pdf >. Acesso em: Ago. 2007.[PDF]

ABRAGAMES. A indústria de desenvolvimento de jogos eletrônicos no Brasil. 2005. Disponível em: <http://www.abragames.com.br/docs/PesquisaAbragames.pdf>. Acesso em: Ago. 2007. [PDF]

ABREU, André de. Videogame: Um bem ou um mal? Um breve panorama da influência dos jogos eletrônicos na cultura individual e coletiva. São Paulo, 2003. Disponível em: <http://andredeabreu.com.br/docs/videogames_bem_ou_mal.pdf>. Acesso em Dez. 2007. [PDF]

ADOBE. Introducing the Adobe Flash platform. 2011.Disponível em: <http://www.adobe.com/devnet/flashplatform/articles/flashplatform_overview.html>. Acesso em: Set. 2011

AFP. Celular inteligente pode substituir o computador. 2007. Disponível em: <http://tecnologia.terra.com.br/interna/0,,OI2052508-EI4796,00.html>. Acesso em: Jan.2008

AGOT, Kevin. Astraware's Board games Review. Palm Addicts. Henderson, Out .2007.Disponível em: <http://palmaddict.typepad.com/palmaddicts/2007/10/astrawares-boar.html>. Acesso em: Out.2007

ALECRIM, Emerson. Formato OGG Vorbis. InfoWester. Sine loco, Maio. 2005. Disponível em: <http://www.infowester.com/oggvorbis.php>. Acesso em: Set. 2007

ALLEN et al. Open source Flash development. New York: Friendsof, 2008. 382p.

AMARO, Pedro. The Clash of Mobile Platforms: J2ME, ExEn, Mophun and WGE. GameDev.net. Coimbra, Maio. 2003. Disponível em: < http://www.gamedev.net/reference/articles/article1944.asp>. Acesso em: Jul.2007

ANDROID DEVELOPERS. The developer's Guide .2011. Disponível em: <http://developer.android.com/guide/index.html>. Acesso em: Jan. 2011

ANDROID DEVELOPERS.SDK - Native Development Tools.2011.Disponível em: <http://developer.android.com/sdk/ndk/index.html>. Acesso em: Jan. 2011

ANDROID MARKET. Android Market Help. 2011. Disponível em: <http://market.android.com/support/bin/answer.py?hl=en&answer=112622>. Acesso em: Jan. 2011

ARANHA, 2004 - ARANHA, Gláucio (2004). A reconfiguração do gesto de leitura e leitor nos textos narrativos mediados pela tecnologia dos Jogos Eletrônicos. Ciências & Cognição; Ano 1, Vol 02. Disponível em: <www.cienciasecognicao.org>

ARAUJO, Marilu. Novos jogos brasileiros no celular. 2005 Disponível em:< http://www.link.estadao.com.br/index.cfm?id_conteudo=4566>. Acesso em: Out. 2007

ARTHUR, Kim Daniel. An introduction to developing for mobile devices Using J2ME/MIDP. Sine loco. Disponível em: < http://www.gamedev.net/reference/programming/features/J2ME1/>. Acesso em: Set. 2007

AZEVEDO, Eduardo (Cord). Desenvolvimento de jogos 3D e Aplicações em Realidade Virtual. Rio de Janeiro: Campus, 2005,319p.

AZEVEDO, Théo. impostos encarecem preço dos videogames no Brasil. UOL Jogos. Sine loco. Dez. 2006. Disponível em: <http://jogos.uol.com.br/reportagens/ultnot/ult2240u116.jhtm>. Acesso em: Ago. 2007

BARBAGALLO, Ralph. Games on the Run: BREW and J2ME. 2002. Disponível em: <http://www.gamasutra.com/resource_guide/20021125/barbagallo_02.htm>. Acesso em: Set. 2007

BATTAIOLA, André Luiz. et al. Desenvolvimento de jogos em computadores e celulares. Revista de Informática Teórica e Aplicada. Porto Alegre: UFRGS v.8, n.2, Out. 2001 (Home Page do Instituto de Informática).

BLENDER. Blender Homepage. 2011. Disponível em: <http://www.blender.org>. Acesso em: Out. 2011

MAPPY EDITOR. www.tilemap.co.ukHomePage. Disponível em: <http://tilemap.co.uk/mappy.php>. Acesso em: Out.2011

BREW MP. Say Hello to the new Brew.2010. <http://www.brewmp.com/why-brew-mp/say-hello-new-brew>. Acesso em: Jan. 2011

BREYER, Felipe B. et al. Avaliação de usabilidade no processo de desenvolvimento de jogos: Definição de métodos de acompanhamento de qualidade para Game Design. In: Brazilian Symposium on Computer Games and Digital Entertainment, 5. 2006, Recife. Artigo. Recife: Porto digital, 2006. p. 2-5.

CARNOY, David. Xbox 360 and PS3: Death to PC gaming? Junho. 2005. Disponível em: <http://reviews.cnet.com/4520-6449_7-6233821-1.html>. Acesso em: Jul. 2007

CLUA, Esteban Walter Gonzalez; BITTENCOURT, João Ricardo. Desenvolvimento de jogos 3D: Concepção, Design e Programação. In: Congresso da Sociedade Brasileira de Computação, 25. São Leopoldo: UNISINOS, 2005. Anais... v. 1, p.1313-1357. Disponível em: <http://www.unisinos.br/_diversos/congresso/sbc2005/_dados/anais/pdf/arq0286.pdf>. Acesso em: Ago. 2007. [PDF]

DARCEY, Lauren; CONDER, Shane. Teach yourself Android applications in 24 hours. United States of America, 2010. 458p

DAVIS, Michele E.; PHILLIPS, Jon A. Flex 3: A beginner's guide .USA: McGraw-Hill, 2008. 274p. [PDF]

DE FALCO, Alessandra. Desenvolvimento de jogos para celular no Brasil. Webinsider. Sine loco, Abr. 2005. Disponível em: <http://webinsider.uol.com.br/index.php/2005/04/29/desenvolvimento-de-jogos-para-celular-no-brasil/>. Acesso em: Out. 2007

DE FALCO, Alessandra. Advergames divulgam marca Tilibra. 2007. Disponível em: <http://www.gamecultura.com.br/index.php?option=com_content&task=view&id=368&Itemid=9>. Acesso em: Out. 2007

EIDOS MOBILE UK. Tomb Raider: Legend. 2006. Disponível em: <http://www.eidos.co.uk/mobile/trl/index.html>. Acesso em: Out. 2007

ELECTRONIC ARTS. EA Academy: How EA Makes Games, 2005. Disponível em: <http://www.jobs.ea.com/how_ea_makes_games.html>. Acesso em: Ago. 2007

ELROM, Elad; JANOUSEK, Scott; JOOS, Thomas. Advanced Flash on devices. California: Friendsof, 2009.713p.

FERNANDES, Tiago; DIAS, Alexandra. Processo de testes para desenvolvimento de jogos para celulares. In: Brazilian Symposium on Computer Games and Digital Entertainment, 5., 2006, Recife. Artigo. Recife: Porto digital, 2006. p.1-2.

FOLHA ONLINE. TecToy investe no mercado de jogos para celular. Folha Online. Sine loco, Jul. 2006. Disponível em: <http://www1.folha.uol.com.br/folha/informatica/ult124u20299.shtml>. Acesso em: Nov. 2007

FUSCO, Camila. Pirataria de software cai no Brasil mas prejuízo supera US$ 1 bi. Computer world. Sine loco. Maio.2007. Disponível em: http://computerworld.uol.com.br/mercado/2007/05/15/idgnoticia.2007-05-15.5103056083. Acesso em: Ago. 2007.

GIBSON, Bruce. Global Mobile Games Market to grow from revenues of $3bn in 2006 to over $17bn in 2011. Juniper research. Hampshire, June.2006. Disponível em: <http://www.juniperresearch.com/shop/viewpressrelease.php?pr=2>. Acesso em: Nov. 2007

GIGUERE, Eric. The Information Module Profile.2004.Disponível em: <http://developers.sun.com/mobility/imp/impintro/>. Acesso em: Out.2007

HAMER, Carol. Creating Mobile Games: Using Java Me Platform to put the fun into your mobile device and cell phone. California: Apress, 2007.416p.

GIMP. GIMP Homepage. 2011. Disponível em: <http://www.gimp.org/>. Acesso em: Out. 2011

GRIFFITH, Chris. Real-world flash game development. MASSACHUSSETS: Focal Press, 2010. 368p [PDF]

IN-FUSIO. ExEn Developer Zone. 2002. Disponível em: <http://developer.In-Fusio.com/exen/information_technology.php#what>. Acesso em: Out. 2007

KENT, Jack. Qualcomm launches Brew MP mobile OS. Screendigest. Sine loco, Jan. 2010. Disponível em: <http://www.screendigest.com/news/mi-14-01-2010-JK-1/view.html>. Acesso em: Set. 2011

KNUDSEN, Jonathan. What's new in MIDP 2.0.2002.Disponível em: <http://developers.sun.com/mobility/MIDP/articles/MIDP20/#game>.Acesso em: Out. 2007

LACERDA, Ana Paula. Internet e celular abrem porta para desenvolvedores de jogos. O Estado de São Paulo. São Paulo, Fev. 2006. Disponível em: http://txt.estado.com.br/editorias/2006/02/07/eco34598.xml. Acesso em: Ago. 2007

LAIRD, J. E., VAN LENT, M. The role of AI in Computer Game Genres. Disponível em: <http://ai.eecs.umich.edu/people/laird/papers/book-chapter.htm>. Acesso em: Set. 2007

LARAMEE, François Dominic. The game Design Process.1999. disponível em: <http://www.gamedev.net/reference/articles/article273.asp>. Acesso em: Ago. 2007

LAU, Allen. Porting Mobile Games: Overcoming the hurdles. Sine loco, Aug. 2004. Disponível em: < http://www.gamedev.net/reference/articles/article2134.asp>. Acesso em: Set. 2007

LEMOS, André. Cibercultura e Mobilidade: a Era da Conexão. México: Razón y Palabra nº 41, outubro/ novembro 2004.

LI, Kenneth. MTV Networks investing $500M in games. USA TODAY, Sine loco, Ago.2007.Disponível em: <http://www.usatoday.com/tech/gaming/2007-08-16-mtv-games-investment_N.htm> Acesso em: Out.2007

LIUKKONEN, Tapio. Computer Game: Problems of game audio. Joensuu. North Karelia University of Applied Sciences. Diplom. (Degree's Thesis): 80,2006. [PDF]

LONGO, Vinicius. RPG online brasileiro Erinia passa a ter acesso gratuito. 2006. Disponível em: <http://ig.forumpcs.com.br/noticia.php?b=190238>. Acesso em: Jul. 2007.

LUZ, Mairlo Hideyoshi Guibo Carneiro da – Desenvolvimento de jogos de computador. Itajubá, 2004. Monografia apresentada ao Departamento de Matemática e Computação da Universidade Federal de Itajubá.

MACIEL, Rui. PS3 x Xbox 360 x Wii: Entenda as diferenças. Wnews. Sine loco. Outubro. 2006.Disponível em: <http://wnews.uol.com.br/site/noticias/materia_especial.php?id_secao=17&id_conteudo=308>. Acesso em: Jul.2007

MCLACHLAN, Dean. Analysis: History of cell-phone gaming. 2006. Disponível em: <http://www.next-gen.biz/index2.php?option=com_content&task=view&id=2090&Itemid=2&pop=1&page=0>. Acesso em: Set. 2007

MICROSOFT. Manual do usuário.Xbox.com Brasil. 2010. Disponível em: <http://support.xbox.com/pt-br/pages/xbox-360/get-started/manuals-specs.aspx#section1>. Acesso em: Jan. 2011. [PDF]

MOITA, Filomena Ma. G. da S. Cordeiro Relações de gênero nos games: os estabelecidos e os outsiders. In Anais do 8º Simpósio Processo Civilizador, História e Educação, em João Pessoa-PB, de 16 a 17 de setembro de 2004.

MORAIS, Márcio de. Superada marca de 100 milhões de celulares em operação. Acessoria de Impressa da ANATEL, Brasília, Fev. 2007. [PDF]

MORRIS, Chris. Can Sony and Microsoft replicate Wii's success?.CNBC.com. Sine loco. Julho 2010.Disponível em: <http://www.cnbc.com/id/38309709/Can_Sony_and_Microsoft_Replicate_Wii_s_Success>. Acesso em: Jan. 2011

NACKE, Lennart. Facilitating the Education of Game Development. Department of Simulation and Graphics. Magdeburg, Otto-von-Guericke University. Diplom (Master’s Thesis): 153, 2005.[PDF]

NASCIMENTO, Ivo Frazão. Desenvolvimento de um Framework para jogos sobre a plataforma BREW. Centro de Informática. Universidade Federal de Pernambuco. Monografia (Graduação): 41, 2003. [PDF]

NINTENDO. Mode d'emploi de la wii chaînes et paramètres.support.nintendo.com. 2010. Disponível em: <http://www.nintendo.com/consumer/manuals/index.jsp>. Acesso em: Jan. 2011.[PDF]

NÍVEA. NIVEA Games. Disponível em: http://www.nivea.com.br/games. Acesso em: Out. 2007

NÓBREGA, Marcelo. 2007 é o ano da nova geração de consoles. Maio. 2005. Disponível em: <http://www.forumpcs.com.br/coluna.php?b=112903>. Acesso em: Jul. 2007.

NOGUEIRA, Wallace Franco de Azevedo ; LOUREIRO FILHO, Emerson Cavalcante ; ALMEIDA, Hyggo Oliveira de . Plataformas para Desenvolvimento de Jogos para Celulares . Infocomp Revista de Ciência da Computação, v. 4, n. 1, p. 53-61, 2005.

OVERPLAY. Over play advergame. 2007. Disponível em: <http://overplay.com.br/advportfolio/cellular/>. Acesso em: Out. 2007

PAGEL, Geovanna. Empresa brasileira desenvolve e exporta entretenimento eletrônico. 2005. Disponível em: <http://www.anba.com.br/pequena.php?id=13>. Acesso em: Out. 2007

P&G BRASIL. Hotsite de divulgação do Sabão em pó Ariel. Disponível em: <http://www.descubraariel.com.br/site/>. Acesso em: Out. 2007

PEREIRA JUNIOR, José Jarbas. O poder de envolvimento dos advergames. Webinsider. Sine loco, Fev.2003. Disponível em: <http://webinsider.uol.com.br/index.php/2003/02/11/o-poder-de-envolvimento-dos-advergames/>. Acesso em: Out. 2007

PINHEIRO, Cristiano Max Pereira. A história da utilização dos games como mídia. In: 4º Encontro Nacional da Rede Alfredo de Carvalho, São Luís, 2006. Disponível em: <>. Acesso em: Jan. 2008.

QUALCOMM. Qualcomm BREW - About BREW. 2007. Disponível em: <http://BREW.qualcomm.com/BREW/en/about/about_BREW.html>. Acesso em: Set. 2007

QUALCOMM. Brew MP – Platform. 2011. Disponível em: <http://www.brewmp.com/platform/platform >. Acesso em: Set. 2011

RAHAL, Fernando de Castro - Desenvolvimento de jogos eletrônicos. São Paulo, 2006. Trabalho de conclusão de curso apresentado ao Curso de engenharia de Computação do Centro Universitário Assunção, UNIFAI.

RAMOS, Daniela Karine. Jogos eletrônicos e a construção do juízo moral, das regras e dos valores sociais. In: II Seminário Jogos eletrônicos, Educação e Comunicação - construindo novas trilhas, 2006, Salvador. Anais II Seminário Jogos eletrônicos, Educação e Comunicação - construindo novas trilhas, 2006. v. 2. p. 1-9.

REUTERS. Celular é a plataforma do futuro para os games.2005. Disponível em: <http://tecnologia.terra.com.br/interna/0,,OI537280-EI4796,00.html>. Acesso em: Set. 2007

RIVELLO. Getting started with Flash Platform game development.2011.Disponível em: <http://www.adobe.com/devnet/games/articles/getting-started-flash-games.html#articlecontentAdobe_numberedheader>. Acesso em: Set. 2011

RODRIGUES,G.D.. Interatividade e virtualização nos jogos eletrônicos. In: CONGRESSO BRASILEIRO DE CIÊNCIAS DA COMUNICAÇÃO, 28., 2005. Rio de Janeiro. Anais... São Paulo: Intercom, 2005. CD-ROM.

RODRIGUES, Lia Carrari; LOPES, Rodrigo A.S. Pereira; MUSTARO, Pollyana Notar giacomo. Impactos sócio-culturais da evolução dos jogos eletrônicos e ferramentas comunicacionais: Um estudo sobre o desenvolvimento de comunidades virtuais de jogadores. In: Brazilian Symposium on Computer Games and Digital Entertainment, 6.2007, São Leopoldo. Artigo. São Leopoldo: UNISINOS, 2007. p. 1-8.

ROGERS, Rick; LOMBARDO, John; MEDNIEKS, Zigurd; MEIKE, Blaze. Android Application development. California: O´Reilly, 2009. 320p.

ROSENZWEIG, Gary. ActionScript 3.0 game programming university. Indiana: QUE, 2008. 455p. [PDF]

SANTANA, Roberto Tengan de. I.A. Em jogos: A busca competitiva entre o homem e a máquina. Faculdade de Tecnologia de Praia Grande Praia Grande.Praia Grande. Monografia (Graduação): 162, 2006. [PDF]

SANTEE, André. Programação de jogos com C++ e DirectX. São Paulo: Novatec Editora, 2005. 399p.

SCHNOOR, Tatiana. Telecom e celular: número de usuários de celular no mundo chegará a 4,6 bilhões até o final de 2012, diz Pyramid. WNews. São Paulo, Julho. 2007. Disponível em: <http://wnews.uol.com.br/site/noticias/materia.php?id_secao=4&id_conteudo=8507>. Acesso em: 06/08/2007.

SEAN. Burger King to offer Branded cell phone Games. Franchise Pick. Sine loco, Oct. 2007. Disponível em: <http://www.franchisepick.com/burger-king-to-offer-branded-cell-phone-games/>. Acesso em: Out. 2007

SEGA MOBILE. Sonic the Hedgehog.2007.Disponível em: <http://www.segamobile.com/gamespage.php?GameID=60>. Acesso em: Out. 2007 (Home Page Sega Mobile)

SCE - Sony Computer Entertainment. Playstation 3 Safety and support. Playstation page.2010. Disponível em: <http://www.playstation.com/manual/pdf/CECH-2001A_2001B-2.85_2.pdf>. Acesso em: Jan.2011. [PDF]

SOUZA FILHO, Fernando.PS3 é lançado no Brasil a R$ 1.999. MSN jogos. Sine loco. Agosto 2010. disponível em: <http://jogos.br.msn.com/noticias/artigo.aspx?cp-documentid=25162735>. Acesso em: Jan. 2011

SPOHN, Dave. PC vs. Console: Hardware for online gaming. 2010. Disponível em: <http://internetgames.about.com/od/hardware/a/pcvsconsole_2.htm>. Acesso em: Jan. 2011

SUN MICROSYSTEMS.CDC: Java Platform Technology for connected devices.2005.Disponível em: < http://java.sun.com/products/CDC/wp/CDC-whitepaper.pdf>. Acesso em: Out. 2007. [PDF]

STÄHELIN, Maycon. Délirus ganha o mundo com jogos para celular. NetMarinha. Florianópolis. Nov. 2006. Disponível em: <http://netmarinha.uol.com.br/noticias.asp?Id=10660>. Acesso em: Out. 2007

STEELE, James; TO, Nelson. The Android developer's cookbook. Boston: Addison-Wesley, 2011. 339p.

SUN MICROSYSTEMS. Java ME Technology.2007.Disponível em: <http://java.sun.com/javame/technology/index.jsp#CDC>. Acesso em: Out. 2007

SYNERGENIX. Mophun. 2004. Disponível em: <http://www.mophun.com/developer/index.php>. Acesso em: Out. 2007

SZALAI, Georg. PwC: Global Gaming Market to Approach $50 Billion by 2011. Gamedailybiz. Sine loco, June. 2007. Disponível em: <http://biz.gamedaily.com/industry/feature/?id=16589>. Acesso em: Jul. 2007.

TEC TOY MOBILE. Sobre a TecToy Mobile. Disponível em: www.ttmobile.com.br. Acesso em: Nov.2007

TORRES, Gabriel. Funcionamento do mp3, 2000. Disponível em: http://www.clubedohardware.com.br/artigos/433. Acesso em: Set. 2007

UOL JOGOS. Coletiva anuncia lançamento do Xbox 360 no Brasil; saiba como foi o evento. 2006. Disponível em: <http://jogos.uol.com.br/ultnot/xbox/ult4101u70.jhtm>. Acesso em: Jul. 2007

VIEIRA FILHO, Vicente. Revolution ai engine - desenvolvimento de um motor de inteligência artificial para a criação de jogos eletrônicos. Faculdade de ciência da computação - Dissertação (graduação) Universidade Federal de Pernambuco. Recife, 2005.

WALDRON, Rick. The flash history. Flash magazine.com. Sine loco. Novembro, 2000. Disponível em: <http://www.flashmagazine.com/news/detail/the_flash_history/>. Acesso em: Set. 2011

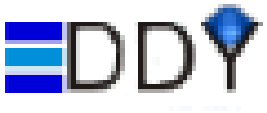
WEBINSIDER. Game brasileiro para celular é destaque na Ásia. Webinsider. Sine loco. Setembro. 2007. Disponível em: <http://webinsider.uol.com.br/index.php/2003/01/28/game-brasileiro-para-celular-e-destaque-na-asia/>. Acesso em: Out. 2007

WELLS, Martin J..J2ME game programming. Massachussets: Premier press, 2004. 768p.

WHARTON SCHOOL. Pirataria e impostos desafiam promissor mercado de games no Brasil. Universiaknowledge@Wharton. Sine loco, Julho. 2004. Disponível em: <http://www.wharton.universia.net/>. Acesso em: Jun. 2007

WIKIPEDIA. History of video game consoles (seventh generation).2011. Disponível em: <http://en.wikipedia.org/wiki/History_of_video_game_consoles_%28seventh_generation%29>. Acesso em: Jan.2011

APÊNDICE


DDY

Game Design

Mecânica do jogo

Gameplay

Jogo de plataforma 2D com visão lateral e elementos de tiro;

Fluxo geral

O jogador percorre cenários que são divididos em três níveis, duas fases de ação e uma onde deve enfrentar um inimigo de maior nível: O chefe do cenário.

Os níveis de ação são constituídos por solo e plataformas que o jogador deve alcançar para isso deverá transpor obstáculos como buracos, precipícios e altura, os obstáculos que podem ser transpostos pelo jogador são definidos pelos itens que já possui: Inicialmente constará com o movimento de pulo simples com o qual poderá alcançar plataformas baixas e transpor pequenos buracos.

Eddy também irá se deparar com locais de difícil acesso que só poderá transpor depois de coletar os *fixes* durante o jogo.

Em cada cenário Eddy deverá buscar pelo Crystal de Oscilum além de combater os robôs enviados pelo Dr.Gênesis para destruí-lo.

Fluxo por partida

O jogador terá uma barra de energia que terá seus indicadores reduzidos a cada dano, quando sua barra de energia estiver completamente vazia, ele perderá uma vida, nesse momento caso tenha mais alguma vida restante o jogador voltará a partir da posição gravada no último checkpoint, caso não tenha mais nenhuma vida o jogo exibirá a tela de Game Over, onde a opção de continuar ou não será exibida, no caso de o jogador escolher continuar ele retornará com a quantidade de vidas gravadas e posicionado no início do nível salvo.

Personagens

Eddy – Protótipo de uma série de robôs de um projeto secreto desenvolvido pela Dra. Crystal, O laboratório da doutora foi invadido e Eddy não pôde ser completado a tempo, Eddy recebeu um rápida programação onde consta a busca pelo mineral Oscilium e a destruição da máquina de controle.

Dr.Gênesis – Cientista maluco que foi expulso da comunidade científica por inventar uma máquina que transforma seres vivos em robôs, desenvolveu uma máquina capaz de emitir um sinal que pode controlar qualquer dispositivo eletrônico, seu intuito é combinar suas invenções para tornar-se o Robô –Deus controlador do universo.

Dra. Crystal – Brilhante cientista que desenvolvia um projeto secreto para monitorar os planetas onde se encontram o mineral raro: Oscilium, tal mineral emite uma frequência de rádio muito singular que o torna especialmente apto a alimentar a máquina de controle do Dr.Gênesis.

Z83-Prototype – Robô supostamente destruído pelo Dr.Gênesis, sua unidade de memória conseguiu sobreviver às ações de destruição e ele sobrevive utilizando os restos de sucata que encontra no que deveria ser seu descanso final, as infindáveis pilhas de sucata do

Cemitério de robôs. Um valioso aliado que irá auxiliar Eddy com informações a cerca do doutor maligno e seus planos.

Elementos do GamePlay

Elementos de apoio e medidores / contadores

Life *Panel*–Mostra o número de Vidas que o jogador ainda tem, caso atinja zero o jogador sofrerá Game Over. Mostra um máximo de 99 vidas;

Fixes Panel – Painel que mostra quais os *fixes* que Eddy já coletou. Há no jogo um total de quatro fixes (Não está presente na versão demo).

Health Meter – Mostra as barras de energia de Eddy recebeu cada golpe recebido consome uma determinada quantidade de pontos de vida de Eddy, cada ponto é representado por uma barra vertical: *Damage Unit.* Todas as barras de *damages* vazias significam uma vida a menos, o número de *damage units* pode ser aumentado através da coleta de itens *Health improve* durante o jogo. Eddy terá inicialmente um total de três barras e evoluirá para um máximo de seis.

Itens

Checkpoint - Marcam pontos de retorno ao longo de uma fase, ao perder uma vida ou carregar novamente um nível o jogador retornará para o último checkpoint que ativou. Os checkpoints devem ser ativados quando o jogador passar por eles.

Data disk– Unidades de armazenamento que os robôs usam para trocar pequenas quantidades de informação sem que as trocas sejam percebidas pelos lacaios robóticos de Gênesis (Não está presente na versão demo).

Talkhopter– vídeo comunicador com motor de voo acoplado, capaz de localizar o destinatário de uma mensagem e prossegui-lo para entregá-la. Seu monitor de visualização permite o envio de mensagem de vídeo, voz ou texto (Não está presente na versão demo).

Healing Boxes - Caixas de metal que quando quebradas liberam itens aleatórios que irão auxiliar Eddy em sua aventura, cada caixa pode conter itens de cura, escudos, ou vidas extras (Não está presente na versão demo);

Small healing tool: Itens em forma de ferramenta (pequena) que restauram 1 barra ao *Damage Counter* de Eddy;

Larger Healing Box: Itens em forma de ferramenta (grande) que restauram3 barras ao *Damage counter* de Eddy;

Extra Life – Acrescentam uma vida extra a Eddy, são marcadas com um ícone de coração vermelho;

Shield cristal – Quando tocadas Conferem a Eddy um *Energy Shiled* (Não está presente na versão demo)*;*

Cristais de Oscylum - São poucos e estão espalhados pelos níveis do jogo, cada cenário do jogo revela uma dessas preciosas gemas e é em busca delas que estão Eddy e o Dr. Gênesis (Não está presente na versão demo);

Cogs - Pequenas Engrenagens distribuídas pelos níveis do jogo, ao coletar50 desses cristais Eddy poderá ter uma vida extra;

Fixes – São itens coletados em alguns níveis do jogo que permitem a Eddy a execução de novos movimentos, são quatro no total:

Attack-fix– aumenta a energia dos tiros da arma de Eddy, aumentado seu poder de destruição e a distância que esses tiros podem alcançar.

Flying-fix – aciona a coordenação de funcionalidades das asa e das turbinas de Eddy, como consequência disso o robô poderá realizar o movimento de voo.

InfiniteClimbFix – Permite a Eddy melhor fixação às paredes permitindo à Eddy ficar fixo à uma parede indefinidamente;

Rolling –fix – Aciona o ataque de rolar de Eddy.

Armas, proteções e ataques

Energy shot – Arma que atira esferas de energia concentrada, os tiros causam 1 ponto de dano ao inimigo;

Energy Shield – Campo de força que envolve Eddy, absorvendo os ataques sofridos, para acionar esse recurso Eddy precisa coletar os *Shield Cristals*, cada Shield cristal pode acionar instantaneamente um escudo, cada escudo absorve até 3 pontos de ataque, [Resistência: 3]

Roling attack – quando abaixado Eddy pode rolar no chão na direção do inimigo, causando-lhe um dano de 2 pontos, durante esse ataque Eddy é invulnerável às ações do inimigo(Não está presente na versão demo) ;

Movimentos Personagem principal

Correr – Eddy poderá mover-se em uma corrida de baixa velocidade para frente e para trás

Salto – Pulo normal com distância média, desloca Eddy sobre obstáculos e inimigos.

Empurrão – Eddy pode empurrar alguns objetos grandes para utilizá-los como ponte tornando-se capaz de atravessar buracos que ultrapassem sua capacidade de pular (Não está presente na versão demo).

Escalada – Eddy poderá se prender às paredes para poder as escalar, inicialmente terá um limite de tempo, após o qual deslizará da parede, esse limite será desativado após a coleta do item InfiniteClimbFix (Não está presente na versão demo).

Voo–Combina impulso vertical e planar, conferindo a Eddy a capacidade de efetuar um voo a partir de qualquer ponto elevando-se e mantendo-se no ar por tempo ilimitado. O movimento é ativado pelo item *Flying-fix*(Não está presente na versão demo).

Inimigos

Robôs de Gênesis:

Robôs criados pelo doutor Gênesis para vasculhar pedras de Oscilum e destruir qualquer outro que também as procure.

MechaBee– Inimigo aéreo que sobrevoa áreas específicas, aguardando a aproximação do jogador, a fim de executar um mergulho em direção ao jogador e atingi-lo com seu ferrão, podem também descrever movimentos constantes tornando-se um obstáculo para o inimigo.[Vida:1][Ataque: 1] ;

MechaWorm – Robô-escavadeira que vaga pelo subsolo e surge de repente surpreendendo o inimigo acima dele, seu nome vem da forma de seu corpo, alongada e cilíndrica;

FlyingRoBomber – Robôs voadores que cruzam a tela lançando bombas-relógio por todo o chão, o cronometro desses dispositivos costuma contar um tempo muito curto, mas seu raio de ação é bastante curto e o simples fato de desviar-se a uma pequena distancia da bomba já pode salvar o jogador.[Ataque: 2 cada bomba][Vida: 1];

FastPunch: Pequeno robô move-se pelo caminho , investindo contra o inimigo, tem punhos grandes e espinhos sobre o dorso.[Vida:1][Ataque:1].

SawLauncher: robô que lança serras circulares que rolam pelo chão em direção ao inimigo;

FloatingTank: Robô humanoide com canhões laser no lugar das mãos e uma potente turbina em lugar de seus pés, movimenta-se no sentido vertical para poder capturar o inimigo no ar.

Sphere Mechanisms:

Mecanismos esféricos flutuantes ou rastejantes que descrevem rotas pré-definidas desferindo ataques de forma periódica, tais ataques são baseados no seu elemento de constituição:

IceSphere: Flutua pelo ar e seu ataque consiste na aspersão de nuvens de gás gelado através de ranhuras que vão de seu núcleo gélido à sua superfície;[Vida: 1][Ataque: 2];

Blazin' Sphere: Planam perto do chão ou ficam fixas nele expelindo jatos de chamas, apresentam um disco giratório central de onde as chamas partem que potencializa o alcance do ataque; [Vida: 4][Ataque: 2];

Electri Sphere: Constituídas por um núcleo magnético as funcionais são encontradas flutuando em alturas variáveis, atacam desferindo descargas elétricas provocadas pelo giro de seus anéis estabilizadores. [Vida:2][Ataque: 3];

Spyke Sphere: Esferas cobertas de espinhos, são mecanismos extremamente versáteis por sua própria simplicidade, são utilizadas executando movimentos em grupo ou solo, também são utilizadas de forma estáticas. Sua função é tornar-se um obstáculo ao jogador, não só tornando o seu caminho mais difícil, mas também se colocando estrategicamente de forma a causar-lhe dano.[Vida: 2][Ataque:1];

Smashin' Sphere: Flutuam em Alturas consideráveis esperando que algum desavisado se coloque embaixo delas e então desligando seus motores caem diretamente sobre a vítima, para tornar seu ataque mais eficaz elas apresentam pesadas barras metálicas presas à sua estrutura.[Vitalidade: 2][Ataque: 3].

Gladiadores (Inimigos chefes):

Gladiadores são Robôs ou Mecanismos especialmente preparados pelo Dr.Gênesis para extrair de Eddy os Cristais de Oscilum em seu poder.

Saw Launcher: Grande robô coberto por uma carapaça de espinhos retráteis, imune a ataques de qualquer espécie, como poderoso ataque esse monstro metálico é capaz de lançar séries de serras de metal pelo ar, para atingi-lo é preciso usar suas serras como degraus para atingir seu único ponto fraco sua cabeça.

DeadlyFist: Robô flutuador que ataca com investidas e é capaz de lançar seus avantajados punhos na direção do inimigo. (Twisted Lab Level).

Flying Bomber: Também conhecido como Ultra Mecha Vulture, além de voar na direção do inimigo esse chefe pode sobrevoar todo o campo de batalha, lançando explosivos sobre seus inimigos. (Volcano Hideout)

Junkie ghost: Uma enorme pilha de sucata, unida por um núcleo energético central, a pesar de simples essa criatura é capaz de raciocínio lógico e estratégias de luta;

Ancient Tech: Máquina do antigo povo Zamuh, colosso de pedra com muita força e resistência.

Multi Ball: Mecanismo do tipo Ball melhorado, ao contrário dos demais de seu tipo ele é capaz de desferir ataques baseados em qualquer elemento.

Alfômega: Robô com grande parte da mente do Dr.Genesis, recurso final aplicado para destruir Eddy e abrir o caminho para o domínio da tecnologia mundial. Suas armas incluem Mísseis teleguiados.

Física

Movimento: A movimentação do jogador de corrida e ataque acontecerão sobre plataformas móveis ou não, dessa forma será necessário análise de gravidade sobre os atores que andem, atores que flutuem ou voem podem ignorar essa força, bem como o ator principal durante a execução do voo, a planagem exige gravidade e um simulação de queda**.**

Tiros não sofrerão influência da gravidade seguindo em linha reta até o fim da tela, desde o ponto inicial ou até atingir algum corpo.

Colisão: As colisões ocorrerão entre atores, entre atores e tiros e entre tiros, atores podem apresentar áreas com dano nulo, ou seja, colisões naquela área não acarretaram dano, há também inimigos com estruturas de proteção-ataque essas áreas acarretarão dano ao jogador caso tente aproximar-se.

Os tiros sofrerão anulação quando colidirem, explodirão no ar não ocasionando qualquer tipo de dano.

Inteligência Artificial

Inimigos que executam movimentos independentes da aproximação do jogador executarão *scripts* comportamentais que deverão descrever seus movimentos pela tela durante sua exibição.

Inimigos que percebam a presença do jogador devem executar máquinas de estados que incluem um comportamento padrão de movimentação na tela até que o jogador se aproxime delas a um ponto no chão cuja distância é igual a três vezes sua largura na tela, em caso de inimigos aéreos este ponto será calculado a partir da projeção de seu centro em linha reta no chão, uma vez detectada a presença do jogador, um ataque é desferido e o ciclo é reiniciado até que o jogador saia do campo de detecção, a tela avance ou o inimigo seja destruído.

Interface do usuário

I–Telas de apoio e elementos de interação *Off-game:*

1 - Tela de Título: Deve apresentar o logo do jogo e o "Menu principal", é a primeira tela a apresentada ao usuário;

2 - Tela de carregamento de *save*: Mostra ao jogador o estado da memória de *saves* de jogos e lhe dá acesso a continuar a partir do *save* que escolher;

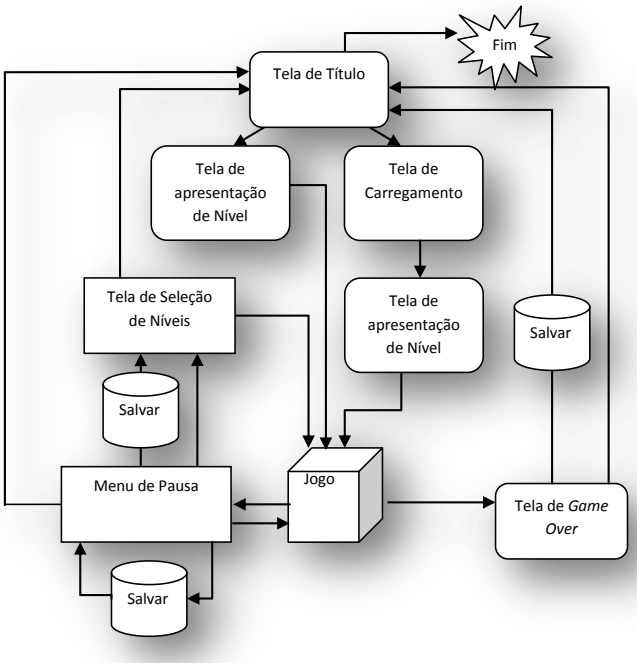
3 – Tela de Exclusão de *Save*: É exibida quando o usuário escolhe apagar um dos *saves* na memória. Exibe um aviso e as opções de confirmação;

4 - Tela de Carregamento de Nível: Mostra ao jogador o Nível que irá jogar e o status do carregamento;

5 – Tela de pausa: Apresentado quando o jogador pausa o jogo e deve exibir o título Pausa e o "Menu de Pausa";

6 - Tela de Game Over: Tela mostrada quando as vidas do jogador se esgotam. Deve apresentar Uma descrição "Game Over", seguida de uma imagem ou pequena animação e o "Menu de fim de jogo";

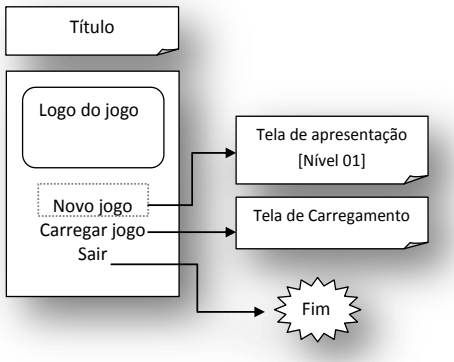
7- Sala de Tele transporte: Apresentará uma imagem de fundo, representando a máquina de tele transporte e o "Menu de Tele transporte", disponível apenas após a finalização dos níveis principais do jogo.



II -Requisitos e esquemas das telas e Menus:

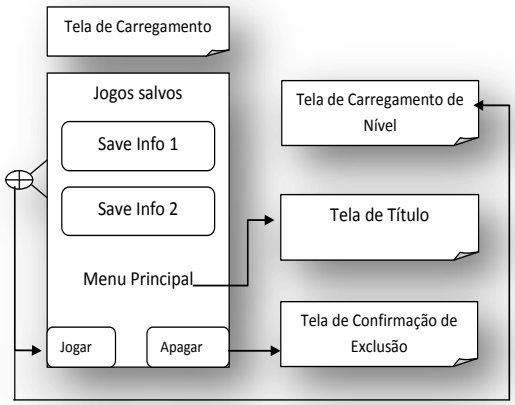
1- Tela de Título e Menu Principal: O jogo deverá iniciar-se com uma A tela de título onde, além do logo do jogo estarão listadas três opções : "Carregar jogo – Passa o jogador para a *Tela de Carregamento* , "Novo jogo" – Direciona para a tela de apresentação do primeiro nível do jogo e "Sair – Encerra o jogo".

Caso o jogador já tenha algum jogo salvo o marcador de seleção nesta tela deve aparecer inicialmente na opção "Carregar jogo".

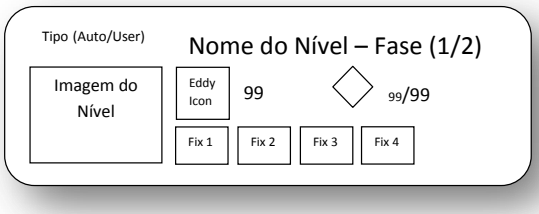


2 - Tela de Carregamento de *Save* e Menu de Carregamento: A tela de Carregamento é exibida quando o jogador escolhe na tela de título a opção: "Carregar jogo", a memória é verificada e a tela de Carregamento adequada é exibida, estarão disponíveis para o jogador um máximo de quatro de *saves*, além do *save* automático.

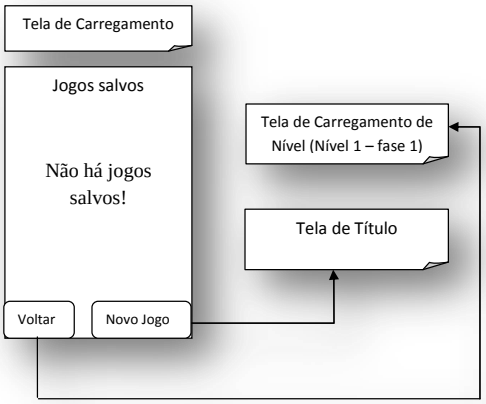
Caso 1 - Pelo Menos um jogo na Memória: Essa tela deve apresentar o "Menu de Carregamento" que consiste numa lista de S*ave data (Save Info),* e duas opções de comando: "Jogar – Retoma a partida gravada no *save*" e "Apagar – Direciona o jogador para a tela de confirmação de exclusão."



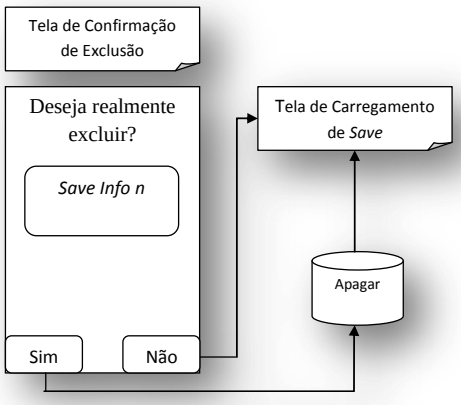
Os s*ave data* são quadros, selecionáveis que exibem informações sobre a partida a que se referem, Os dados exibidos num *save data* são: A origem do Save: Automático (Feito automaticamente pelo jogo no fim de cada Nível completado) ou Usuário (Feito pelo usuário a qualquer momento pelo Menu de pausa, retorna-o para o último checkpoint antes do último save), uma imagem do Nível em que foi efetuado e o nome deste nível, Um ícone representando Eddy seguido do número de vidas que o jogador possui; Um ícone representando os Cristais de Planônium e o número desses cristais que já coletou e do total de cristais do jogo, finalmente, uma lista de ícones representando os fixes que já coletou. Fixes ainda não coletados serão representados por quadrados vazios.



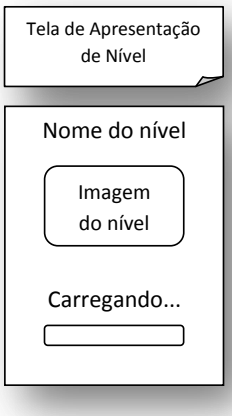
Caso 2 – Nenhum jogo salvo: No caso de não haver nenhum *save* para ser carregado, a tela de carregamento de *saves* deverá exibir no lugar dos *Save Infos* uma mensagem de aviso "Não há jogos salvos!" e as opções de Comandos: "Voltar"–Redireciona o jogador para o Menu Principal e "Novo jogo" – Direciona o jogador à tela de Carregamento do Nível 1 do cenário 1.



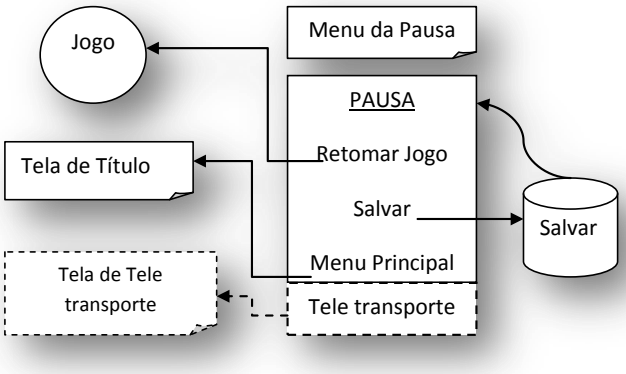
3 - Tela de Exclusão e Menu de exclusão: É Exibida quando o usuário escolhe a opção apagar a partir da tela de Carregamento de *save*, deve exibir a mensagem "Deseja realmente excluir?", seguida do *save Info* selecionado na tela anterior e duas opções de comando, "Sim" – Exclui o *Save* e retorna o usuário para a tela de Carregamento de *save* e "Não" – Retorna o usuário para a tela de Carregamento de *Save.*



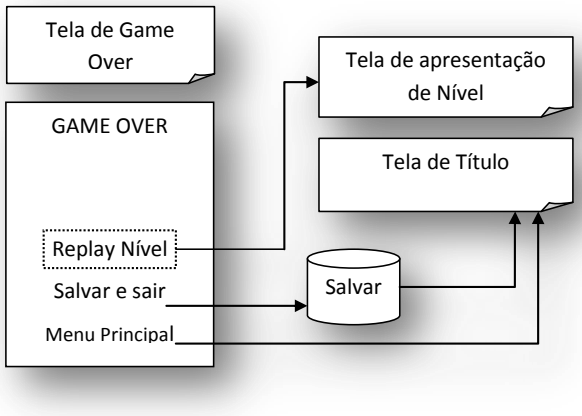
4 – Tela de Carregamento de Nível: É mostrada durante o carregamento da tela do jogo no nível escolhido, deve apresentar o título do cenário e nível, uma Imagem desse nível, e uma barra de status.



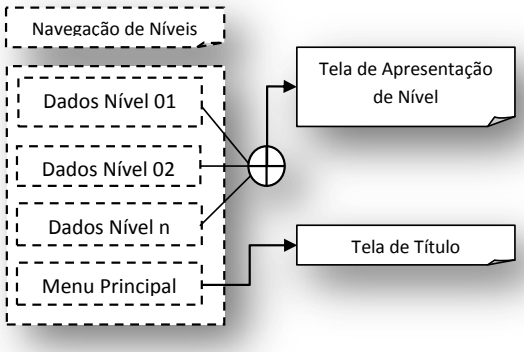
5 - Tela de Pausa e Menu de pausa: Acionado quando o jogador pausa o jogo, deverá mostrar uma pequena lista de opções: "Retornar ao jogo" – Faz com que o menu de pausa seja fechado e o jogador volte a jogar, "Salvar" – Faz com que o estado atual do jogo seja armazenado para carregamento posterior, "Menu Principal" – Encerra o jogo, sem salvar e retorna para a tela de título. Depois do fim do jogo Uma nova opção estará disponível: "Tele transporte" - Direciona para a *Tela de navegação de níveis*", (No esquema abaixo representada com linhas tracejadas).



<u>6 - Tela de *Game Over e* Menu de fim de jogo:</u> Deve apresentar a mensagem "Game Over" destacada e um Menu, em forma de lista, com as opções: "Jogar Nível Novamente – Carrega o nível em o jogador perdeu, desde seu início com o número inicial de vidas de um novo jogo"; "Salvar Nível e abandonar – Salva estado de jogo atual e retorna para a *Tela de título* "Abandonar Nível - retorna para a *Tela de título* sem salvar o estado atual.

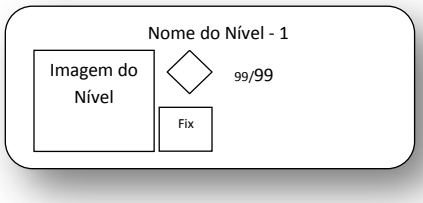


7 - Tela da sala de Tele transporte Menu de Tele transporte: Disponibilizada quando o jogo é finalizado, ele representa a sala tele transporte, nela o jogador escolhe qual o nível que deseja jogar, O menu de tele transporte apresentará uma lista de Informações de Nível (Nível Info), nos quais o jogador pode "clicar" para jogar o nível desejado, deverá também disponibilizar uma opção "Menu Principal" - Encerra o jogo, sem salvar e retorna para a tela de título.



Cada Nível Info deverá informar o nome do Nível, o número da fase, uma imagem da fase, um ícone do *fix* que pode ser coletado nele e um ícone de cristal de Planônium, seguido de dois números, o primeiro representando quantos foram coletados pelo jogador e o outro o total de cristais disponíveis na fase.

Ao acessar a sala de Tele transporte pela primeira vez, o jogador só poderá jogar o primeiro Nível do cenário secreto devendo, como nas demais fases, encontrar o cristal de Oscilum para liberar a próxima fase.



III - Objetos da GUI

Todas as telas de apoio deverão utilizar o mesmo esquema de fontes e cores e as opções, a marca de seleção será representada pela mudança de cor do item iluminado, para itens de texto isso constitui a cor da fonte, para itens com mais dados como os *save infos* seus dados devem ser organizados sobre um retângulo de mesma cor das fontes de opção de texto e sofrerão as mesmas mudanças.

Os comandos nas telas devem seguir os padrões dos sistemas da maioria dos aparelhos, onde teclas direitas e esquerdas podem ser utilizadas para confirmar comandos na tela, confirmação na tecla esquerda e opção voltar/Cancelar na direita, todas as telas devem além das opções especificadas na seção anterior mostrar tais comandos, a seleção deverá contemplar tanto a seleção direta, clicando com o botão confirmar do aparelho diretamente sobre o item, como iluminando a opção e acionando os botões direito e esquerdo.

IV - Relação geral dos Objetos de Vídeo e arte do jogo:

1 - Aspectos gerais da Arte:

Eddy é um jogo de gráficos 2D e scroll horizontal, O jogo deve graficamente simples com cores fortes e pequena variação em um mesmo objeto, Os objetos gráficos do jogo, principalmente os inimigos, Eddy e os itens de jogo devem ser produzidos com gráficos 3Dpré-renderizados.

As animações de personagens devem ser produzidas como séries de fotografias de seus modelos 3D animadas como sprites2D, efeitos especiais, deverão ser produzidos em desenhos 2D ou combinação deles aos modelos como, por exemplo, nos efeitos de danos a Eddy e aos inimigos.

As texturas devem ser reservadas para os cenários, que poderão mesclar elementos de modelagem tridimensional e de desenho, os atores do jogo devem usar cores simples.

Por tratar-se de um jogo de robôs é importante fugir dos visuais muito metálicos, sobretudo nos personagens, a fim de criar um visual diferenciado, devem ser utilizados visuais mais plásticos nos personagens, armaduras coloridas e brilhantes para os robôs e máquinas exceto nos cenários.

2 - Elementos de Arte e animação2D

2.1 - Elementos da GUI *Offgame*

1 – Logo do Jogo;
2 - Tela de Título;
3 - Tela de carregamento de *save;*
4 – Tela de Exclusão de *Save;*
5 - Tela de Carregamento de Nível;
6 – Tela de pausa;
7 - Tela de Game Over;

8–Tela da Sala de Tele transporte;
Ícone/Imagens dos níveis;

2.2 - Cenários:

Pântano de Homera;

2.3 - Itens:

Health Bringer;
Health Relief;
Cristais de Oscylum;
Cristais de Planônium;
StepShots;
GunBooster Fix;
JetPack-fix;
WingedJump-fix;
SkyExplorer-fix;

2.4 - Efeitos especiais:

Eletrocussão de Eddy;
Explosão de Eddy;
Raios;
Jatos de fogo;
Efeitos de pancada (Choque físico); Tiros;
Jatos de gás congelante;
Efeitos de golpes de espada;
Explosões;

3 - Elementos de Arte 3D

Todos os personagens do jogo deverão ser tridimensionalmente modelados, e renderizados para animação por meio de *sprites*(2D);

Itens apresentados durante o jogo também deverão ser desenvolvidos dessa forma;

Os indicadores e ícones nos menus de apoio e telas Off-Game poderão ser desenvolvidos inteiramente em técnicas 2D.

Cenários poderão combinar elementos desenhados e pré-renderizados, mas sua composição será em 2D.

Sons e músicas

I - Músicas

A idéia geral das músicas não é empolgar o jogador, mas refletir a paisagem pela qual passa, ou mesmo a tensão emocional no momento equivalente da história, dessa uma música mais dramática, pode repetir-se em diversos momento no jogo, especialmente relacionada ao inimigo principal.

As músicas, especialmente aquelas referentes aos níveis que se passam nos laboratórios, devem misturar um ar tecnológico, com sons sintetizados de vozes de robôs, ou sons de descargas elétricas etc. O cenário do Pântano deve criar um clima de suspense, mas sem obscuridade, um bom exemplo é o alcançado com a trilhas dos níveis de pântano do jogo: Donkey Kong 2 – Diddy's Kong Quest da Rareware. O cenário do templo deve resgatar uma paisagem de civilizações antigas, com música que lembre as músicas orientais, de preferência com uma batida que lembre as percussões de tambores africanos. *Robot disposal* deve remeter um clima mais frenético, músicas que misturam rock e música eletrônica, Para o cenário no vulcão deve remeter a um clima mais natural como, músicas indígenas ou indianas.

O personagem principal pode ter um tema, que poderá ser tocado nos momentos de vitória, escolha de continue, finalização do jogo (De forma mais enérgica) e pode ser o tema da tela principal do jogo. O mesmo tema pode ser refeito de forma a ter um aspecto mais triste, a fim de ser utilizado na tela de game over.

Os momentos em que Eddy encontra um fixe também devem ter um tema próprio, esse tema deve ter duração muito curta, cerca de 5 segundos já são suficientes.

É Necessário que os arquivos de som tenham tamanhos pequenos, a fim de que possam adequar-se a aparelhos com menores capacidades de memória e processamento, além disso, devem ser produzidas versões desses sons em formatos polifônicos ou MIDs, a fim de que o jogo possa ser executado numa maior gama de aparelhos, incluindo os que não suportam a execução de formatos como MP3, WMA e WAV.

II - Sons e efeitos

Navegação Off-game:
Confirmação de Comando;
Cancelamento de Comando;
Efeitos especiais:
Laser das armas;
Descarga elétrica;
Explosão;
Bipe de radar;
Estiramento de espinhos;
Som de lançamento de chamas;
Sopro, vento (Rajadas de gás congelante).

Characters
Queda de Robô;
Salto;
Impacto metálico (Colisão com os inimigos);
Descarga elétrica (Colisão com carga elétrica);

Flutuação

Game Play Elements
Coleta de Oscilum;
Coleta de Fix;
Coleta de Item Health.

Terreno, ambiente
Lama (Poças em Homer Swap);
Passos metálicos (Eddy andando sobre grades);
Batidas em metal (*Robot Dispo sal*).

História (Roteiro)

Durante um voo de reconhecimento sobre o Pântano de Homera o laboratório móvel da Doutora Cristal é atingido pelas naves do Doutor Gênesis, ciente do perigo que a captura das informações nos computadores da nave significava, a Doutora ativa a cápsula contendo o robô Eddy e a lança no meio do Pântano, enviando-lhe um Talkhopter com suas instruções: Coletar os Cristais antes do Dr.Gênesis e impedir que seus planos de dominação sigam em frente.

A aventura começa no Pântano de Homera, quando o sensor de Cristais de Oscilum de Eddy indicando-lhe a localização da pedra de Oscilum. Depois de vencer os perigos de Homera, Eddy chega até o local indicado por seus sensores: Clareira de Hammy, ali se depara com o primeiro Gladiador de Gênesis, antes de enfrentá-lo o seu inimigo lhe informa que sim ele detém uma pedra de Oscilum, mas o Dr.Gênesis manteve a pedra ali apenas para atrasar Eddy enquanto saqueava os destroços da nave de Crystal. Eddy enfrenta-o e após a sua vitória constata que além do já mencionado Cristal, ele deixa também cair um disco. Eddy acessa o disco e descobre lá um mapa para um local chamado Montanha Muh, com o seu sensor destruído na batalha e sem mais pistas, Eddy resolve seguir para o local do mapa.

Ao chegar a Muh, Eddy logo vê as naves de Gênesis sobrevoando a montanha, para não ser logo descoberto, Eddy prefere utilizar a entrada marcada no mapa do Gladiador como não recomendável. A primeira parte de sua entrada não é percebida e Eddy tem de lidar apenas com os perigos do lugar, no fim dessa primeira parte (Fase 1), o teto da entrada desaba e o grande ruído chama a atenção das tropas no inimigo que durante o resto da travessia irá tentar frustrar a passagem de Eddy até o centro do vulcão. Atingindo o centro do vulcão Eddy se depara com mais um Gladiador que está recolhendo um cristal de uma das paredes da caverna, sem muita conversa esse robô trava uma luta com Eddy.

O cristal preso à parede ainda não havia sido completamente removido quando a luta inicia-se, mas durante a batalha ele cai, revelando também um painel com figuras estranhas, consultando seu banco de dados Eddy descobre tratar-se de uma escrita antiga, esse painel fala da antiga guerra e da localização de outra pedra. Sem saber se o Dr.Gênesis já conhece tal informação, Eddy parte imediatamente para o local indicado.

O local indicado pelas escrituras antigas foi o mausoléu da Rainha Abyrra, de localização desconhecida até então, nem mesmo o Dr.Gênesis sabe desse cristal, mas a ausência dos lacaios de Gênesis não será sentida, já que os próprios Zamuh, construtores da tumba se encarregaram de enchê-la de perigosas máquinas e armadilhas a fim de que, ninguém consiga chegar ao interior da tumba onde está o cristal.

A travessia da tumba é desafiadora, mas Eddy consegue vencer mais esse desafio e chega até câmara do cristal, ao pegar o cristal recebe um sinal não decifrável que vem de dos restos de sua cápsula em Homera, Enquanto busca a saída, Eddy depara-se com o guardião do Cristal, mais uma vez Eddy terá de enfrentar a tecnologia Zamuh, só que dessa vez numa versão bem maior e mais forte. A explosão do Guardião derrotado revela a única saída possível.

Esperançoso de encontrar algum sinal dos tripulantes da nave Eddy volta ao local do acidente, lá um novo Talkhopter aparece, mas esse não faz parte daqueles da doutora Crystal, uma voz misteriosa dá mais informações a Eddy, revelações sobre o plano do Doutor de controlar todos os robôs e a localização de mais um dos cristais de Oscilum. Mais uma vez Eddy decide confiar no destino e segue as instruções recebidas.

O benfeitor misterioso forneceu a Eddy, um conjunto de coordenadas que, segundo ele são do local de exploração onde Gênesis procura por mais um cristal, esse local situa-se nas terras geladas de Fenris e seria o jazigo de uma amostra de Oscilum que fora encapsulada e enterrada. Ao chegar próximo do local indicado, Eddy depara-se com os robôs do doutor Gênesis e percebe que a informação realmente e verdadeira, ele então segue seu caminho nas terras gélidas de Fênris, seguindo os robôs do inimigo a fim de encontrar a pedra, ao chegar ao local final Eddy percebe que não há pedra alguma, antes que ele desconfie de seu informante, mas um Gladiador aparece e lhe informa que a pedra já não está mais lá e que Eddy na irá seguir em frente com sua importunação a Gênesis, Na tentativa de tomar do inimigo a pedra, mais uma luta é travada, mas o robô não tinha a pedra.

Uma nova mensagem chega do informante, ele quer encontrar-se com Eddy, o sinal de suas comunicações com Eddy foi detectado por Gênesis, mas uma importante informação ainda precisa ser dada, além disso, o benfeitor diz ter um cristal que quer dar a Eddy, antes que Gênesis o tome dele.

Ciente da urgência Eddy parte para o local indicado: Deserto de Anchers mais conhecido por todos os robôs como o cemitério dos Robôs, um local assustador para um robô, exatamente por isso Eddy, achou que o seu benfeitor poderia realmente se esconder lá. Mais uma vez alguns lacaios de Gênesis já chafurdavam o local, O que aumentou a certeza de Eddy, seguindo seu caminho em meio às carcaças, robôs de Gênesis e perigos do local, ele repentinamente cai no que parece um enorme buraco. O túnel o leva para o esconderijo do informante, quando a presença de Eddy é detectada, o informante se apresenta: Z83, um protótipo nos moldes do projeto Eddy, também construído para evitar o sinal de Gênesis, copiando o modelo da doutora, Gênesis esperava reverter ou anular a proteção do mecanismo, mas algo não deu certo, pensando ter desativado completamente o robô, Gênesis o lançou no lixo.

Z83, ao contrário do que Gênesis havia pensado, continuava recebendo dados da rede de Gênesis, e seu novo plano havia sido descoberto e essa era a informação que Eddy precisava saber: Gênesis não está concentrado somente na coleta de cristais para o controlador universal de Robôs, mas também está produzindo uma máquina capaz de tornar animais em robôs e seus testes finais já incluíam com sucesso os humanos.

Z83 entrega para Eddy a localização do Laboratório de Gênesis, que já se prepara para os primeiros testes com o controlador, utilizando as pedras que possui, quando o informante se prepara para entregar a Eddy a sua pedra de Oscilum, mais um guardião aparece e se interpõe entre Eddy e seu benfeitor, enquanto Eddy luta contra o chefe, Z83 é sequestrado e junto com ele sua pedra.

A Eddy resta somente seguir as coordenadas dadas pelo amigo sequestrado, a fim de salvá-lo e acabar com os planos de Gênesis.

Ao chegar à entrada da fortaleza do inimigo, Eddy recebe uma nova mensagem, dessa vez de Gênesis, em suas boas vindas, o doutor diz que já esperava por Eddy e que lhe reserva boas surpresas.

Ao finalizar a primeira parte de sua aventura pelas instalações do inimigo, Eddy vê Z83 no chão, ao aproximar-se dele, seu amigo se reativa, mas há um brilho diferente saindo de seus olhos, um brilho avermelhado, diferente daquela luz azul de que Eddy se lembrava. Enquanto observa seu amigo, uma nova mensagem de Gênesis chega e esclarece tudo: Z83 está sob seu controle e sua missão é destruir Eddy e tomar dele todas as pedras que possui.

A luta contra Z83 começa e Eddy, fugindo de seus ataques consegue ver mais uma coisa que não havia nele antes, um pequeno dispositivo em seu peito, sem dúvida aquilo era a fonte do controle de Gênesis. Partindo para atacar o tal ponto, consegue derrubar seu amigo, sem causar mais danos a ele.

Uma vez livre do controle, Z83 informa que o laboratório de Gênesis é na verdade, uma nave pronta para entrar em órbita, onde sua máquina de robotificação está implementada com raio potencializado, com a combinação dessas duas máquinas, Gênesis pretende empreender um transformação em massa no planeta, antes mesmo que Z83 termine de falar, tudo em volta começa a vibrar, a nave já está iniciando sua partida para a órbita de Zilag.

Uma nova mensagem de Gênesis ecoa pela nave: “Apertem os seus cintos! Tudo segue conforme o planejado”, sem entender exatamente o que seu inimigo quis dizer, o robô continua seu caminho, encontrando novos desafios, enquanto segue pelos motores, da aeronave em direção à sala de controle, onde o raio de transformação é ativado. Antes que Eddy possa atingir o seu objetivo, um novo robô do inimigo aparece, mais um Gladiador, mas antes que a luta comece, um novo recado de gênesis, o vilão adverte Eddy que lhe entregue todas as pedras, antes que alguém possa se machucar e então um imagem não esperada é mostrada: A doutora Crystal que diz a Eddy que não entregue os cristais, mas que destrua os planos de Gênesis.

Eddy atende ao pedido da doutora e trava uma luta contra o Gladiador, vencendo o desafio, depois de uma trabalhosa luta, ele ouve um grito muito alto e reconhece a voz: Gênesis! Eddy então percebe que ele está muito perto, destruindo uma porta no fim da sala onde se encontra, o robô vê no alto a prisão de Crystal, em meio a uma sala escura, e mais uma vez a voz de Gênesis, só que dessa vez sem o ruído típico dos comunicadores.

O doutor diz ter poupado energia na busca dos cristais, atraindo Eddy até seu esconderijo, com astúcia conseguiu não só usar Z83 e Eddy em seus planos, mas que agora poderia acabar com todos os empecilhos, Eddy e Crystal, de uma só vez!

As luzes se acendem e o doutor aparece no meio da sala, e dali parado como se nada pudesse atingi-lo, anuncia seu plano: robotificar a todos e controlá-los, depois robotificar a si mesmo tornando-se o imperador de todos os robôs, tendo em si embutido o raio controlador. Eddy então parte pra cima do doutor, mas ao atravessá-lo descobre tratar-se de uma projeção holográfica, enquanto busca sua fonte Eddy, ouve a risada de Gênesis e um enorme robô entra na sala.

- Cumprimente seu fim e meu futuro corpo: Alfômega! Grita o doutor Gênesis e uma luta muito difícil começa a ser travada, ao tomar o primeiro impacto de Eddy, Gênesis revela mais uma carta: Alfômega abre um compartimento em seu peito e lá se encontra a prisão de Crystal é preciso atacar Gênesis o mais rápido possível antes que ele destrua todos começando pela doutora.

A luta contra Gênesis é difícil, já que Alfômega apresenta diferentes e poderosos truques a todo tempo, mas Eddy consegue derrubar o robô, ao cair Alfômega derruba os cristais de Oscilum que estavam em poder de Gênesis. É hora de salvar a doutora Crystal de dentro do robô, assim que Crystal é retirada, o robô volta a se mexer, cabos saem dele em direção às paredes da nave, mesmo machucado Gênesis pretende investir novamente contra

Eddy e já que perdeu os cristais que davam força a Alfômega, tenta sugar a energia da nave. Alfômega se levanta com muita dificuldade, é possível ver descargas elétricas percorrerem seu corpo, e ao levantar um de seus braços um sinal de alerta é acionado: Sobrecarga de energia.

Eddy percebe que a situação pode ficar muito pior e manda a doutora tomar uma das naves salva-vidas e fugir, Alfômega precisa ser destruído e Gênesis neutralizado. O herói se coloca em posição de luta, mas ao tentar lançar o primeiro ataque, o grande robô de Gênesis sofre uma grande explosão, quando Eddy tenta se aproximar para ver se Gênesis está bem uma nova explosão ocorre, e várias explosões menores se sucedem, a voz da doutora é então ouvida no ar, Crystal havia se conectado ao sistema de comunicações da nave.

O núcleo principal de energia da nave estava fora de controle e irá explodir, uma grande viga rompe-se e o teto desaba, tornando impossível que Eddy chegue até Gênesis, agora Eddy deve encontrar um jeito de continuar vivo, o robô começa a correr em busca de mais uma nave salva-vidas, ao que parece, não há mais nenhuma, Eddy então se pergunta se Crystal conseguiu escapar, nesse instante z83 aparece e diz estar procurando por Eddy desde partida de Crystal, duplamente aliviado, Eddy e Z83 entram num módulo de fuga descoberto por Z83 e partindo conseguem fugir da última de definitiva explosão.

De volta a terra firme de Zilag, Z83 teve toda a sua carcaça reformada pela doutora Crystal, em retribuição por toda a ajuda que deu. A doutora por sua vez reúne todas as novas informações que ela e Z83 tiveram tempo de recolher dos computadores da nave e Eddy descansa de sua aventura, na câmara de reparos, ao sair da câmara Eddy é recebido pela doutora que diz ter feito uma importante descoberta.

Crystal conseguiu reunir dados suficientes nas pesquisas de Gênesis que indicavam a existência de mais um cristal. O Oscilum foi mais uma vez considerado perigoso, dessa vez numa pesquisa militar ocorrida em apenas, 3 décadas, uma pesquisa tão secreta que nem mesmo se sabia da existência da base onde ela era realizada, como Gênesis teve acesso a tais dados é impossível saber, mas o fato é que a tal base foi enterrada no deserto de Jônia, onde diziam ter sido localizada a capital da civilização Jônia e que também dizem ter sido um abundante bosque, o fato é que a lendas afastavam as pessoas do local, mas Gênesis não era do tipo de pessoa que acreditava em tais coisas, Eddy decide então, partir para o local especificado pelos dossiês do Doutor para verificar se a pedra ainda estava bem guardada.·.

Eddy prepara-se para ir, mas é interrompido por Crystal, ela lhe diz que como está não poderá vencer os obstáculos e perigos das instalações. A doutora havia desenvolvido vários upgrades para Eddy, mas com medo de que fossem roubados, pouco antes de ter sua nave atacada, ela os espalhou em vários lugares, infelizmente Z83, não pôde recolher todos, mas conseguiu o primeiro upgrade deles: O Jetpack-fix.
Conseguindo recolher todos os fixes (mais três) Eddy poderá vencer todos os desafios das instalações chegando até a câmara de contenção onde está o cristal. È hora de visitar novamente os locais já visitados e com auxilio dos fixes descobrir novos desafios neles, até ter todos os itens necessários.

Após sua nova aventura de coleta das partes perdidas Eddy segue para a localização dada nos dossiês de Gênesis, lá se depara com a entrada aberta, e começa a desconfiar que gênesis estivera lá, ao entrar depara-se com os robôs como aqueles que havia enfrentado quando buscava os outros cristais, confirma sua suspeita: Certamente Gênesis já estivera naquele lugar, mas com o doutor fora de ação, quem poderia estar controlando os robôs?

Novos desafios estão presentes nesse lugar alguns deles intransponíveis sem os novos recursos, mas Eddy consegue vencê-los e chega até a câmara de armazenamento do cristal, mas o cristal não está lá, a cápsula de contenção está rompida e o computador de acesso à cápsula, completamente destruído. Enquanto imaginava o tamanho da coisa ou pessoas que poderia fazer tal estrago, Eddy é surpreendido, por um grande robô, outro Gladiador. Ao

observá-lo Eddy vê em seu centro um grande cristal de Oscilum e logo entende quem está controlando os demais robôs.

Devido à grande influência do poder da pedra, o robô já está completamente descontrolado, isso aliado ao fato de ele ser capaz de convocar qualquer dos outros robôs para ajudá-lo durante o combate, torna a luta extremamente difícil, mas Eddy mais uma vez sai vencedor e consegue tirar do robô a pedra, levando-a a salvo para o laboratório.

Ciente de que em nenhum lugar do planeta as pedras estariam seguras, Crystal as encerra em uma cápsula, posta em um foguete, cujo destino é Zora, a estrela da galáxia Têura, lar do planeta Zilag, lá se espera que, se os cristais não forem destruídos, estejam inalcançáveis.

Níveis

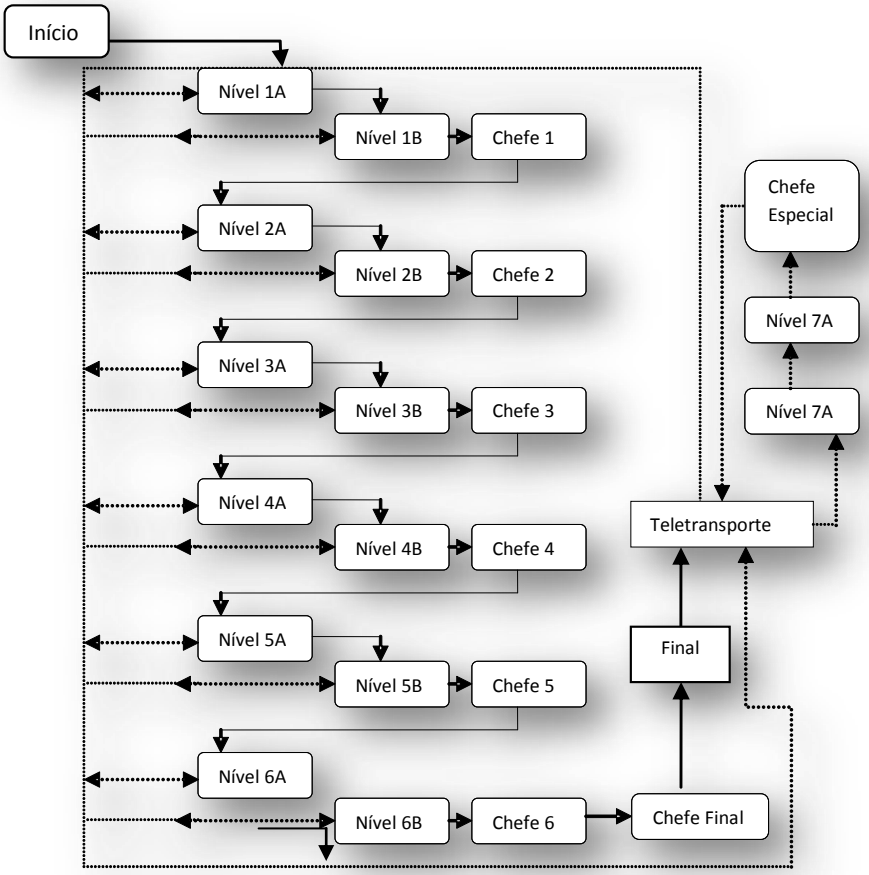
I - Fluxo

Eddy é constituído por seis níveis principais e um cenário extra. A fim de completar o jogo é necessário vencer todos os níveis principais, aqueles nos quais se passa a história principal do jogo, e seus respectivos chefes.

Cada cenário terá dois níveis, A e B que deverão ser vencidas nessa ordem, ao fim dessas o jogador enfrentará um chefe, o qual deverá derrotar para alcançar o próximo cenário. Um nível só será considerado como vencido se, pelo menos um dos cristais de Oscylum que nele está for coletado.

Ao completar os seis cenários principais do jogo, o jogador terá acesso à sala de tele transporte onde poderá retornar a qualquer um dos níveis ou selecionar o nível extra, com os níveis principais completos o jogador também terá acesso ao menu de seleção de salto de níveis, pelo qual poderá deixar qualquer nível em que esteja, no momento que desejar, retornando à sala de tele transporte.

O diagrama que se segue, mostra o fluxo entre os níveis durante o jogo: Os caminhos cheios são os do fluxo normal do jogo, ou seja, o caminho que o jogador deve percorrer para finalizar o jogo, seguindo a ordem da história. Os caminhos pontilhados representam os possíveis caminhos após a finalização do jogo. Os chefes não poderão ser diretamente acessados através de tele transporte.



II - Descrição Geral dos níveis e suas paisagens

Cenário 1 – *Acido rock!*

Local: Homera Swamp

O Pântano de Homera não é exatamente um pântano, mas é assim chamado pela coloração esverdeada de suas águas estranhamente cheias de ácido, altamente corrosivo e mortal. A queda da nave de Eddy o leva primeiramente ao subterrâneo dessa área, um lugar escuro e de poucas formas de vida, nesse ambiente existem apenas fungos, armadilhas de ácido e rocha acinzentada, a o vencer essa primeira etapa Eddy chegará à superfície do local, aqui pequenas porções de rocha são cercadas pelo grande lago Homera, os fungos predominam, mas sua variedade de cores e tamanho é muito maior, além disso, pequenas plantas também estão presentes. A natureza da água fazem com que o ar próximo ao lago tenha uma espessa camada de névoa que faz com que o lago visto de cima pareça uma grande massa de águas claras e limpas, no entanto acima dessa área o céu é limpo e claro.

Cenário 2–*Hot Rock!*

Local: Montanha Muh

As terríveis consequências do uso do Oscilum em armas durante uma guerra levaram a antiga civilização Zamuha lançar o cristal dentro de um vulcão pra se proteger. O Oscilum de Muh está dentro da mais profunda galeria cavada na rocha pelo magma. A atmosfera da montanha Muh é muito perigosa, pedaços de rocha flutuam sobre um imenso lago de magma,

além disso, pedras soltas podem cair a qualquer momento sobre a cabeça de um corajoso desavisado, sem mencionar os esguichos de vapor e magma que podem surgir de súbito no caminho.

Cenário 3–*An Old Tomb!*

Local: Abyrra temple

As terríveis consequências da guerra deram-se porque ambos os lados usaram do poderoso artefato, quando os Zamuh venceram os Jônis, descobriram que eles também tinham descoberto o poder da pedra e tomando a pedra Jônis lançaram-na no vulcão, mas a pedra Zamuh depois de ter sido proibida foi guardada no mausoléu da Rainha Abyrra, condutora do exército vitorioso Zamuh. Abyrra foi profundamente afetado pelos horrores da guerra e ordenou que a pedra fosse guardada por armadilhas.

Cenário 4 – *A new Tomb!*

Local: Fenris Icy Tomb

Em algum lugar do continente congelado de Fenris está enterrada uma cápsula contendo mais um cristal de Oscilum, O caminho até o cristal da Cápsula OSC01 é composto por uma estrada coberta de neve com gelo escorregadio em algumas partes, além disso, muitas passagens estão bloqueadas pelo intenso frio e outras foram completamente destruídas revelando rachaduras perigosas nas rochas, não há vida nesse local, pelos menos não vida macroscópica, só grandes estruturas de gelo acumulado nas quais ainda é possível ver os aventureiros que anteriormente desafiaram o local.

Cenário 5 –*Trip in trash*

Local: Robot disposal

O Deserto de Anchers é mais conhecido como Aterro de robôs, desde muito tempo velhas naves, máquinas e principalmente robôs e suas partes são lançados aqui quando não funcionam mais, esse foi também o local escolhido pela Dr.Crystal para depositar o primeiro protótipo da séria ‘E’, o Edda (Eddy alpha), nesse robô a doutora escondeu uma unidade de memória contendo informações sobre a localização de mais uma provavelmente a última Pedra de Oscilum. A paisagem desse nível é composta por areia e carcaças, os amontoados de metal retorcido formam não só plataformas e pontes, mas também perigosas armadilhas, o que parece ser uma segura passagem pode ser uma versão metálica de areia movediça. Pedaços formam lâminas afiadas pelo tempo e enchem as fossas do grande depósito, tornando-se armadilhas mortais, no entanto algumas das passagens de cabo permanecem e como um bonde os aventureiros podem atravessar algumas partes do deserto.

Cenário 6–*Flying Danger*

Local: Genesis’ Lab Airship

Laboratório móvel do Dr.Gênesis, que acaba de ser finalizado, quando da chegada de Eddy. o cenário compõe-se do interior de uma nave, semelhante ao um avião só que com mais espaço e com estruturas de metal à mostra, computadores e máquinas como casulos, fios e cabos no teto e no chão, Também fazem parte do cenário pedaços de armas, armas e munições, também mapas e globos, além das portas bloqueadas por teias de raios laser.

Cenário Extra–*Yet another Place to go!*

Local: Deserto de Jônia

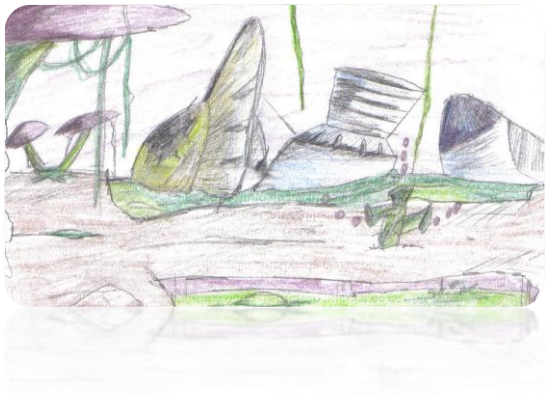
Instalação militar abandonada: Pesquisas com outra parte do meteorito resultaram em um grande acidente e o projeto foi abandonado e a área enterrada. Este nível contém desafios adicionais que só podem ser vencidos após a obtenção de todos os fixes. A estação militar estende o visual do laboratório do Dr.Gênesis, unindo-o com um visual desértico do local onde está instalado, as terras áridas de Jônia, Local da decisiva batalha antiga, O cenário combina uma vastidão de dunas a pedaços de ruínas espalhadas e restos de metal retorcido e pedaços de robôs e armas.

Níveis

Cenário 1: Acid rock!

Background: A cápsula de Eddy é lançada no Pântano de Homera e seu detector de cristais aponta a emissão de radiação de um cristal próxima ao local da queda, a missão de Eddy é seguir essa emissão até encontrar o cristal.

Localização: Pântano de ácido do planeta zilag. O Pântano de Homera é composto por uma paisagem de cores muito pouco variadas marcada pelo verde do lodo e do lago de ácido que evapora formando uma nuvem espessa e perigosa para os pilotos desavisados, é um local perigoso para a aviação e muitas carcaças de naves estão espalhadas por todo lado. Em Homera predominam as plantas Quózomas, semelhantes a cogumelos verdes com copas roxas, felizmente inofensivos, por outro lado o Pântano mais inóspito de Zilag é também lar dos terríveis fungos Trombói, esses habitantes secretam um ácido muito poderoso que é capaz de corroer o mais resistente material.



Elementos presentes no cenário:

Chão pedregoso e com manchas de ácido,
Lagos e poças de ácido,
Pedaços de naves (Estáticas, naves humanas, discos voadores e pedaços de metal retorcido), [REMOVIDO].
Árvores Kugag (Assemelham-se a cogumelos grandes, seu tronco é como um pé de cebola e de sua copa caem vários cipós – figura abaixo).

Perigos do nível (Itens do cenário que causam danos)

Bolhas de ácido – em Homera existem duas fontes dessas bolhas as plantas Trombóides e algumas partes do lago de ácido. [REMOVIDO] As bolhas saem de suas fontes e flutuam no ar em linha reta até explodirem em algum ponto da trajetória;

Lagos e poças de ácido – A composição dominante dos líquidos que formam os lagos em Homera é de um ácido muito poderoso, que até onde se sabe, não pôde nem mesmo ser analisado.

Chefe do nível: Deadly Fists (Não na versão demo)

Cinematics do nível:

"Aventura à vista" – Introdução, (Antes do início do Nível), a derrubada da nave de crystal e seu comando para Eddy iniciar sua busca;
"Uma pista a seguir" – Eddy descobre no disco que caiu do chefe a localização de uma nova pedra.

Inimigos

Mecha Bee - 01 -

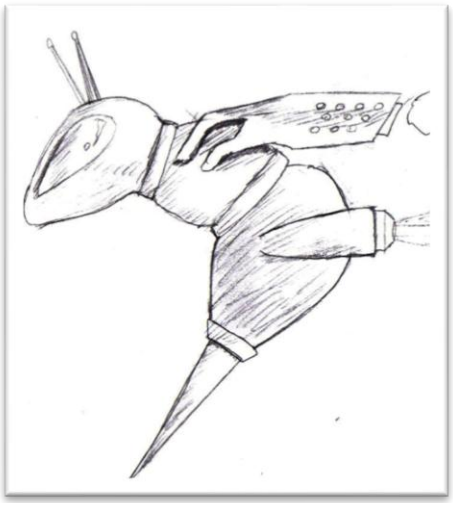
Robô planador com corpo semelhante ao de uma abelha pode sobrevoar o solo em diversas alturas desde poucos centímetros do chão até dezenas de metros dele. Seu ponto de ataque principal é o seu ferrão, uma poderosa lâmina de aço acoplada em sua extremidade mais baixa.

Atributos:

Vida: 1 ponto;
Ataque: 2 pontos;
Defesa: 0 pontos;

Tipo: Aéreo;
Dimensões: Normal.

Esboço:



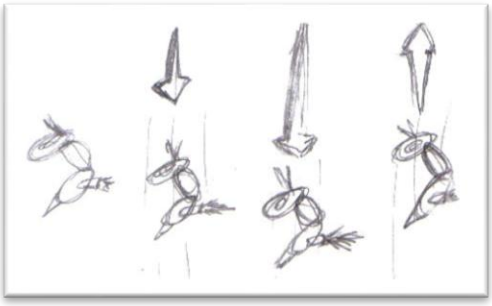
Outros:

Há, nos diversos níveis do jogo, três tipos de MechaBee de acordo com seu tipo de ataque:

1- Oscilator : Executam Ataque fixo-vertical;
2 - Simple: Infringem dano colocando-se como Obstáculo;
3 – Hunter : Executam a Investida aérea.

Ataques:

1 – Ataque fixo vertical: Os MechaBee's descrevem um movimento repetitivo, subindo e descendo, acima de um ponto fixo do solo, esse movimento pode ser executado por grupos desses robôs e com alternação de velocidade e altura máxima do movimento;



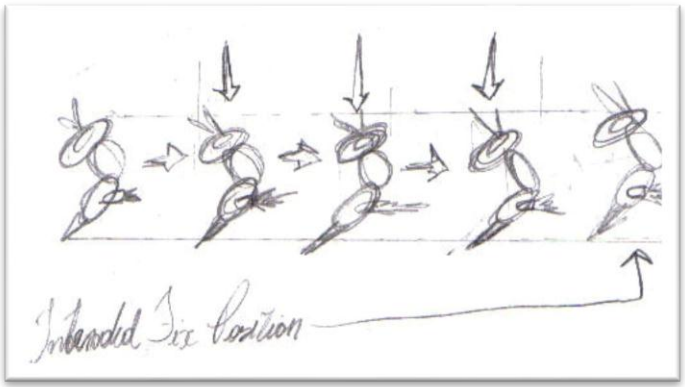
2 - Investida Aérea: Investida em direção ao jogador, o robô detecta o jogador em determinada posição e inicia uma queda prevendo o avanço do mesmo.

3 - Obstáculo: Robôs planam em um ponto fixo do ar, ficando em posições estratégicas, a fim de funcionar como obstáculo no caminho do jogador.

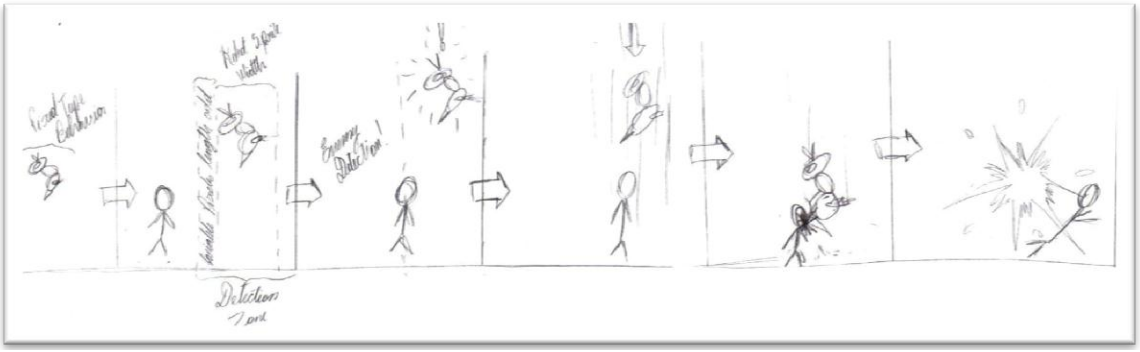
<u>Comportamento:</u>

Emissão de fogo: Durante seu voo os Mechabee's emitem tamanhos variáveis de chamas de sua turbina traseira, essa turbina está diretamente ligada ao seu motor de voo, logo, qualquer Mechabee ativo independente de seu tipo irá emiti-las, durante todo o voo.

Fixos - A potência do motor desse Robô provoca nele, quando parado no ar, pequenas vibrações que o fazem oscilar ligeiramente de sua posição;



Investida – Esse tipo de robô também permanece em um ponto fixo até detectar a presença de um inimigo, <u>sua movimentação até o momento da detecção do inimigo é igual à do tipo fixo</u>, após a investida esse robô fica fixo por 3 segundos e depois explode.



FASTPUNCH

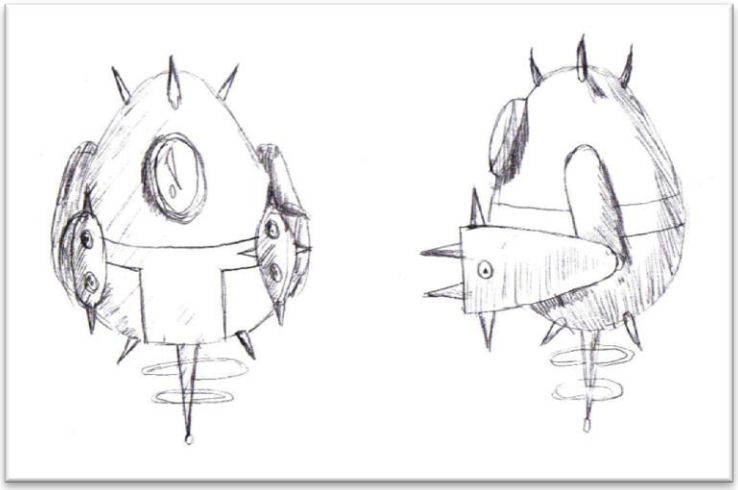
FastPunch é um robô de pequeno porte, terrestre com uma forte carapaça que tem na sua parte frontal dois punhos grandes que ele utiliza para causar danos aos seus inimigos, apesar de seus punhos não se moverem, possuem poderosos espinhos que aliados à rápida movimentação desse robô, torna seus ataques bastante eficazes.

Atributos:

Vida: 1;
Ataque: 2;
Defesa: 0

Tipo: Terrestre;
Dimensões: Normal.

Esboço:



Ataques:

1-Ataque Investida: O robô corre em direção ao inimigo para atingi-lo com seus punhos.

Comportamento:

FastPunch aguarda parado a aproximação do inimigo, quando percebe a presença do jogador ele avança em sua direção até encontrá-lo, ou um obstáculo, nesse momento, FastPunch muda de direção e segue em fuga, até encontrar um novo obstáculo, e mudar novamente de direção: Fast Punch é acionado pela presença do jogador iniciando um movimento contínuo de fuga e mudança de direção quando encontra algum obstáculo.

SPYKESPHERE

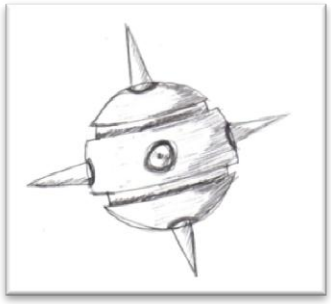
Esferas cobertas de espinhos, são mecanismos extremamente versáteis por sua própria simplicidade, são utilizadas executando movimentos em grupo ou solo, também são utilizadas de forma estática. Sua função é tornar-se um obstáculo ao jogador, não só tornando o seu caminho mais difícil, mas também se colocando estrategicamente de forma a causar-lhe dano.[Vida: 2] [Ataque:1];

Atributos:

Vida: 2;
Ataque: 1;
Defesa: 0;

Tipo: Terrestre/Aéreo;
Dimensões: Normal.

Esboço:



Outros:

As spykeSpheres podem ser de três tipos:

Horizontal: Rolam sobre o solo, até uma distância pré-definida, depois retornam ao ponto inicial, repetindo o movimento, são geralmente encontrados em buracos e espaços apertados;

Vertical: descreve um movimento repetitivo, subindo e descendo, acima de um ponto fixo do solo, esse movimento pode ser executado por grupos de esferas e com alternação de velocidade e altura máxima do movimento;

Estáticos: Esferas com problemas em seus circuitos que as impedem de se movimentar são depositadas por lugares diversos, nos cenários do jogo, formando armadilhas perigosas para o jogador descuidado;

Flutuantes: Mantém uma posição fixa no ar.

Ataques:

As Spykespheres não executam nenhum ataque, funcionam somente como obstáculos.

Comportamento:

Rotação: Os esferas do tipo vertical e do tipo flutuantes, executam simultaneamente o seus movimentos característicos e um movimento de rotação;