Universidade Federal de Pernambuco

Graduação em Ciência da Computação
Centro de Informática

Um estudo sobre a relação entre os artistas e os programadores dentro da esfera de criação de jogos.

Trabalho de Graduação

Leonardo Moreira André

Orientador: Geber Lisboa Ramalho

Recife, 2010

Universidade Federal de Pernambuco

Centro de Informática

Um estudo sobre a relação entre os artistas e os programadores dentro da esfera de criação de jogos.

Trabalho de Graduação

Leonardo Moreira André

Pesquisa de Graduação apresentada no Centro de Informática da Universidade Federal de Pernambuco por Leonardo Moreira André, orientado pelo PhD. Geber Lisboa Ramalho, como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Geber Lisboa Ramalho

Recife, 2010

"Virá o dia em que um rei reinará com justiça e as autoridades governarão com honestidade."
Isaías, o profeta

AGRADECIMENTOS

Esses foram os quatro anos mais difíceis da minha vida e não chegaria aqui, se Deus não me ajudasse. Embora tenha me afastado um pouco d'Ele, Ele ainda assim permaneceu comigo. Por isso, quero agradecer de corpo e espírito primeiramente a Deus por ser tão bom comigo e por ser um grande amigo.

...

Em segundo lugar, quero agradecer a minha família por toda dedicação, atenção, carinhos e apoio que obtive. Aos meus pais que sempre se sacrificaram para que um dia eu chegasse até aqui (este trabalho também é deles). A minha namorada, Ana Luiza, por ter me dado forças e motivação sobrenaturais que nenhum líder conseguiria me proporcionar. À vida dos meus irmãos que sempre me ensinaram qual caminho traçar nesta caminhada acadêmica, pois sempre foram mais experientes do que eu.

A todos os meus amigos que me levantaram quando eu caí por diversas vezes e que me ensinaram tudo o quanto eu precisava. E, em especial a Marcos, Pedro e Felype, pela parceria que me proporcionaram nos projetos difíceis e estressantes da faculdade; sem eles não teria chegado até aqui.

Aos meus amigos Júlio e Bráulio que, embora eu tenha me ausentado tanto devido a dedicação à faculdade, ainda assim permaneceram comigo.

Por fim, agradeço ao meu orientador, super gente fina que me deu o suporte necessário para a elaboração deste trabalho.

RESUMO

Antigamente, a arte de desenvolver jogos era trabalho exclusivo de um simples programador. Hoje, podemos afirmar que esse quadro mudou completamente: grandes grupos, envolvendo diferentes áreas, constituem uma equipe de projeto para criação de jogos. O objetivo principal deste trabalho é analisar e elaborar um estudo profundo sobre a relação entre o grupo dos artistas e a equipe de programadores dentro do ambiente de desenvolvimento de jogos digitais.

Palavras-chave: Jogos, artista, programador, interação.

ABSTRACT

It used to be that the art of developing games was exclusively work of programmers. Today we can say that this scenario has completely changed--large groups, encompassing different areas, constitute a project team to create games. The main objective of this report is to analyze and study deeply the relationship between the group of artists and the group of programmers within the context of digital game development.

Keywords: games, artist, programmer, relationship, interaction.

SUMÁRIO

1. INTRODUÇÃO 1
1.1. CONTEXTO DO PROBLEMA 2
1.2. OBJETIVOS 3
1.3. MOTIVAÇÃO 3
1.4. ESTRUTURA DO TRABALHO 4
2. CONCEPÇÃO DO DESENVOLVIMENTO DE JOGOS 5
2.1. POR QUE ESTUDAR DESENVOLVIMENTO DE JOGOS ELETRÔNICOS? 6
2.2. ETAPAS DE DESENVOLVIMENTO 6
2.3. DISCIPLINAS CLÁSSICAS DE DESENVOLVIMENTO DE JOGOS 11
3. PROCESSO DE DESENVOLVIMENTO DE CÓDIGO 13
3.1. HISTÓRIA DO PROGRAMADOR NA FABRICAÇÃO DE JOGOS 14
3.2. FERRAMENTAS PARA AUXÍLIO NO PROCESSO DE DESENVOLVIMENTO CÓDIGO PARA JOGOS. 16
3.3. MOBILE, UMA DIFERENÇA PECULIAR 17
4. PROCESSO DE DESENVOLVIMENTO DE ARTE 19
4.1. OS "ARTISTAS" DE JOGOS: UM CONCEITO NÃO FORMADO 20
4.2. FERRAMENTAS PARA O AUXÍLIO NO PROCESSO DE DESENVOLVIMENTO DE ARTE 21
4.3. ETAPAS DO PROCESSO DE DESENVOLVIMENTO 25
5. CARACTERÍSITCAS RELACIONAIS ENTRE OS PROGRAMADORES E OS ARTISTAS 28
5.1. QUE INFORMAÇÕES SÃO TROCADAS ENTRE OS ARTISTAS E OS PROGRAMADORES? 29
5.1.1. DOS ARTISTAS AOS PROGRAMADORES 29
5.1.2. DOS PROGRAMADORES AOS ARTISTAS 32
5.2. EM QUE MOMENTO HÁ A TROCA DE INFORMAÇÕES? 32
5.3. COMO É FEITA A TROCA DE INFORMAÇÕES? 33
5.4. ESSA TROCA DE INFORMAÇÕES É SUFICIENTE PARA O DESENVOLVIMENTO DO TRABALHO? ... 34
5.5. EXISTE DEPENDÊNCIA NOS TRABALHOS ENTRE OS ARTISTAS E OS PROGRAMADORES? 35
5.6. QUE FERRAMENTAS PODEM AXILIAR ESSA RELAÇÃO? 36
6. CONCLUSÃO 38
6.1. GUIA DE PESQUISA 39
6.2. TRABALHOS FUTUROS 39
REFERÊNCIAS BIBLIOGRÁFICAS 40

LISTA DE FIGURAS

FIGURA 1: ORDEM DAS ETAPAS DO DESENVOLVIMENTO. 7
FIGURA 2: BRAINSTORM PARA O DOCUMENTO CONCEITUAL 8
FIGURA 3: DOCUMENTO CONCEITUAL PARA O DOCUMENTO TÉCNICO 9
FIGURA 4: NOTA DOS TESTES REALIZADOS PELA LEVEL UP! NO JOGO COMBAT ARMS (2010)., ... 10
FIGURA 5: GEOMETRIAS RETALHADAS COMPÕEM O BOXING – 1980 (ATARI) 14
FIGURA 6: DESENHO BASEADO EM PINTURA DE PIXELS NO RIVERRAID - 1982 (ATARI) 15
FIGURA 7: GRAND THEFT AUTO: EPISODES FROM LIBERTY CITY. (DIRECTX 9.0C) 17
FIGURA 8: HALF-LIFE. (OPENGL) 17
FIGURA 9: SUPER MARIO WORLD (BIDIMENSIONAL) 21
FIGURA 10: SUPER MARIO GALAXY (TRIDIMENSIONAL) 22
FIGURA 11: COMPARAÇÃO ENTRE BITMAP E SVG (BITMAP À ESQUERDA E SVG À DIREITA) 24
FIGURA 12: CICLO DE ATIVIDADE DO ARTISTA (CONCEPÇÃO) 26
FIGURA 13: CONCEPÇÃO -> MODELAGEM 26
FIGURA 14: VERSÃO INICIAL DO MENU PRINCIPAL (MANIFESTO GAMES STUDIO - ANIMAL SHARDS) 31
FIGURA 15: ÚLTIMA VERSÃO DO MENU PRINCIPAL (MANIFESTO GAMES STUDIO - ANIMAL SHARDS) 31

1. INTRODUÇÃO

Este capítulo relata uma breve contextualização do problema, as principais motivações para a elaboração deste trabalho, os seus objetivos da pesquisa realizada e, ao fim, apresenta como estão organizados os capítulos descritos.

1.1. CONTEXTO DO PROBLEMA

Em 1958, o primeiro jogo eletrônico, intitulado "*Tennis for two*", foi criado utilizando-se de um osciloscópio e um computador analógico. Uma simulação bem simplificada do tênis de mesa (Ramos, 2008). Após três anos, em 30 de julho 1961, uma equipe de estudantes do MIT (*Massachusetts Institute of Technology*) criou o famoso "*Spacewar!*", um jogo eletrônico desenvolvido a partir de um enorme computador de milhares de dólares.

A dificuldade do mercado consumidor de jogos eletrônicos ao acesso a esses computadores, ou qualquer dispositivo eletrônico semelhante, gerava conseqüências infelizes que impediam a produção ou venda dos jogos.

Antigamente, a arte de desenvolver jogos era trabalho exclusivo de um simples programador. Não era exigido muito recurso ou habilidades diferenciadas para elaborar um jogo. O desenvolvimento do jogo tinha apenas como objetivo principal a diversão. Alguns programadores, por desejos pessoais, construíam jogos eletrônicos como forma de descontração ou até mesmo para demonstrar superioridade de conhecimento aos colegas. O lucro não era fonte primária dos desenvolvedores. Não é a toa que muitos desenvolvedores utilizavam o desenvolvimento de jogos como *hobby*.

Porém, com o passar dos anos – diga-se de passagem, não foram muitos – a arte de desenvolver jogos foi dando passos largos em direção ao mercado financeiro. Os objetivos de criação de jogos eletrônicos, hoje, são bastante diferentes dos de 40 anos atrás. O amplo mercado para investir em jogos eletrônicos trouxe mudanças significativas para a indústria de jogos.

Essa mudança afetou a forma de como construir jogos. Com a globalização e a modernização tecnológica, a facilidade de acesso aos computadores e o crescimento do nível de exigência do mercado consumidor, a equipe de criação de jogos passou de um simples trio de programadores a um grande diversificado grupo de desenvolvimento. Dentre esse grande grupo, podemos notar a existência de várias equipes multidisciplinares: **marketing, gerência, game design, programadores,**

artistas, músicos, geógrafos, físicos, ambientalistas, historiadores, entre outros (Ramalho); surge-se, então, a necessidade de estudar a relação que os regem no importante processo de criação de jogos eletrônicos.

1.2. OBJETIVOS

Entre tantas ligações que podemos obter através da permuta das várias disciplinas vistas na seção anterior, enfatizaremos na específica relação entre os artistas e os programadores. Nessa dependência, há ainda muito a ser analisado. Devido à grande falta de informação, sabe-se pouco a respeito de como é dado o entrelaçamento profissional entre essas duas partes.

O objetivo principal deste trabalho é analisar e elaborar um estudo profundo sobre a relação entre o artista e o programador dentro da esfera de criação de jogos. Dentro desta análise, as seguintes contribuições serão destacadas:

- Analisar o processo de desenvolvimento de jogos utilizados nas empresas.
- Elaborar um resultado das pesquisas obtidas e da análise do processo de desenvolvimento através deste trabalho.

1.3. MOTIVAÇÃO

A grande motivação se dá através do desafio já comentado: a falta de pesquisas de estudos e informação a respeito de como é dada a relação profissional entre os artistas e os programadores.

Devido a essa escassez, o desafio da elaboração do trabalho apresentado fica ainda maior. Por consequência, a maior parte dos resultados e estudos obtidos será de forma empírica e objetiva. Ou seja, através das aplicações de entrevistas realizadas nos grupos acadêmicos e nas empresas de desenvolvimento de jogos eletrônicos o resultado será construído.

1.4. ESTRUTURA DO TRABALHO

Este trabalho é composto por seis capítulos. No capítulo segundo, são abordados aspectos importantes a respeito dos jogos eletrônicos para oferecer uma introdução mais clara a respeito do contexto do trabalho realizado.

Após demonstrar a fundamentação de um jogo eletrônico, os dois capítulos subsequentes apontam em separado a profissão do artista e do programador de jogos.

No capítulo quinto, o mais importante deste estudo, serão abordados os questionamentos que rodeiam a relação entre os artistas e os programadores dos jogos eletrônicos.

Ao final, são feitas considerações finais acerca do trabalho.

2. CONCEPÇÃO DO DESENVOLVIMENTO DE JOGOS

Este capítulo mostra um pouco sobre os desenvolvimentos de jogos no âmbito geral. Traz inicialmente uma idéia mercadológica do setor como forma de motivação para desenvolver jogos eletrônicos. Logo depois, apresenta as etapas a serem cumpridas para a criação de um jogo. E por fim, as divisões e os papéis dos desenvolvedores do projeto.

2.1. POR QUE ESTUDAR DESENVOLVIMENTO DE JOGOS ELETRÔNICOS?

A pergunta pode ser respondida por diversos modos de pensamento. Financeiramente, o mercado de jogos eletrônicos tem se desenvolvido bastante em todo o mundo. É um setor de crescimento inigualável na área de entretenimento digital, e tem aquecido o mercado financeiro de vários países. Somente nos Estados Unidos foram movimentados 11,7 bilhões de dólares em 2008 (um crescimento de 30% em relação ao ano de 2007). No Brasil, somando hardware e software, o produto interno bruto no setor de jogos é de R$87,5 milhões (Queiroz, 2009).

Pesquisando no Google a frase “mercado de jogos”, observar-se-á uma grande quantidade de menções sobre o crescimento do setor na economia. Encontra-se até mesmo uma recente matéria sobre o aumento econômico na área mesmo frente à crise de 2009 que assolou o mercado mundial (Gandra, 2009).

Além dos termos financeiros, existe também outra motivação para criação de jogos digitais. Diferentemente dos outros tipos de softwares, desenvolver jogos eletrônicos possui um desafio maior: nenhum usuário é obrigado a jogar. Ou seja, nenhum consumidor precisa efetivamente comprar um jogo. Aplicativos como edição de texto, construção de planilhas financeiras, gerenciamento de recursos podem ser necessários para o usuário dentro de sua corporação. Porém, com jogos, essa necessidade não existe.

Portanto, podemos observar o quão desafiador é produzir um jogo digital. Desenvolver jogos exige uma habilidade/criatividade para “seduzir” o usuário com uma iteração usuário-máquina eficiente (Alves, 2007).

2.2. ETAPAS DE DESENVOLVIMENTO

Por mais que se tente relacionar o desenvolvimento de um jogo como uma produção de software comum, os jogos digitais possuem uma série de particularidades que diferem no desenvolvimento de um software em geral. Os processos oriundos da

Engenharia de Software, por muitas vezes, sofrem algumas alterações para se comportar de maneira efetiva no projeto de desenvolvimento do jogo.

Se for para comparar, atualmente, o desenvolvimento de jogos se assemelha mais à produção de um filme que a um programa de computador propriamente dito. Dessa forma, o processo de criação de jogos é bem mais complexo que um software comum e exige maior atenção quanto ao seu desenvolvimento. Conforme dito anteriormente, o desenvolvimento de jogos conta com a participação de várias equipes multidisciplinares: **marketing, gerência de projetos, game design, produtores, programadores, artistas, músicos, geógrafos, físicos, ambientalistas, historiadores**, entre outros (Ramalho).

É bom saber que produzir jogos deixou de ser pura diversão e se tornou um modelo de negócios. Como todo modelo, existe a linha do tempo do desenvolvimento de jogos digitais que está dividida em cinco fases: **concepção, pré-elaboração, elaboração, prototipagem e pós-venda** (Sloper, 2005).

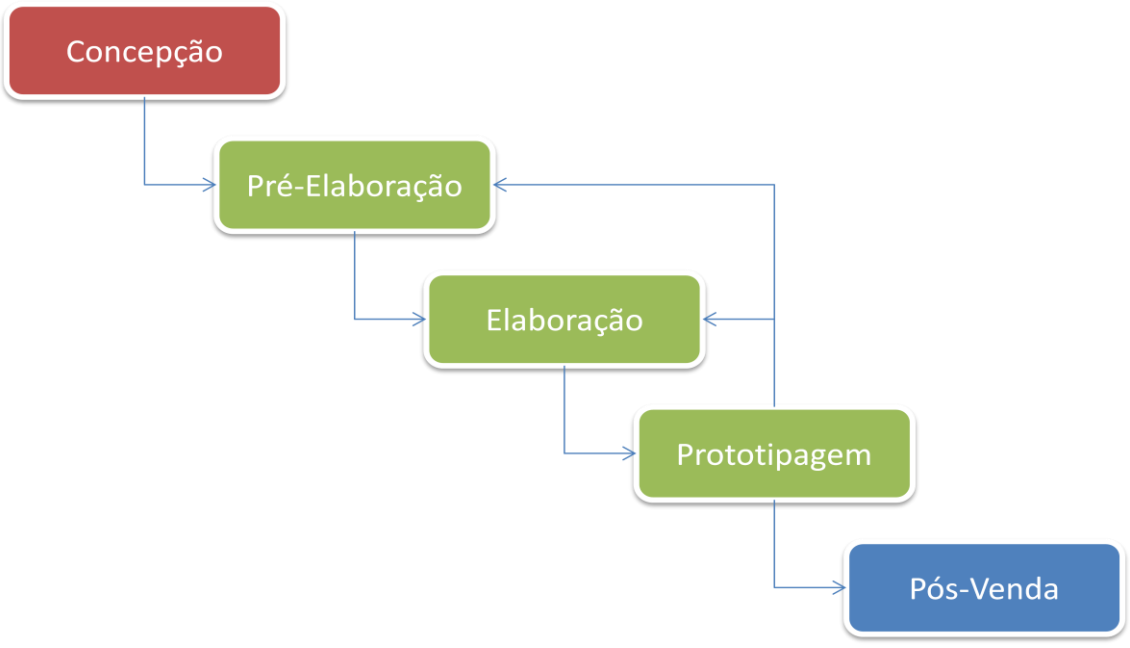

Figura 1: Ordem das etapas do desenvolvimento.

O desenvolvimento do jogo se inicia na fase de concepção com a criação do documento conceitual do jogo. Esse documento contém informações primárias sobre o projeto a ser desenvolvido tais como: análise de mercado, tomadas de decisões, modelagem do jogo. Além disso, o documento tem o objetivo de integrar todos os envolvidos do projeto do jogo, deixando-os cientes da concepção da ideia que será seguida pelo grupo. Esta é a fase de concepção do jogo.

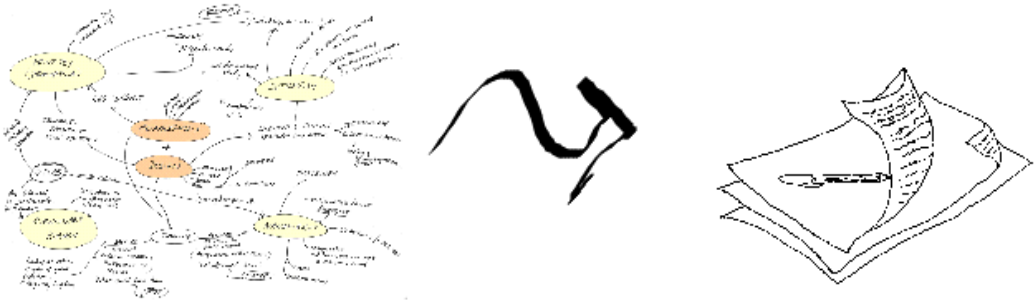

Figura 2: Brainstorm para o Documento Conceitual

Após a saída do documento conceitual na fase de concepção, o projeto é preparado para entrar na **fase de pré-elaboração**. Durante esta nova fase os recursos são alocados para cada equipe. Além disso, nesta fase é criado o GDS (*Game Design Specification*) o documento mais importante do projeto.

O documento de *game design* é responsável por detalhar todo o *game-play* a todas as equipes envolvidas na criação do jogo. Sem ele, o jogo não existe de forma plena, e se existir, grandes serão as possibilidades de ser um fracasso.

Durante a fase de pré-elaboração, algumas equipes terão de elaborar um documento técnico correspondente a cada área de desenvolvimento: *technical design document* (Barros, 2009). Seu objetivo é servir como um guia para cada atividade de cada equipe que será realizada na fase seguinte, a de produção. Nem todas as equipes devem elaborar um documento desse tipo; somente as equipes que irão definir especificações técnicas de produção.

Alguns exemplos de conteúdo de um documento técnico para a equipe de programação são: definir quais bibliotecas serão utilizadas na linguagem de

desenvolvimento do jogo, como se dará a padronização da codificação durante o desenvolvimento, qual repositório será utilizado para compartilhar códigos - ex. CVS, SVN (Bloom).

Para a equipe de artistas, o conteúdo de um documento técnico pode navegar entre as propostas: formato dos arquivos, resolução, extensão de arquivos suportados pelas *engines* utilizadas no desenvolvimento (Bloom).

Figura 3: Documento Conceitual para o Documento Técnico

Após reunir todas as informações necessárias no GDS para o desenvolvimento, inicia-se, então, a produção do jogo. Esta é uma etapa fundamental para o desenvolvimento deste trabalho, pois, é na elaboração do jogo que se exige maior interação entre as equipes de trabalho.

É na **fase de elaboração** que os programadores iniciam o desenvolvimento do código-fonte, enquanto que, ao mesmo tempo, outras equipes iniciam seus trabalhos. A equipe de artes dá início ao processo de criação de cenário, personagens, objetos que são responsáveis pela parte gráfica do jogo. Percebe-se então que, a partir daqui, a relação entre as duas equipes fica bastante homogênea.

É importante observar que para adestrar e coordenar esse desenvolvimento em paralelo e, ao mesmo tempo interligado, foi criado anteriormente o GDS. Através do documento game design as equipes poderão ser norteadas de maneira clara e objetiva durante a produção de seus artefatos.

Já na **fase de prototipagem**, surge a necessidade de realizar testes no jogo desenvolvido. De acordo com o grau dos problemas encontrados, o desenvolvimento do jogo volta para etapa de elaboração ou pré-elaboração. Se for um problema de grau

maior, como por exemplo, mudança de arquitetura, a etapa que se deve retornar é a de pré-elaboração. Caso seja uma falha menor que esteja apenas no código-fonte, ou uma simples mudança num determinado personagem construído pelos artistas, o desenvolvimento retorna para etapa de elaboração (Barros, 2009).

Os testes podem ser fechados (*alpha*) ou abertos (*beta*). No teste *alpha*, somente os desenvolvedores do jogo estão presentes. Porém, os usuários que serão beneficiados com o jogo poderão, de uma forma restrita imposta pelo desenvolvedor, utilizar o jogo. No teste *beta*, o produto, embora se encontre no estado de desenvolvimento, já está em um nível decente para utilização, mesmo que sem garantia alguma.

Recentemente a Level Up! selecionou cinco mil usuários para realizar testes fechados no jogo Combat Arms. Brevemente estariam entrando para a fase de testes abertos. A figura abaixo mostra um trecho da nota:

Figura 4: Nota dos testes realizados pela *Level Up!* no jogo *Combat Arms* (2010).,

Na fase pós-venda a empresa publicadora e a desenvolvedora passam a observar o mercado consumidor do produto lançado. Isso serve de estudo para

possíveis melhorias no futuro, ou até mesmo para base de desenvolvimento de novas versões ou expansões do jogo (Barros, 2009).

2.3. DISCIPLINAS CLÁSSICAS DE DESENVOLVIMENTO DE JOGOS

Como dito anteriormente, a fabricação de jogos conta com apoio de um grupo multidisciplinar. Das equipes abordadas anteriormente, este tópico discutirá superficialmente algumas das principais existentes no mercado: gerente de projetos, *game designer*, programador, artista, sonoplasta e engenheiro de teste.

Gerente de Projetos: Ele é responsável por acompanhar o desenvolvimento do jogo. Possui uma visão macro de como o projeto deve fluir: orçamento, cronograma, recursos humanos. Está presente em todas as etapas do desenvolvimento (Irish, 2005).

Game Designer: É o papel mais importante no desenvolvimento do jogo. Está presente na grande maioria das equipes de desenvolvimento de jogos, mesmo quando na há um profissional responsável somente por esta tarefa, algum outro membro da equipe executa de alguma forma essa função (Hendrick, 1998). É responsável por criar o enredo jogo em si, semelhante a um roteirista (Barros, 2009).

Programador: Responsável por toda lógica e produção de código-fonte na linguagem determinada pelos atributos do GDS. É ele que utiliza as ferramentas e APIs vinculadas a produção do jogo. Por exemplo, as APIs gráficas, as quais mais utilizadas atualmente são OpenGL e DirectX, e os ambientes de desenvolvimento (IDE). Porém, não só na codificação e na parte gráfica pode atuar um programador; existem outras áreas que são desenvolvidas por um programador: redes, simulação física, inteligência artificial, etc.

Artistas: Responsável por criar todos os elementos gráficos do jogo. É uma das maiores equipes e, dentro dela, existem vários especialistas. Assim como os programadores, os artistas podem atuar em diversas áreas para produzir um jogo: criação de personagens, estrutura do cenário, texturas, etc.

Sonoplasta: Semelhantemente ao artista, o músico tem o papel de criar todo o áudio do jogo. O profissionalismo é tão respeitado que existem alguns jogos que utilizaram orquestras para desenvolver o som do jogo (*Final Fantasy Series*). O papel pode ser dividido basicamente em duas áreas: efeitos e som ambiente (*soundtrack*).

Engenheiro de Testes: Outro papel fundamental para o desenvolvimento. O engenheiro de testes é responsável por garantir a qualidade final do produto. Além de realizar testes sobre o código, também passa por sua aprovação a usabilidade e jogabilidade (*game play*). Se não for um profissional capacitado, todo o processo de desenvolvimento pode ser comprometido.

Dentre esses papéis aqui definidos, a proposta deste trabalho é focar na relação de apenas dois deles: **programadores e artistas**. Portanto, os próximos dois capítulos serão voltados para um estudo individual deles.

3. PROCESSO DE DESENVOLVIMENTO DE CÓDIGO

O presente capítulo enfocará as atividades do programador no processo de desenvolvimento de código nos jogos atuais. Inicialmente será mostrada uma breve história do programador na participação na criação de jogos eletrônicos; posteriormente, serão abordadas algumas linguagens e APIs de programação para jogos

3.1. HISTÓRIA DO PROGRAMADOR NA FABRICAÇÃO DE JOGOS

Desde a concepção do desenvolvimento de jogos eletrônicos que o papel do programador se faz presente. O primeiro deles foi William Higinbotham, criador do *Tennis for Two* – 1958. William, sozinho, conseguiu captar a idéia do jogo eletrônico através de um osciloscópio analógico.

Embora a maioria dos jogos antigos fosse elaborada e desenvolvida por uma só pessoa, a ideia de divisão de trabalho em equipe (quando existia) no desenvolvimento de jogos já era utilizada há algum tempo. O segundo projeto elaborado, *Space War!*, foi criado por um grupo de estudantes do MIT. O principal desenvolvedor foi Steve Russell, juntamente com auxílio de seus colegas. Porém, todos eles sendo programadores de softwares.

Feito unicamente por programadores, os jogos eletrônicos possuíam extensas linhas de código no seu desenvolvimento. Não existia nenhuma prática ou técnica de programação, apenas código e mais código para programar.

Portanto, desenvolver jogos era uma prática puramente lógica e matemática. Não requeria muito esforço artístico por parte do desenvolvedor, apenas o mínimo necessário para representar os objetos do jogo (apenas algumas derivações de polígonos e/ou pintura de pixels). Como exemplo, algumas figuras abaixo demonstrarão a ideia de elaboração de alguns personagens.

Figura 5: Geometrias retalhadas compõem o Boxing – 1980 (Atari)

Figura 6: Desenho baseado em pintura de pixels no RiverRaid - 1982 (Atari)

É importante observar que mesmo sendo os jogos mais antigos, eles já possuíam uma pequena implementação de Inteligência Artificial, mas bastante eficiente para a época.

Porém, com o passar do tempo, os jogos foram "amadurecendo" bem como o processo de desenvolvimento dos programadores. Começou-se a criar algumas estruturas de programação bem como técnicas eficientes para a prática. Essas técnicas surgiram graças ao estudo do processo de desenvolvimento de softwares e hardwares. Pode-se dizer que, paralelamente ao avanço na busca de novas tecnologias e nas pesquisas de modelos de desenvolvimento, a produção de jogos pode, então, ganhar grandes dimensões.

Juntamente com o desenvolvimento de hardwares, também se desenvolveram ferramentas voltadas para o auxílio da construção dos jogos. As APIs direcionadas para os jogos e os motores (*engines*) serviram de alavanca na indústria dos games. Na próxima seção serão apresentadas algumas dessas ferramentas.

3.2. FERRAMENTAS PARA AUXÍLIO NO PROCESSO DE DESENVOLVIMENTO CÓDIGO PARA JOGOS.

Sendo uma parte restrita do conjunto de softwares em geral, o jogo eletrônico também pode ser programado em praticamente todas as linguagens de programação. Porém, a mais utilizada entre elas é a linguagem C/C++. Alguns fatores que contribuem bastante para o uso em grande escala desta linguagem são: a praticidade do uso, seu alto desempenho, a existência de um grande conjunto de APIs voltadas para o desenvolvimento de jogos para essa linguagem e a grande quantidade de tutoriais, livros e apostilas disponíveis em bibliotecas/internet voltadas para o desenvolvedor.

Juntamente com a linguagem de programação, o desenvolvedor pode se valer de um conjunto de ferramentas que o auxiliem para a produção do jogo. Por exemplo, podemos utilizar uma API gráfica para ajudar no desenvolvimento de jogos que envolvam elementos de processamento gráfico bastante complexos.

As APIs gráficas mais utilizadas no mercado são OpenGL e a Direct3D. Embora a OpenGL seja *opensource*, ainda assim, entre as duas, a mais utilizada é a Direct3D (LUZ, 2004). A Direct3D é uma parte das APIs que forma o tão conhecido DirectX e por ser desenvolvida pela Microsoft, só pode ser utilizada em ambientes tipo Windows. DirectX fornece um pacote de atualizações mais constantes e possui suporte à redes, execução de áudio, vídeo entre outras coisas ferramentas que possam auxiliar no desenvolvimento do jogo.

Entre alguns jogos desenvolvidos com DirectX tem-se o famoso *Grand Theft Auto*.

Figura 7: Grand Theft Auto: Episodes from Liberty City. (DirectX 9.0c)

Analisando o grupo dos jogos elaborados com OpenGL existe o conhecido *Half-Life*, produzido em 1998.

Figura 8: Half-Life. (OpenGL)

3.3. MOBILE, UMA DIFERENÇA PECULIAR

Saindo da plataforma *hard* (PCs e consoles) e passando para um grupo mais *soft*, a plataforma *mobile* oferece a programação de jogos voltada mais para a utilização da linguagem Java. Neste caso essa linguagem se sobrepõe à C/C++ devido a sua alta portabilidade entre os dispositivos; tudo isso graças à arquitetura da

linguagem que permite que suas aplicações sejam executadas sobre uma máquina virtual (JVM – *Java Virtual Machine*).

Para celular, também existe um conjunto de ferramentas que auxiliam o desenvolvimento do jogo. Por exemplo, a Game API disponibilizada a partir da versão 2.0 do MIDP (*Mobile Information Device Profile*), fornece um apoio exemplar para o desenvolvedor de jogos para celular utilizando a API java (Mahmoud, 2005). É uma API muito utilizada entre as empresas que desenvolvem jogos casuais para celular.

Porém, dentro dessa esfera de desenvolvimento para dispositivos móveis, existe um grande mercado em ascensão que está mexendo com o ramo das empresas: o desenvolvimento para aplicações voltadas para *mobiles* da Apple (*iPhone*, *iPad* & *iPod*).

Existem diversas *engines* voltadas para o desenvolvimento de jogos 3D para *iPhone*. Como por exemplo, a Unity possui grandes ferramentas de auxílio para codificação dos jogos para *iPhone* e seu preço varia de US$300.00 até US$1,500.00. A GarageGames também possui uma *engine* 2D para *iPhone* (iTorque 2D) que custa US$750.00.

4. PROCESSO DE DESENVOLVIMENTO DE ARTE

Este capítulo mostrará as atividades do artista gráfico quanto no processo de desenvolvimento de arte durante a produção dos jogos digitais. Inicialmente será apresentado o conceito do artista gráfico no mercado de jogos; depois, serão abordadas as ferramentas que auxiliam o trabalho do artista; e, por fim, a descrição das etapas do processo de desenvolvimento da arte do jogo.

4.1. OS “ARTISTAS” DE JOGOS: UM CONCEITO NÃO FORMADO

O conceito de arte vem muito antes da idéia dos jogos eletrônicos. Define-se arte como sendo a expressão de um ideal de beleza nas obras humanas.

Transpondo para uma era mais remota, podemos observar arte desde o tempo paleolítico da humanidade onde o ser humano utilizava-se de instrumentos rudimentares para produzir seus desenhos como forma de simples comunicação.

E a aplicação da arte é exatamente para esse feito: comunicar-se. No ramo da computação gráfica não é diferente. Nos jogos, a arte é desenvolvida para que aja uma comunicação maior entre o usuário e o jogo (aplicação), chamando de interação usuário-máquina.

Porém, Auriea Harvey afirma que a arte gráfica não pode ser chamada de arte, e que, consequentemente, o artista gráfico não pode ser chamado de artista (Pratt, 2010). Ainda segundo Harvey, a produção de arte está morta. Após a ascensão do capitalismo, a arte moderna cedeu às formas restritivas de governo, bem como a ambição pelo dinheiro as suas origens emotivas que transpareciam o desejo do artista de expressar seus sentimentos como comunicação.

- O que seria o artista gráfico então?

Na verdade o termo “artista gráfico” surgiu com a necessidade de mercado. Obviamente não se pode comparar um especialista em modelagem 3D com Da Vinci ou Michelangelo. O artista gráfico é um termo genérico para expressar a definição de um profissional que utiliza o conceito ou uma idéia e a transforma em imagens e textos como forma de comunicação.

Assim como o homem das cavernas possuíam seus instrumentos, os artistas gráficos de jogos eletrônicos também possuem algumas ferramentas essenciais para o desenvolvimento de sua arte. Abaixo, serão abordadas algumas dessas principais ferramentas para a produção do jogo.

4.2. FERRAMENTAS PARA O AUXÍLIO NO PROCESSO DE DESENVOLVIMENTO DE ARTE

O artista é responsável por toda parte de criação de arte 2D e 3D nos jogos. Elementos do tipo cenários, personagens, objetos, *sprites*, animações, etc; fazem parte do grupo de tarefas que o artista deve desenvolver.

Antes de definir quais ferramentas são utilizadas pelos artistas, é necessário separar duas maneiras de se produzir arte na computação gráfica: desenvolvimento no plano 2D e elaboração de modelagem 3D.

Plano (ou 2D)

O conceito de arte bidimensional não é contemporâneo: as pinturas nas cavernas da idade da pedra e os quadros da idade medieval mostram que a humanidade já trabalhava com a idéia de plano há bastante tempo.

Matematicamente, um plano é um objeto geométrico definido em duas dimensões (2D). Tradicionalmente, os artistas gráficos computacionais trabalham no ambiente em que é definido por retas perpendiculares x e y, ou seja, os elementos só possuem duas dimensões: largura e altura.

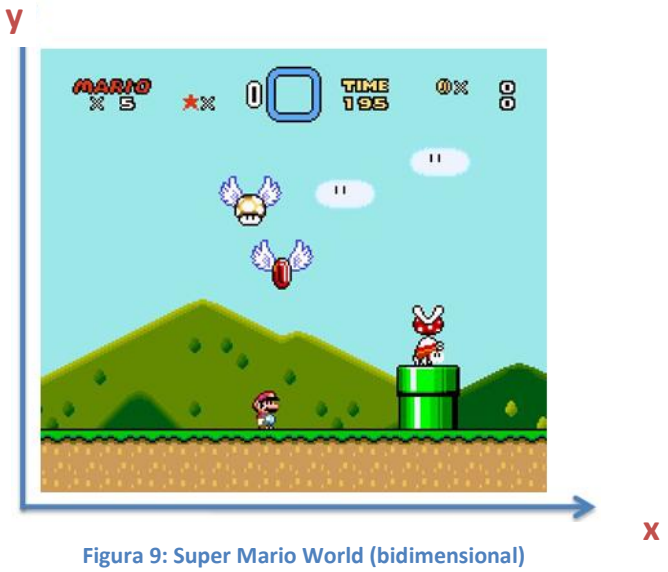

Figura 9: Super Mario World (bidimensional)

Modelagem 3D

Analogamente à arte 2D, a modelagem 3D também teve suas raízes em tempos antigos. Podem-se relatar aqui as famosas esculturas greco-romanas no momento em que grandes nomes da História eram lapidados com perfeição no gesso ou algum outro tipo de material.

Semelhantemente ocorre na computação gráfica: o artista projeta uma modelagem de um objeto no computador "esculpindo" da forma que lhe convier o seu trabalho.

Modelagem 3D, ou modelagem tridimensional, é a área da computação gráfica que tem o objetivo de criar elementos em três dimensões, podendo ser classificados em: renderização (estático) e animação (dinâmico), com ou sem participação do usuário. No caso de jogos, obviamente, possui uma grande interatividade do usuário.

Figura 10: Super Mario Galaxy (tridimensional)

Ferramentas de trabalho

Após uma breve descrição das maneiras de se produzir arte em computação gráfica pelos artistas, pode-se descrever algumas ferramentas de trabalho que são bastante utilizadas por esses profissionais.

Os artistas, diferentemente dos programadores – que mantém geralmente algumas poucas ferramentas de desenvolvimento devido à escolha da linguagem de programação –, possuem um imenso leque de ferramentas que podem ser utilizadas em conjunto para o desenvolvimento de arte.

Bidimensionais:

Na execução de trabalhos bidimensionais, existem artistas que utilizam ferramentas desde Microsoft Paint (ferramenta considerada mais simples) até o Adobe Fireworks ou Adobe Photoshop (ferramenta de maior complexidade). Cada ferramenta possui suas vantagens e desvantagens para o determinado tipo de trabalho que será executado. O artista responsável tem a obrigação de conhecer o funcionamento dessas ferramentas, bem como suas características para melhor execução de seu trabalho.

Ainda para o desenvolvimento bidimensional, existe uma particularidade para as tarefas relacionadas aos jogos de celulares. Para o ambiente *mobile* o surgimento de imagens tipo SVG (*Scalar Vector Graphics*) trouxe novas possibilidades de criação de interfaces poderosas com animações e efeitos gráficos conhecidos de forma que a aplicação não fique tão “pesada”.

Uma das qualidades do formato SVG é que seu tamanho, comparado com outros formatos, é bastante reduzido, e ainda possui características importantes como controle de qualidade. Mesmo quando há redimensionamento, a imagem não perde qualidade. Trata-se de uma linguagem XML que descreve de forma vetorial desenhos e gráficos bidimensionais, quer de forma estática, quer dinâmica ou animada (OGLIARI).

Figura 11: Comparação entre bitmap e SVG (bitmap à esquerda e SVG à direita)

Tridimensionais:

Tratando de desenvolvimento para ambientes 3D, existe outra vasta lista de ferramentas que auxiliam os artistas no que diz respeito à elaboração de arte tridimensional para jogos.

Para gerar elementos 3D as ferramentas mais comuns são o 3D Studio Max, o Maya e o Blender. Não existe, entre elas, a melhor e a pior; o que pode existir é qual se adapta com melhor desempenho para uma determinada equipe de trabalho e qual tipo de trabalho a ser realizado. Obviamente se um artista for contratado por uma empresa de jogos que só utilize o Maya, não faria muito sentido ele usar, sozinho, o 3D Studio Max.

As ferramentas possuem atualizações recentes e fóruns de discussão muito concorridos; fatores que contribuem para a escolha de uma delas para realizar o trabalho.

Na próxima seção, será abordada a descrição dos processos de desenvolvimento de arte auxiliados por essas ferramentas.

4.3. ETAPAS DO PROCESSO DE DESENVOLVIMENTO

As etapas do processo de desenvolvimento do jogo para o artista devem estar bem definidas no *Game Design Document*. O artista deve obedecer às regras impostas pelo designer e trabalhar da melhor forma possível para que haja uma harmonia entre o seu trabalho individual e o desenvolvimento do projeto como um todo.

Como todo processo, as entradas a são analisadas e as saídas maculadas a partir de um conjunto de informações passadas. Para a produção de arte, basicamente existem duas etapas no processo de desenvolvimento a serem trabalhadas pelo artista: concepção (entrada) e modelagem (saída).

Concepção

Durante a fase de concepção da arte, existe uma grande quantidade de troca de informações entre os membros da equipe. É nesta fase que há uma maior interação entre os artistas e o game designer.

A concepção é necessária para que os membros da equipe – tantos os artistas, como as outras equipes – tenham uma breve noção do que será desenvolvido na etapa de modelagem. Serve como uma maneira de comunicação primária entre o artista e o game designer; um primeiro contato entre as idéias.

Na fase de concepção, o artista desenha em uma folha de papel todos os elementos do jogo que serão modelados. Essa forma de trabalho é bastante útil e vantajosa.

Uma das grandes necessidades é a mudança do protótipo. Como vantagem, essas alterações na fase de concepção geram um custo infinitamente menor caso fosse realizada na fase de modelagem.

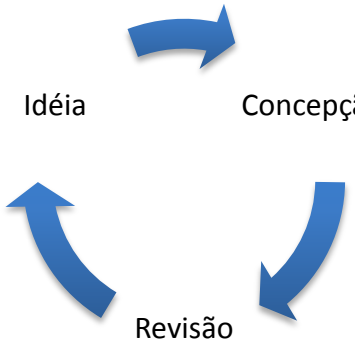

Figura 12: Ciclo de atividade do artista (concepção)

A idéia passada pelo game designer gera a concepção da arte. Depois de terminado o trabalho na concepção, o artista levará o resultado para a equipe realizar a revisão; esta pode, ou não, gerar outra idéia.

Esta é a forma mais comum de conceber um protótipo e, a partir de então, dar início a etapa de modelagem.

Modelagem (elaboração)

A construção do elemento prototipado em sua versão final ocorre na etapa de modelagem. A partir dos desenhos, os artistas inspiram-se e desenvolvem uma arte gráfica no computador. Ou seja, é a passagem da idéia que estava concebida no papel para o mundo digital.

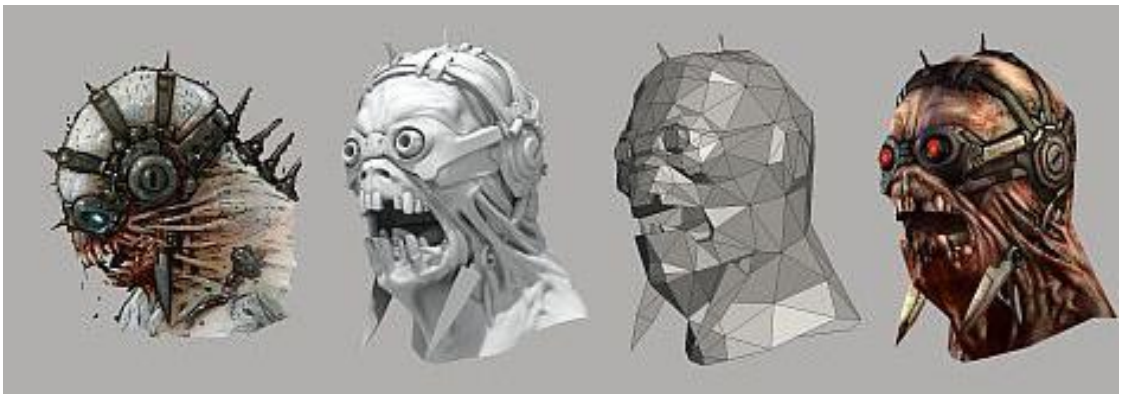

Figura 13: Concepção -> Modelagem

Após esse estado, pode-se dizer que o personagem está pronto; ou seja, é possível obter a exatidão de como o personagem será no jogo.

Tendo a introdução do papel do programador e do artista dentro da esfera de ambiente de desenvolvimento de jogos eletrônicos, o próximo capítulo mostrará os aspectos relacionais entre essas duas equipes.

5. CARACTERÍSITCAS RELACIONAIS ENTRE OS PROGRAMADORES E OS ARTISTAS

Será abordado neste capítulo o ponto mais importante deste trabalho: o relacionamento entre os programadores e os artistas dentro do ambiente de desenvolvimento de jogos. Focando a objetividade proposta, o capítulo deverá responder as seguintes perguntas:

- Que informações são trocadas entre os artistas e os programadores?
- Em que momento há troca das informações?
- Como é feita a troca dessas informações?
- Essa troca de informações é suficiente para o ambiente de trabalho?
- Existe dependência entre as equipes?
- Que ferramentas podem auxiliar essa relação?

Como resultado das pesquisas de campo realizadas dentro das empresas de desenvolvimento de jogos eletrônicos no centro do Recife, Pernambuco, tais como a Meantime e Manifesto, pôde-se chegar a um resultado interessante.

5.1. QUE INFORMAÇÕES SÃO TROCADAS ENTRE OS ARTISTAS E OS PROGRAMADORES?

Esta pergunta não é tão simples de responder, pois diversas variáveis estão em questão para ambos os lados; tanto aos artistas, quanto para os programadores. É necessário identificar quais elementos correspondem ao fluxo **artistas -> programadores** e quais outros correspondem ao fluxo **programadores -> artistas**.

5.1.1. DOS ARTISTAS AOS PROGRAMADORES

O resultado da pesquisa mostrou relevante a dependência de trabalho de ambas as equipes. Dentro desta dependência, notou-se uma característica bastante forte para a equipe de arte: existe uma tendência inconsciente de que a equipe de arte funciona como uma empresa subcontratada dentro da equipe de projeto de desenvolvimento. Isso é tão verdade que algumas empresas de criação de jogos eletrônicos utilizam serviços terceirizados para desenvolver sua arte (*freelancers*).

Tal atitude não chega a ser um problema, porém enfraquece um pouco a comunicação entre os artistas contratados (quando do caso) e os programadores. E gera um pouco de sentimento de inferioridade por parte dos artistas, o que pode ser um pouco preocupante para a equipe.

Um dos problemas observado durante a pesquisa, fora o fato de que as informações geradas por parte dos programadores são, geralmente, muito técnicas e que, às vezes, gera certo desconforto dentro da comunicação entre as equipes.

Para solucionar esse problema, o próprio mercado teve a necessidade de se criar o cargo do *Technical Artist*. Ele atua como uma ponte entre os programadores e os artistas e tem a responsabilidade de garantir a integração dos recursos artísticos sem sacrificar a visão artística global ou exceder os limites técnicos impostos pelos programadores.

A principal área de atuação do *Technical Artist* inclui a criação e manutenção do fluxo de produção de arte, bem como tomar decisões sobre quais pacotes e ferramentas de trabalho se deve usar.

O papel é relativamente novo na indústria de jogos, mas ele está se tornando cada vez mais importante devido ao fato de que os consoles e hardwares para PC estão se tornando muito complexos. Portanto, o *Technical Artist* saberá quais informações devem ser transmitidas dos artistas para os programadores.

A principal informação que o artista envia para o programador é a arte em si. A arte é o produto (saída) do trabalho realizado a partir das restrições e concepções (entradas) requeridas pelo programador e game designer.

Porém a arte final nem sempre está pronta quando o programador a precisa. Para resolver esse problema, os artistas criam uma prototipagem iterativa do conteúdo do jogo (*assets* gráficos) (MOURA, 2009). Os *assets* asseguram a equipe de artistas uma idéia primária de produto requisitado pela equipe de programadores e *game designers*.

Até mesmo para evitar perda de produção e tempo, alguns elementos dos *assets* – dependendo do nível de acabamento – podem ser utilizados para a versão final do produto da arte.

Abaixo, segue um exemplo de como um *asset* pode ajudar o programador no desenvolvimento de um *menu* de jogo.

Figura 14: Versão inicial do menu principal (Manifesto Games Studio - Animal Shards)

Figura 15: Última versão do menu principal (Manifesto Games Studio - Animal Shards)

5.1.2. DOS PROGRAMADORES AOS ARTISTAS

Por parte dos programadores a pesquisa revelou que as informações transmitidas aos artistas são de cunho técnico e específico. Informações como: resolução, tamanho, playground, taxa de bits de ima imagem, limite de processamento, API gráfica utilizada, entre outras, são passadas para os artistas se basearem.

Os programadores por sua vez, dificilmente produzem algum resultado para os artistas. Existe mais uma cobrança do que é pedido. Por exemplo, em jogos tipo flash, em que a produção é dividida em frames, quando a equipe de programação está manipulando um determinado frame, existe um contato prévio com a equipe de artes para que, quando a arte for integrada no flash, já siga um determinado padrão pela equipe de programadores definida que não precise ser alterado futuramente.

Existem casos em que alguns formatos de arte não possuem suporte para determinadas aplicações utilizadas pela equipe de programadores. Para isso, os programadores devem comunicar (caso não exista registrado) quais serão os formatos e dimensões utilizados pelos artistas.

Essas informações trocadas são de extrema importância, pois evitará surpresas desagradáveis durante o processo de desenvolvimento dos jogos. Portanto, é necessário também definir os momentos de troca de informações entre as equipes.

5.2. EM QUE MOMENTO HÁ A TROCA DE INFORMAÇÕES?

Sendo bastante objetivo quanto a resposta da pergunta, em todo o processo de desenvolvimento do jogo existirá a troca das informações. Porém, existem alguns momentos que devem ser detalhados para um melhor entendimento.

Durante a fase de concepção, há pouca interatividade entre os artistas e os programadores. Esse fato ocorre porque o projeto em si ainda está nascendo e, para esclarecer, há maior relação das equipes com o game designer e não entre elas.

Porém, durante a fase de elaboração a interação entre os programadores e os artistas trona-se mais sólida e necessária. Neste momento, os artistas se apressam em fazer *mockups* (maquetes) para dar uma melhor visualização de como o jogo deverá ser enquanto que os programadores dão início ao desenvolvimento de código.

No mundo ideal os programadores começariam apenas quando todos os *assets* estivessem prontos; pois assim obter-se-ia um melhor resultado durante o desenvolvimento. Porém, na prática, muitas vezes os programadores utilizam imagens horripilantes distribuídas pelos artistas ou até mesmo caixas sólidas para representação de elementos. Os *assets* são inseridos à medida que são produzidos.

Ao final da fase de elaboração, onde os *assets* e toda a parte gráfica em geral estão prontos, ocorre grande troca de informações entre as equipes. Neste momento serão discutidos os detalhes finais do desenvolvimento do jogo.

Muitas vezes, infelizmente, há ainda algumas coisas para serem alteradas, fator que agrega mais atraso da entrega do projeto e despende mais recursos das equipes.

O que se pode observar, é que a maior parte da interação surge da necessidade de se fazer alterações nas imagens, realizar otimizações no programa ou corrigir animações que tenham sido notadas com alguma falha durante a integração com o projeto.

Na próxima seção, será abordado como é conduzida essa relação entre as equipes.

5.3. COMO É FEITA A TROCA DE INFORMAÇÕES?

As formas como são trocadas as informações entre a equipe dos programadores e a equipe dos artistas podem variar bastante entre as empresas de jogos. Isto se deve ao fato de cada empresa escolher livremente qual estratégia de desenvolvimento usar.

Dentro da empresa da Manifesto, por exemplo, opta-se pelo uso de *stand-up meetings* diários. Portanto, as informações são trocadas pessoalmente, de forma falada. Nos *reviews* é mostrado o que está sendo feito; dessa forma, sugere-se que os membros de equipes diversas possam opinar sobre o trabalho do companheiro. É uma forma interessante de promover maior interação entre as equipes, bem como enriquecimento do trabalho.

Em outras empresas, principalmente as que possuem grande número de desenvolvedores, é necessário um controle maior. Neste caso, vale a pena o uso de ferramentas de controle, bem como aplicação de processos regulamentares. Tais ferramentas e processos podem gerar documentos que registrem o que fora discutido, para que nada seja esquecido ou adulterado.

No caso de grandes empresas, é também estudada a frequência que essas informações são trocadas entre as equipes. Registros esporádicos podem gerar desinteresse e esquecimento no trabalho da equipe alheia. O desinteresse é revelado devido a criação de expectativa que nem sempre corresponde ao trabalho que efetivamente foi criado.

Por exemplo, um determinado artista cria um protótipo de um personagem requisitado. Neste momento o programador gera uma expectativa do que será criado pelo artista. Depois de um mês de trabalho sem algum tipo de comunicação, o artista termina com o personagem e apresenta o elemento que não condiz com o que o programador imaginava.

Para evitar esse tipo de problema, é necessária uma maior frequencia de entrega de trabalho às equipes envolvidas.

5.4. ESSA TROCA DE INFORMAÇÕES É SUFICIENTE PARA O DESENVOLVIMENTO DO TRABALHO?

Dependendo de como é utilizada a troca de informações (seção anterior), tais informações podem não ser suficiente para o desenvolvimento do trabalho de ambas

as equipes. No caso da Manifesto, a empresa afirma que a forma adotada é a melhor para a empresa. Pois, neste caso, todos os elementos das equipes sabem o que os outros estão fazendo (ou planejando fazer) naquele dia de trabalho, visto que se utilizam de *meetings* diários.

Já no caso de empresas que utilizam os processos e ferramentas como documentação em relação à comunicação das equipes, é possível que exista certo nível de carência de na troca de informações, devido à burocracia e à formalidade nos processos. Dessa forma, nem sempre os participantes conseguem expressar de forma clara e objetiva o necessário. Surge, portanto, uma pequena dificuldade de gerenciar o entendimento dos membros das equipes de desenvolvimento.

Uma solução para viabilizar esse problema é diminuir o nível de burocracia entre as equipes (até mesmo aproximando-as fisicamente umas com as outras) e orientar no sentido de que é necessário o mínimo de entendimento, porém evitar o formalismo nos processos de registro de informação.

5.5. EXISTE DEPENDÊNCIA NOS TRABALHOS ENTRE OS ARTISTAS E OS PROGRAMADORES?

Como foi visto, existe uma dependência muito grande por parte dos programadores aos trabalhos dos artistas. Para que o programador tenha uma idéia completa do jogo, é necessário que se tenha em mãos a arte (*mockups* ou *assets*) pronta. Porém, nem sempre isso está definido.

Isso pode gerar um grande problema de “gargalo” entre as equipes. O efeito gargalo é caracterizado como sendo o nível de ociosidade do grupo. Como os artistas estão trabalhando em suas obras, às vezes, quando não há mais o que codificar, a equipe de programadores pode ficar com tempo ocioso esperando que a arte fique pronta. Só de horas perdidas de produção, já é possível perceber a falta com que esse problema traz para a equipe de desenvolvimento.

Para evitar esse tipo de problema existem algumas soluções; dentre elas, algumas relevantes e outras paliativas.

A solução paliativa consiste em alocar os integrantes das equipes que estão ociosos em outro projeto. Essa tomada de decisão irá resolver o problema da ociosidade, porém existem algumas outras desvantagens que devem ser observadas:

- **Desestímulo:** o indivíduo perde o foco do trabalho que está realizando quando é transferido de um projeto para outro.
- **Aprendizado:** o trabalhador precisará de um tempo para entender o projeto para qual ele será realocado.

A solução interessante para este tipo de problema está em realocar recursos para a equipe de artistas começarem a desenvolver a arte juntamente com um *Technical Artist* antes da fase de elaboração, ou seja, na fase de concepção. Neste caso, quando os programadores iniciassem a desenvolver seus códigos, toda a produção primária dos artistas estaria pronta. As vantagens, além de facilitar a vida profissional dos programadores, são manter o foco das equipes no projeto e dispor de mais tempo de trabalho para a equipe dos artistas na fase de elaboração, uma vez que os protótipos já estariam prontos. A desvantagem é ter um custo maior com mais um membro na equipe, o *Technical Artist*.

5.6. QUE FERRAMENTAS PODEM AXILIAR ESSA RELAÇÃO?

As ferramentas que podem auxiliar no processo de interação entre as equipes são as conhecidas atualmente pelo mercado. Porém, a pesquisa realizada na Meantime constatou que aplicações de gerenciamento de *bug tracker* como Mantis, são complicadas e de difícil entendimento por parte dos artistas.

Porém, como é de extrema importância que todos utilizem esse tipo de ferramenta, foi sugerido pela equipe de artistas durante a pesquisa, que a aplicação Basecamp da empresa 37signals fosse utilizada, pois é superior e menos complexa que

Mantis ou Trac. Além de servir como um *bug tracker*, serve também como um gerenciador de projetos. Sua desvantagem é por ser uma ferramenta paga.

6. CONCLUSÃO

Este trabalho teve o objetivo de apresentar um estudo sobre a relação entre os programadores e os artistas no ambiente de desenvolvimento de jogos. O desenvolvimento foi criado através da identificação da falta de pesquisas e estudos a respeito deste relacionamento.

6.1. GUIA DE PESQUISA

Após uma breve introdução envolvendo a concepção de desenvolvimento de jogos atuais, foram mostrados individualmente os papéis de cada uma das equipes estudadas: **programadores e artistas**.

Ao final, foi apresentada uma análise sobre a pesquisa realizada nas empresas de desenvolvimento de jogos no Recife. Esse estudo revelou o quão importante é conhecer o relacionamento entre equipes de projetos, não só entre artistas e programadores, mas entre uma vasta combinação de todas as equipes em si.

6.2. TRABALHOS FUTUROS

Como foi dito, existe muito a ser estudado. Sobre o tema deste trabalho, existem alguns assuntos que poderia servir de base para futuras pesquisas de graduação ou pós-graduação.

O cargo pouco utilizado, mas que revelou de extremo interesse na pesquisa fora a criação do *Technical Artist* nas empresas que desenvolvem jogos eletrônicos. Com certeza existem muitos recursos a serem explorados nesse sentido, fortificando mais ainda a relação entre artistas e programadores.

A utilização de *stand-up meetings* diários durante a elaboração de projetos mostrou ser uma boa prática de gerenciamento. Conversas diárias eliminam a ausência de informações e aumenta a integridade entre as equipes.

REFERÊNCIAS BIBLIOGRÁFICAS

ALVES, C.; RAMALHO, G.; DAMASCENO, A.; 2007. Challenges for Mobile Games development: The Meantime Case Study. Requirements Engineering Conference, 15th IEEE International.

BARROS, R.; 2009. Investigando equipes multidisciplinares em projetos de jogos: uma pesquisa qualitativa.

BLOOM, P.; MOORE, J.; OTLOWSKI, C.; ZAXZKIEWICZ, P.; CALVERT, V. L.; Technical Design Document for Puzzle Kingdom. Acesso em 05/2010.
Disponível em:
http://www.pishoque.net/soulriders/portfolio/PK-TDD.doc

FLYNT, J., SALEM, O. 2005. Software Engineering for Game Developers. Thomson Course Technology. Premier Press.

GANDRA, A.; 2009. Mercado de jogos eletrônicos cresce no Brasil apesar da crise Web. Acesso em 05/2010.
Disponível em:
http://www.inovacaotecnologica.com.br/noticias/noticia.php?artigo=mercado-de-jogos-eletronicos-cresce-no-brasil-apesar-da-crise&id=

HENDRICK, A.; 1998. Fev. Hiring Game Designers. Game Developer Magazine.

HAYES, J.; 2008. The Code/Art Divide: How Technical Artists Bridge The Gap.
Acesso em 06/2010.
Disponível em:
http://www.gamasutra.com/view/feature/1651/the_codeart_divide_how_technical_.php

LUZ, M. H. G. C.; 2004. Desenvolvimento de Jogos de Computador. UNIFEI

MAHMOUD, Q.; 2005. Getting Started With the MIDP 2.0 Game API
Acesso em 06/2010.
Disponível em:
http://developers.sun.com/mobility/midp/articles/gameapi/

MOURA, L. J. B.; BREYER, F. B.; GIULIA, C. A. L.; FILHO, V. V.; ALBUQUERQUE, M. T. C. F.; 2009. Avaliação de Assets Gráficos para produção de jogos digitais.

IRISH, D.; 2005. The Game Producer's Handbook. Thomson Course Technology. Premier Press.

OGLIARI, R. S.; Imagens SVG e Java ME.
Acesso em 06/2010.
Disponível em:
http://www.javanoroeste.com.br/javanoroeste/novo/artigos/svg_javame.html

PRATT, C. J.; The Art History... Of Games? Games As Art May Be A Lost Cause.
Acesso em 05/2010.
Disponível em:
41http://www.gamasutra.com/view/news/27133/The_Art_History_Of_Games_Games_As_Art_May_Be_A_Lost_Cause.php

QUEIROZ, H.; 2009. Mercado Nacional de games deverá crescer em 2009 e 2010. Web.
Acesso em 05/2010.
Disponível em:
41http://www.forumpcs.com.br/noticia.php?b=256915

RAMALHO, G.; Desenvolvimento de jogos. Apresentação. Acesso em 03/2010.
Disponível em:
http://www.cin.ufpe.br/~game/aulas/desenv.zip

RAMOS, J.; 2008. Uma breve História dos Jogos. Web. Acesso em 04/2010. Disponível em: http://demonweb.wordpress.com/2008/06/18/uma-breve-historia-dos-jogos/

SLOPER, T.; 2005. Game Production and Project Management. Introduction to Game Development. Cengage Learning, 809-854.