Análise de Tecnologias para a Implementação de Jogos Web

Resumo

Este artigo aborda as questões mais relevantes na produção de jogos *single-players* baseados na Web. Apresenta as razões que motivam o estudo desse tipo de aplicação, indica os requisitos mínimos de tecnologia necessários para sua elaboração, relata as características daquelas que são consideradas as tecnologias emergentes para a produção desse tipo de software e propõe um conjunto de estratégias para superar os problemas encontrados no uso de um pacote composto por três delas (Extensible 3D (X3D) com Xj3D, Flash e JavaScript / ECMAScript).

Palavras-chave: Jogos baseados na Web, Extensible 3D, Xj3D, Shockwave Flash

Contato com os autores:

1. Introdução

Em razão de políticas governamentais, redução dos preços de componentes de hardware e de software, mudanças culturais e também em função da crescente necessidade de inclusão digital, o acesso a internet vem crescendo em todo o mundo [Magalhães 2006]. Com isso aplicações tradicionais (aquelas que rodam localmente) vêm se tornando cada vez menos atrativas, dando espaço a aplicações mais poderosas baseadas na Web. [Fraternali 1999]

A maior liberdade para a utilização das aplicações dada aos usuários é uma das principais vantagens trazidas por essas novas aplicações. Vez que para o seu uso não é preciso instalar (no sentido mais estrito do conceito) softwares, bibliotecas e demais elementos necessários para a sua execução. Tal benefício se torna mais evidente ao se considerar que os usuários podem utilizar as aplicações baseadas na Web de qualquer lugar do mundo a partir de uma conexão Internet.

Para os que implementam software, o benefício se dá pela adoção de uma única plataforma de desenvolvimento, o ambiente Web. Situação que torna desnecessária a implementação de infindáveis versões para as mais variadas plataformas [Lau and Kunii 2003]. Sem o inconveniente das múltiplas versões, é possível alcançar diferentes populações de usuários, sendo possível, inclusive, promover atualizações das aplicações simultaneamente para todos elas, algo impensável para as aplicações tradicionais.

1.1. Aplicações gráficas baseadas na Web

No contexto da mudança do ambiente desktop para a Web [Lau and Kunii 2003], o aumento na disponibilidade e a considerável redução dos preços dos hardwares para a aceleração de gráficos 3D em PC's são indicados como aspectos fomentadores da proliferação de um tipo promissor de aplicação: as aplicações gráficas baseadas na Web.

Essas aplicações, ao contrário das suas equivalentes tradicionais, trabalham em um ambiente distribuído, a Internet e para isso possuem a arquitetura cliente-servidor. Elas preferencialmente respeitam o modo de exploração típico da Internet, no qual o usuário recebe o conteúdo instantaneamente após clicar em um link. [Fielding and Taylor 2002]

A gama de áreas em que podem ser empregadas é bastante vasta, sendo em geral, equivalente a das que rodam localmente. Engenharias, medicina, administração e física com suas ferramentas de desenho auxiliado por computador (Computer-Aided Design - CAD), aplicações visuais de diagnóstico, esquemas de representação de dados financeiros e simulações de fenômenos naturais são, respectivamente, ótimos exemplos de domínios em que se pode utilizar com sucesso aplicações gráficas baseadas na Web. [Lehman 1986]

Dos campos que podem fazer uso dessas aplicações, o entretenimento digital é talvez o que pode receber o maior impacto pelo emprego dessa nova tecnologia, pois os elementos mínimos necessários para a criação de jogos podem ser encontrados nessas aplicações, tais como, interatividade, sonorização, animações vetoriais compostas por elementos geométricos texturizados, etc.

1.2. Estudo de caso

Em função da necessidade de se avaliar mais criteriosamente o potencial de uso dessa nova tecnologia para a produção de jogos na Web, uma pesquisa, em curso, objetiva identificar os problemas existentes e propor estratégias para solucioná-los.

O método de pesquisa utilizado é do estudo de caso e o objeto analisado é o Edugraph, jogo educacional single-player concebido no âmbito do projeto de pesquisa financiado pelo CNPq, LudicLearning. [Projeto 2006]

Voltado ao ensino de conceitos básicos de computação gráfica para alunos de graduação dos cursos de Engenharia, Arquitetura e Design, este jogo é

composto por uma animação introdutória e cinco fases, sendo três bidimensionais e duas tridimensionais.

Seu enredo constitui-se da saga do Edu, personagem estilizado que em um mundo fictício se vê preso em um labirinto e do qual só consegue escapar após superar uma série de desafios que constituem as 5 fases do jogo. Os conteúdos educacionais explorados no Edugraph são utilizados implicitamente pelo jogador, quando da resolução desses desafios.

Atualmente o seu desenvolvimento se concentra na elaboração das fases tridimensionais, bem como na integração delas com a animação introdutória e as três fases bidimensionais (em Adobe Macromedia Shockwave Flash), as quais já foram implementadas.

Esse trabalho de implementação e pesquisa, a exemplo das informações obtidas na elaboração das fases iniciais já implementadas, é a fonte das análises e sugestões contidas nesse artigo.

2. Requisitos de tecnologia

Dentre as tecnologias existentes atualmente para a criação de aplicações Web, apenas um conjunto relativamente pequeno possui os requisitos mínimos necessários para o desenvolvimento de jogos.

Em termos da produção de conteúdo, estes requisitos compreendem o suporte a manipulação de diversas mídias (texto, imagens, sons, animações) [Dalgarno 2001]. Em termos da implementação, os recursos requeridos são: a criação e manipulação de gráficos vetoriais, vasta capacidade de controle por programação, criação de estruturas de dados complexas, obtenção de dados de dispositivos de entrada convencionais (teclado, mouse, etc), detecção de colisão entre elementos, suporte a programação por manipulação de eventos e integração com outras tecnologias. Em termos da operação no ambiente Web, os requisitos são: independência de navegador e de sistema operacional, suporte interno (built-in) ou formas de instalação automática nos navegadores mais populares, e formas otimizadas de compactação dos dados geométricos e/ou de criação de mecanismos de gerenciamento de fluxos de dados. Em termos de aspectos comerciais e ideológicos, os requisitos são: baixo custo de utilização, padronização internacional, código aberto, vasta documentação e ferramentas de criação, edição e publicação.

2.1. Topologias

A organização da aplicação, a forma como ela é executada e como é realizado o processamento que transporta o resultado visual da manipulação do ambiente gráfico ao usuário, são dois fatores relevantes na elaboração de aplicações gráficas baseadas na Web. A influência destes fatores na produção desse tipo de software é tão importante quanto o respeito aos requisitos anteriormente citados. A opção por uma determinada topologia influencia tanto a forma como será organizado e implementado o código, quanto à escolha da tecnologia a ser utilizada.

O fato das aplicações gráficas baseadas na Web possuírem a arquitetura cliente-servidor e de que a Internet é uma rede composta por *hosts* sob os quais não se pode fazer qualquer asserção sobre capacidade de processamento e memória, contribui por tornar crítica a opção por uma determinada topologia.

A utilização de uma topologia com execução centrada no servidor, por exemplo, acarretará, provavelmente, no uso intenso de banda de rede, em problemas com interatividade causada pela latência na comunicação pela rede e na necessidade de configurações de hardware com alto poder de processamento para o servidor. Note-se que para a implementação de aplicações com essa topologia, as tecnologias utilizadas devem permitir a criação de servidores capazes de manipularem ambientes gráficos, processar dados de entrada enviados pelo cliente, gerar e enviar na forma de imagens compactadas ao cliente pelo menos 12 quadros por segundo (taxa de quadros mínima para a apresentação de animações contínuas). Também devem permitir a criação de clientes que possam ser executados em navegadores Web, os quais devem ser capazes de obter e enviar dados de dispositivos de entrada, além de exibir as imagens produzidas pelo servidor.

Como vantagens, esta topologia permite a utilização de um cliente que necessite de uma configuração de hardware com capacidades de processamento e memória mínimas e facilita a obtenção da independência de plataforma.

A topologia centrada no cliente implicará, possivelmente, em uma maior latência para o início da execução da aplicação, em problemas para a promoção de independência de plataforma e na necessidade de configurações de hardware com alto poder de processamento para a execução do cliente.

As vantagens do uso dessa topologia são o baixo uso de banda de rede, a possibilidade de uso de servidores Web convencionais e as triviais: obtenção e processamento de dados oriundos de dispositivos de entrada e a geração e exibição do ambiente gráfico.

As tecnologias para criação de aplicações nessa topologia devem permitir a alteração da configuração de software do cliente (navegador Web) e a resolução de forma automatizada de eventuais problemas de permissão de uso de recursos do sistema e de instalação de bibliotecas e outros artefatos de software necessários à execução da aplicação.

Assim, torna-se claro que a utilização de uma determinada topologia implica diretamente na escolha das tecnologias a serem empregadas; vez que os requisitos para cada uma delas são bastante distintos.

No estudo de caso realizado, a topologia escolhida foi a de processamento centrado no cliente. Esta escolha se deveu aos seguintes fatores: o fato de os usuários potenciais da aplicação (jogadores) possuírem, em geral, máquinas com considerável poder de processamento gráfico, da aplicação requerer respostas rápidas, e de se lidar com menos problemas relacionados a processamento distribuído.

Ressalte-se que as questões analisadas neste texto, relacionadas à tecnologia escolhida são pertinentes ao contexto de um jogo com ambientes bi e tridimensionais, baseado na Web, com topologia de

construção e processamento centrada no cliente (navegador Web).

3. Tecnologias emergentes

Em "Emerging Web Graphics Standards and Techonologies" [2003], Lau e outros autores discutem as vantagens da utilização de gráficos vetoriais e de ambientes virtuais distribuídos e apontam o que consideram as tecnologias mais promissoras para a criação de conteúdos gráficos 2D e 3D interativos para Web. Segundo eles, o *Scalable Vector Graphics* (SVG) em conjunto com a *Synchonizaded Multimedia Integration Language* (SMIL) constituem a solução mais indicada para a elaboração de conteúdos 2D baseados na Web. As características que corroboram esta indicação são a facilidade de integração dos documentos com outras tecnologias da Web (*HyperText Markup Language* - HTML e *Cascading Style Sheets* - CSS), o compartilhamento natural de dados com outros documentos derivados da *Extensible Markup Language* - XML e a possibilidade de trabalho em conjunto com outras linguagens de programação para Web (*Active Server Pages* - ASP, *Perl Hypertext Pages* - PHP, *Java Server Pages* - JSP e *Javascript / ECMAScript*).

Para conteúdos 3D, interativos baseados na Web, eles apontam a *Extensible 3D* (X3D) e a *Java 3D* (J3D) como sendo as tecnologias mais adequadas.

A primeira por ser um melhoramento da *Virtual Reality Modelating Language* (VRML), uma linguagem que já foi amplamente utilizada para a criação desse tipo de conteúdo. O melhoramento inclui alterações na interface de programação de aplicações avançadas, formatos de codificação adicionais e uma arquitetura baseada em componentes que permite a modularização dos conteúdos.

A segunda tecnologia, a *Java 3D* (J3D), é apontada como adequada por prover ao programador uma interface de programação de aplicações (*Application Programming Interface* - API) de alto-nível, com uma poderosa estrutura de dados para a manipulação de ambientes tridimensionais, denominada de grafo de cena, além de todas as facilidades da linguagem de programação Java. A independência de plataforma, com a possibilidade de finalização dos gráficos (*rendering*) em diferentes interfaces de programação de aplicações (API) de baixo-nível (DirectX e OpenGL) são outras características apontadas como relevantes pelos autores.

Além dessas tecnologias, vale lembrar que em termos de gráficos 2D, a *Macromedia Shockwave Flash* (ou simplesmente Flash) também possui requisitos interessantes para a produção de conteúdos gráficos interativos baseados na Web. [Steinmeyer 2005] Isso se comprova ao se considerar que a tecnologia Flash constitui atualmente um padrão de mercado, sendo amplamente utilizada nos mais diferentes conteúdos para Web, o que vai de simples *banners* a complexos e extensos sites. Note-se que cartões virtuais são bons exemplos de aplicações populares tipicamente construídas em Flash.

O fato de possuir uma ferramenta que engloba tanto a criação dos elementos gráficos, quanto à elaboração da programação necessária para manipulá-los, bem como muitos recursos para a criação de animação (animação por quadros-chave, por trajetória, por morphing, etc.) e manipulação de sons, são os principais (não únicos) atrativos dessa tecnologia. A presença de suporte interno (*built-in*) nos navegadores mais populares e a forma inteiramente automatizada de detecção e instalação dos softwares necessários para sua execução (plugin's) em diferentes navegadores também merecem especial destaque. [Murray and Church 2003]

Para a integração e comunicação dos conteúdos 2D e 3D elaborados é considerado o uso das tecnologias *Java Media Framework* (JMF) e a linguagem *JavaScript / ECMAScript*. A primeira por ser uma extensão da linguagem Java (linguagem base de ambas as soluções 3D vistas como emergentes) para a manipulação de conteúdos de mídia diversos. A segunda por ser uma linguagem de programação para a internet amplamente difundida e executada no cliente Web (navegador), esquema operacional da topologia de processamento escolhida, que no estudo de caso analisado é centrado no cliente (navegador Web).

Assim, percebe-se facilmente que a escolha das tecnologias mais adequadas deve ser norteada pela verificação mais detalhada dos requisitos de cada uma delas, razão pela qual uma analise mais detalhada é apresentada nas seções seguintes.

3.1. Scalable Vector Graphics (SVG) e Synchronized Multimedia Integration Language (SMIL)

Assim como qualquer outro padrão de representação de gráficos, o SVG possui suporte ao desenho de todas as primitivas gráficas básicas (linhas, polígonos, retângulos, círculos e elipses). Também permite a criação de caminhos por meio de linhas, curvas de Bezier e arcos elípticos, bem como o preenchimento de polígonos com cores uniformes e/ou gradientes e a aplicação de filtros. As transformações afins de rotação, translação e mudança de escala, bem como o enviesamento (*skew*), são passíveis de aplicação. [Bowler 2001]

A sua descrição completamente textual por meio de *tags* XML, o torna uma solução especialmente interessante, pois permite fácil integração dos seus documentos com diferentes tecnologias. CSS e *Extensible Style Language* (XSL), são bons exemplos de tecnologias para descrição de estilos que podem ser utilizadas com o SVG. O seu suporte ao *Document Object Model* (DOM) permite que os documentos SVG sejam utilizados em conjunto com as mais variadas linguagens de programação de conteúdos para Web. Dentre elas, pode-se citar a JavaScript / ECMAScript, PHP e ASP. [Lau 2003]

Em razão do esforço realizado pelo *World Wide Web Consortium* (W3C) pela sua adoção em novos dispositivos de hardware e também devido à sua definição com base no estabelecido pelo *International*

Color Consortium (ICC), o que busca viabilizar o uso do SVG em múltiplas plataformas sem alterações nas cores dos gráficos, uma versão mais leve do SVG foi criada para utilização em dispositivos com pequenos poder de processamento e espaço de memória. O SVG-Tiny permite que gráficos SVG sejam gerados em PDA's e telefones celulares, entre outros dispositivos. [Bogaard 2004]

Segundo Probets e outros [2001], animações podem ser criadas em SVG por três maneiras básicas: usando o modelo declarativo de animação do próprio SVG, manipulando o DOM e usando a SMIL. Destas, a mais completa é a usando a SMIL. As demais são adequadas apenas para casos em que é desnecessária a sincronização de diferentes tipos de mídias, o que não ocorre na elaboração de conteúdos multimídia complexos. Por essa razão, recomenda-se a utilização da linguagem SMIL em conjunto com o SVG para a criação de conteúdos multimídia.

Usada para a criação apenas da mídia e/ou de apresentações multimídia integradas com fluxos de áudio, vídeo, imagens e texto, a SMIL possui uma gramática derivada da XML. Essa condição a torna adequada a operar em conjunto, a exemplo do SVG, com os padrões CSS e XSL. [Bulterman 2005; Flammia1998]

Os 5 pontos mais relevantes para uso dessa linguagem são [Rutledge 2001]:

- Integração - SMIL é uma linguagem que permite somente a integração de mídias em uma única apresentação, mas não a sua criação,
- Disposição – Parâmetros como as dimensões e o posicionamento dos itens são bons exemplos de informações de disposição utilizadas na SMIL.
- Sincronismo – A SMIL permite a sincronização de diferentes itens de mídias, associando, por exemplo, textos como legenda de filmes, sons como explicações de animações, etc.
- Ligação – Pelo uso de *hyperlink's*, a SMIL permite a criação de referências a elementos em outros endereços Web ou mesmo em outras partes da própria apresentação.
- Adaptabilidade - Permite que as apresentações sejam desenvolvidas considerando não apenas um, mas vários ambientes de execução. Com isso, por exemplo, uma mesma apresentação pode apresentar legendas ao ser executada em diferentes lugares do mundo.

Os dois empecilhos mais relevantes à utilização do SVG e da SMIL para a criação de aplicações gráficas 2D interativas para Web estão relacionados à dificuldade de elaboração e distribuição das aplicações.

A possível dificuldade na elaboração se deve ao fato de que os principais softwares para criação de documentos SVG geralmente não possuem funcionalidades que permitam a criação de documentos SMIL. O mesmo acontecendo com os softwares para a criação dos documentos SMIL em relação ao SVG. Essa característica pode representar um empecilho ao se considerar que o desenvolvimento de aplicações é um processo cíclico em que as várias fases são continuamente revisadas pelos desenvolvedores. Em termos práticos a utilização de pelo menos duas ferramentas (uma para a criação de documentos SVG e outra para a criação de documentos SMIL) é indesejável por causar uma intensa mudança de contexto de desenvolvimento.

No que diz respeito à distribuição, uma barreira existente é a falta de suporte interno a essas tecnologias nos navegadores mais populares. O Microsoft Internet Explorer (MSIE), por exemplo, navegador mais popular do mercado segundo sites tradicionais especializados em estatísticas sobre a internet tais como TheCounter.com Global Statistics (www.thecounter.com/stats/), OneStat.com (www.onestat.com/), AdTech (adtech.info/) e Net Applications (www.netapplications.com/), não possui suporte interno (built-in) a aplicações nos formatos SVG e SMIL [Save.. 2006]. Deles somente o Mozilla Firefox apresenta suporte para ambas as tecnologias a partir da sua versão 1.5. [Mozilla.. 2006]

3.2. Macromedia Shockwave Flash (Flash)

Flash é um ambiente de desenvolvimento integrado (*Integrated Development Environment* - IDE) e Flash Player é uma máquina virtual usada para executar arquivos Flash. Na linguagem coloquial, entretanto, há uma mistura de conceitos. Flash pode ser tanto o ambiente de autoria, o *player* ou o arquivo da aplicação. Os arquivos Flash, normalmente chamados de filmes Flash, usualmente tem a extensão .swf e podem aparecer como elementos de páginas Web ou serem executados separadamente no Flash Player.

Os filmes Flash são constituídos por elementos de mídia, tais como, imagens, sons, vídeos, objetos gráficos, texto, por scripts que definem o seu comportamento e por um tipo especial de símbolo, o clipe de filme (*movie clip*). Clipes de filmes são mini-filmes Flash, o que equivale a dizer que são constituídos, como o filme principal, de elementos de mídia, scripts e por clipes de filme. Assim como elementos de mídia convencionais, eles possuem propriedades, tais como coordenadas, brilho, ângulo de rotação, altura e largura, que podem ser alteradas por meio dos scripts e pela interface da IDE Flash. Analogamente, os filmes Flash equivalem às aplicações em si e os clipes de filme a sub-programas especializados em tarefas específicas do sistema. [Battaiola et al. 2005]

A interface da IDE Flash representa metaforicamente o ambiente de produção de um filme. Assim, o conceito de cena é amplamente aplicado. Os atores atuantes, nesse contexto, são representados pelas mídias manipuladas. Enquanto que a locação, o camarim, a cenografia, a marcação de cena e os scripts que orientam a atuação dos atores são representados por painéis.

O painel que representa a locação onde se passa o filme é denominado dentro da aplicação de "cena". Neste painel é possível visualizar o conteúdo que está sendo desenvolvido e seus controles permitem iniciar e interromper a execução do filme, entre outras coisas. O

camarim é representado pelo painel denominado "biblioteca", onde é apresentado o elenco de atores do filme. Eventualmente, em caso de cenas muito complexas, a divisão dos atores em vários elencos é recomendada. Com isso, é possível ter mais de uma biblioteca para uma mesma cena.

A estrutura cenográfica é composta por painéis de criação e edição de conteúdos gráficos. Eles contêm as ferramentas básicas de desenho, como traçado de retas, desenho de curvas e de polígonos, ferramentas de preenchimento e paletas para manipulação de cores. Outros recursos permitem a produção de efeitos visuais mais sofisticados como degrades e reflexo.

A marcação das cenas do filme é representada por um painel denominado "linha do tempo". Nele, a entrada e a saída dos personagens da cena é representada por barras horizontais. Como um mesmo ator pode representar vários personagens em uma mesma cena, é comum ter diferentes barras para demonstrar a sua entrada e saída. Essa situação em que um mesmo ator incorpora vários personagens numa mesma cena é denominado dentro do Flash de "instanciação". [Battaiola et al. 2005]

As ações que cada personagem deve executar são determinadas, assim como nos filmes convencionais, por scripts. "Ações" é o nome dado ao painel onde são editados os scripts de cada personagem. As linguagens utilizadas para a criação de scripts, a partir da versão 8 da IDE, são a JavaScript e a ActionScript. Javascript / ECMAScript é uma linguagem de programação de aplicações para Web detalhada em uma próxima seção. [Flash.. 2006a]

Considerada um dialeto do Javascript / ECMAScript, a ActionScript é uma linguagem orientada a objetos, com sintaxe semelhante a do Java e suporte a programação dirigida por eventos e ao DOM. Com isso, a tarefa de elaborar scripts para determinar o comportamento dos elementos de um filme pode ser definida como a construção de uma hierarquia de objetos que trocam mensagens entre si em função, principalmente, de eventos ocorridos.

Dentre os eventos padrão tratados pela ActionScript estão os externos e internos. Exemplos de eventos externos são: acionamento / pressionamento e liberação de teclas do teclado e acionamento / deslocamento e pressionamento de botões do mouse. Exemplos dos internos: aqueles que ocorrem em função de colisões entre elementos, entrada em novo quadro (frame) do filme, etc. O suporte ao DOM permite a sua comunicação com outras linguagens e padrões Javascript / ECMAScript, JSP, ASP, CSS e XSL são bons exemplos de linguagens e padrões que podem se comunicar com ActionScript via DOM. [Flash.. 2006b]

3.3. Extensible 3D (X3D) com Xj3D

X3D é um padrão aberto de formato de arquivo baseado em XML criado para a comunicação de dados 3D tanto para aplicações convencionais, quanto para aplicações em rede. Possui um conjunto rico de funcionalidades para uso em engenharia, visualização científica, desenho auxiliado por computador (CAD), arquitetura, visualização de dados médicos, treinamento e simulação, multimídia, entretenimento e educação, entre outros. [X3D.. 2006]

Construído a partir da VRML, o X3D apresenta um conjunto de novas funcionalidades, tais como interface avançada de programação de aplicações, formatos de codificação de dados adicionais, conformação mais estrita e arquitetura dividida em componentes, o que permite o suporte modular ao padrão. Além destas, possui também os recursos básicos da VRML (gráficos 3D e 2D, animação, áudio e vídeo distribuídos no espaço, interação com o usuário, navegação, objetos definidos pelo usuário, escrita de scripts, operação em rede e simulação física). [Abstract.. 2006]

A X3D conta também com uma arquitetura que permite, ao contrário da VRML, a utilização de subconjuntos da especificação. Os denominados perfis (*profiles*) são compostos de blocos modulares de funcionalidades denominados componentes. São quatro os perfis básicos existentes em X3D: intercâmbio, interativo, imersivo e completo. Eles permitem a utilização desde elementos primitivos para comunicação entre aplicações somente (perfil intercâmbio), até a de todos os elementos contidos na especificação (perfil completo).

A organização dos conteúdos tridimensionais acontece no X3D através de uma estrutura de dados básica. O grafo de cena X3D é uma estrutura que contém todos os objetos do ambiente tridimensional e seus relacionamentos, os quais são descritos através das arestas que ligam os vértices que representam os elementos e transformações.

A sintaxe derivada da XML permite a integração dos conteúdos com outros padrões Web. Web Services e SVG são bons exemplos de tecnologias que podem operar perfeitamente integradas com o X3D.

O Xj3D é uma toolkit Java de código aberto (regida pela licensa LGPL), criada para a visualização e a manipulação de conteúdos escritos em X3D e VRML. Elaborada pelo grupo de trabalho da Web3D que idealizou o X3D, ela foi desenvolvida inicialmente como carregador de arquivos X3D para a API Java 3D (descrita a seguir). Atualmente, contudo, o seu grau de desenvolvimento a tornou a principal ferramenta usada pelos grupos de teste da especificação X3D.

A sua adoção pela comunidade X3D se deve ao fato de ter ganhado funcionalidades, das quais se destacam: a possibilidade de uso de diferentes vias (tais como Java 3D, DirectX e OpenGL) para finalização de cenas (*rendering*) e a utilização da interface de programação externa do X3D (SAI) [Xj3D 2006a, Xj3D.. 2006b]

3.4. Java 3D (J3D)

Para facilitar a composição e controle de cenas, a J3D é constituída de uma estrutura de dados, denominada grafo de cena, que é capaz de englobar desde dados geométricos a comportamentos dos modelos. O grafo de cena é, tecnicamente falando, um grafo acíclico direcionado no qual uma estrutura

hierárquica de relacionamentos entre os objetos da cena é representada. [Barrilleaux 2000; Wang. et al. 2002]

Ele é composto geralmente por um nodo *VirtualUniverse*, o qual define um universo. Como filho, esse nodo deve possuir ao menos um nodo do tipo *Locale*, o qual define um sistema de coordenadas onde estará situado o subgrafo que contém os elementos da cena. Este é composto tipicamente por dois *BranchGroup*, nodos especiais utilizados para agrupar nodos com características semelhantes. Um destes nodos agrupa os elementos em si, ou seja, os modelos geométricos, as aparências, os comportamentos, as localizações, os sons e as luzes, pertencentes à cena. Outro determinar os parâmetros de visualização da cena, tais como, posicionamento, volume de visualização, orientação [Bouvier 2000].

Os modelos geométricos animados podem ser inseridos na cena de duas formas básicas: pela instanciação de primitivas existentes na J3D (cubo, elipse, esfera, etc) ou pela carga de modelos elaborados por ferramentas de modelagem. Como o método de instanciação é inviável para a elaboração de modelos complexos, a carga de modelos é a via mais recomendável para a criação de cenas. Por essa razão, a J3D API possui carregadores internos (built-in) para modelos nos formatos VRML e Object, além de uma classe abstrata que permite a implementação de carregadores para os demais formatos de modelos geométricos. [Selman 2004]

Atualmente, há carregadores eficientes implementados usando essa classe abstrata para modelos animados em VRML e X3D. Uma lista completa de carregadores de modelos geométricos para J3D é mantida permanentemente no endereço (http://java3d.j3d.org/ utilities/loaders.html).

Além desses recursos, a J3D, por ser uma extensão padrão da linguagem Java, conta com todos os recursos dessa linguagem para a construção de estruturas de dados complexas, comunicação em rede, formulação de aplicações complexas, interfaces, etc. A independência de plataforma e as formas automatizadas para a instalação e a atualização dos componentes de software necessários para a execução permitem que suas aplicações sejam utilizadas sem maiores dificuldades por usuários de diferentes sistemas operacionais e navegadores. [Wijkman et al. 2003]

Outra característica importante derivada do relacionamento da J3D com os demais conteúdos Java é a possibilidade de utilização de dispositivos de entrada não usuais, como joysticks. Rosenblum e Macedônia [Rosenblum and Macedonia 1999] salientam o fato de aplicações J3D poderem possuir um número indeterminado de dispositivos. Eles ressaltam também a necessidade de tais dispositivos respeitarem ao denominado acesso em 6 graus de liberdade (*six-degrees-of-freedom*), o qual é suportada pelas 9 operações de acesso genérico existentes na Jinput (Java Input Controller API), extensão Java destinada ao controle de dispositivos de entrada não usuais.

3.5. Java Media Framework (JMF)

Fundamentalmente, a JMF é uma extensão da Java para a manipulação de áudio e vídeo. Mais formalmente, a *Java Media Framework Application Programming Interface* (JMF API) é uma API opcional para a extensão das funcionalidades do núcleo da linguagem Java. Suas principais funcionalidades são [Terrazas et al. 2002]: independência de plataforma, manipulação integrada de áudio e vídeo como objetos de mídia, suporte para um significativo número de tipos de conteúdo de áudio e vídeo e codec's, execução de mídia, gravação de mídia para arquivo, captura de mídia a partir de dispositivos como câmeras e microfones, recebimento de fluxos de mídia transmitidos através da Internet, transmissão de fluxos de mídia através da Internet, multiplexação / demultiplexação (combinação ou divisão) de mídias, transcodificação (alteração para um formato diferente) de mídias, unificação em um quadro de trabalho (*framework*) de suporte a todas as operações em mídias (por exemplo, efeitos) como processamento, extensibilidade para suportar outros formatos e plugin's e natural integração com a API Java existente.

Dirigida a eventos, a JMF também possui manipulação de exceções e suporte a sub-programas (threads). O seu modelo de gravação, processamento e apresentação de mídias é semelhante ao usado nos equipamentos convencionais, ou seja, a manipulação de mídias com JMF possui, a exemplo da manipulação convencional, fases bem distintas.

A captura de vídeo para uma fita feita a partir de uma câmera de vídeo, a execução do vídeo pelo vídeo-cassete e a apresentação do vídeo nos dispositivos de saída (televisor e alto-falantes) são representadas metaforicamente na JMF pelas fases de obtenção da mídia a partir de uma fonte de dados (captura a partir de um dispositivo de entrada, leitura de arquivo ou mesmo recebimento a partir de um fluxo de mídia a partir da internet), processamento (mixagem, edição ou mesmo simples execução da mídia) e apresentação (envio da mídia processada para os dispositivos de saída adequados). [Java.. 1999]

A JMF possui suporte a manipulação de um vasto conjunto de mídias, porém a sua comunicação com conteúdos SVG e SMIL é bem complexa. JMF não implementa comunicação com o DOM, modelo usado por esses padrões para a comunicação com outras tecnologias.

No que diz respeito à comunicação com conteúdos Flash, a situação é menos problemática. JMF permite manipulação de conteúdos Flash e a sua inclusão em aplicações Java, dentre elas Applets, no entanto, tal manipulação possui severas restrições. A maior delas se refere à versão dos conteúdos Flash manipulados. O sistema Flash está, presentemente, em sua oitava versão, mas somente a segunda versão do formato SWF é suportada pela JMF.

3.6. Javascript / ECMAScript

ECMAScript é uma linguagem baseada em objetos (objects), em outras palavras, todas as facilidades das

máquinas em que será executada são providas por objetos. Desta forma, um programa ECMAScript é um conjunto de objetos que se comunicam. Formalmente, um objeto ECMAScript é uma coleção não ordenada de propriedades (*properties*) que podem conter zero ou mais atributos (*attributes*) que determinam como cada propriedade pode ser usada. Propriedades são *containers* que mantém outros objetos, valores primitivos (*primitive values*) ou métodos (*methods*). [Standard 1999; Goodman 1998]

A linguagem define 5 tipos internos (*Undefined, Null, Boolean, Number* e *String*) e uma coleção de objetos internos (*built-in objects*), os quais incluem objetos para a representação de funções (*function object*), vetores (*array object*), palavras (*string object*), números (*number object*), datas (*date object*), etc. Também define os operadores e estruturas de controle e iteração clássicas e possui uma sintaxe semelhante a do Java. [Flanagan 1998]

A hierarquia de objetos da ECMAScript constitui uma representação dos elementos comuns em páginas Web. Quadros, botões e campos de formulário são alguns desses elementos representados nessa hierarquia. Cada navegador Web que suporta a execução de códigos ECMAScript implementa esse conjunto de elementos, ao qual é dado o nome de DOM. Graças a esse modelo, páginas HTML com código ECMAScript podem ser visualizadas nos mais diferentes navegadores.

Outro benefício advindo da utilização do DOM é a natural comunicação com padrões baseados nesse modelo. CSS, *Dinamic HTML* (DHTML), XLS, SVG, a linguagem Java (nas Applets Java) e também os conteúdos Flash são bons exemplos de padrões que podem ser utilizados em conjunto com a ECMAScript [Goodman 1998].

O fato da ECMAScript ser uma linguagem executada pelo cliente da comunicação Web (navegador) pode ser apontado como uma fator em potencial de desencorajamento de uso. Se comparada a linguagens executadas pelo servidor (servidor Web), tais como, PHP, ASP e JSP, essa linguagem é mais suscetível à manipulação indevida por parte de usuários mal intencionados, o que torna os sistemas desenvolvidos nessa linguagem mais vulneráveis a ataques.

3.7. Tecnologias utilizadas no estudo de caso

Com base na análise comparativa das tecnologias 2D emergentes (Tabela 1), pode-se observar que o SVG e a SMIL constituem a solução mais interessante do ponto de vista tecnológico, pois além de respeitarem os requisitos relativos à confecção de conteúdos, também possuem características que facilitam o estudo e a utilização, tais como baixo custo, padronização internacional e código aberto.

Contudo, ao se considerar questões de mercado e a necessidade das aplicações serem independentes de plataforma, a Flash se torna mais adequada. Isso em função de que a necessidade de instalação de plugins, bibliotecas e demais artefatos de software não ser amigável ao usuário (*user-friendly*). Por essa razão e também por já se dispor dos conteúdos 2D completamente implementados, o Flash foi utilizado no Edugraph.

Requisito	SVG & SMIL	Flash
Manipulação de diversas mídias	V	V
Gráficos vetoriais	V	V
Vasta capacidade de programação	V	V
Estruturas de dados complexas	V	V
Tratamento adis positivos de entrada	V	V
Integração com outras tecnologias	V	V
Independência de plataforma	V	V
Suporte interno (built-in)	X	V
Compactação de dados	X	V
Baixo custo de utilização	V	X
Padronização internacional	V	X
Código aberto	V	X

Tabela 1 – Análise comparativa das tecnologias 2D emergentes

Com respeito ao 3D, a análise comparativa das tecnologias emergentes (Tabela 2) revela um quadro mais favorável à solução formada por X3D e Xj3D. Apesar de não haver suporte interno dos navegadores Web mais populares a essas tecnologias (problema também existente no J3D), elas em conjunto possuem todos os demais requisitos estabelecidos como importantes para a elaboração de aplicações gráficas baseadas na Web.

Requisito	X3D & Xj3D	Java 3D
Manipulação de diversas mídias	V	V
Gráficos vetoriais	V	V
Vasta capacidade de programação	V	V
Estruturas de dados complexas	V	V
Tratamento adis positivos de entrada	V	V
Integração com outras tecnologias	V	X
Independência de plataforma	V	V
Suporte interno (built-in)	X	X
Compactação de dados	V	V
Baixo custo de utilização	V	V
Padronização internacional	V	X
Código aberto	V	V

Tabela 2 - Análise comparativa das tecnologias 3D emergentes

Requisito	JMF	Javascript
Manipulação de diversas mídias	V	V
Estruturas de dados complexas	V	V
Tratamento adis positivos de entrada	V	V
Independência de plataforma	V	V
Suporte interno (built-in)	X	V
Baixo custo de utilização	V	V
Padronização internacional	V	V
Vasta documentação	V	V
Comunicação com SVG e SMIL	X	V
Comunicação com Flash	X	V
Comunicação com X3D e Xj3D	V	V
Comunicação com J3D	V	V

Tabela 3 – Análise comparativa das tecnologias de integração

No que diz respeito à tecnologia de integração dos conteúdos 2D e 3D, a análise comparativa (Tabela 3) revela que a JavaScript / ECMAScript é, das tecnologias investigadas, a mais adequada à realização da tarefa. A extensão Java JMF, além de não ser

suportada pelos navegadores Web mais populares não permite a manipulação de conteúdos 2D em SVG com SMIL. Condição que não seria problemática considerando que a tecnologia usada para a confecção desse tipo de ambiente adotada é a Flash.

Contudo, também para essa tecnologia 2D há severas restrições quanto à possibilidade de manipulação. Somente conteúdos gerados sob a versão 2 do formato swf do Flash podem ser utilizados pela JMF, o que torna o seu uso inviável, vez que atualmente a versão do formato em uso é a 8.

A JavaScript / ECMAScript ao contrário, se revela bastante a mais adequada para a aplicação em questão, pois conta com recursos suficientes para suprir a todos os requisitos anteriormente estabelecidos.

4. Estratégias para a elaboração de jogos baseados na Web

Em razão das tecnologias 2D e de integração de conteúdos Flash e JavaScript / ECMAScript serem soluções bastante eficazes e difundidas para a criação de conteúdos bidimensionais e de aplicações baseados na Web, respectivamente, apenas as questões problemáticas relativas à criação de aplicações 3D serão abordadas nesse artigo.

4.1. Dependência de plataforma

Um dos aspectos fundamentais das aplicações baseadas na Web é a possibilidade de sua utilização a partir de qualquer lugar do mundo, sem que para isso sejam necessárias configuração de hardware e de softwares específicas. Por essa razão, o uso de tecnologias que privilegiam a criação de aplicações independentes de plataforma foi considerado.

O uso da *toolkit* Xj3D para a criação dos conteúdos 3D é um exemplo disso. Construída para manipulação de conteúdos X3D, ela conta com toda a estrutura da linguagem Java para a promoção de independência de plataforma. Tal estrutura possui como grande trunfo um interpretador (*Java Virtual Machine* - JVM) que é instalado por padrão em muitos sistemas operacionais e que, além disso, pode ser implantado nos sistemas que não o possuem automaticamente, sem que para isso seja necessário qualquer esforço adicional do usuário.

Contudo, apesar disso, as aplicações criadas com essa *toolkit* ainda enfrentam problemas de dependência de plataforma. Isso ocorre porque a Xj3D é uma *toolkit* que estende a linguagem Java e não é naturalmente suportada pela JVM. Wijkman e outros [2003] propõem duas soluções passíveis de aplicação a esse problema.

A primeira delas é empacotar a aplicação em arquivos JAR (*Java Archieves*) com descritores de conteúdo (*manifests)* especiais mostrando quais extensões Java são requeridas para a sua execução e onde (endereço Web) elas podem ser obtidas. A JVM, ao abrir arquivos JAR com essa característica, identificam se os artefatos de software necessários estão devidamente instalados e, no caso de não estarem, os instala automaticamente a partir do endereço Web passado.

Essa técnica permite que os artefatos de softwares necessários sejam instalados sob demanda, respeitando-se inclusive a versão requerida. Outra vantagem é a de que é possível especificar a instalação de diferentes artefatos de software para diferentes sistemas operacionais; sem que para isso qualquer esforço adicional seja necessário.

A segunda solução proposta por Wijkman é a criação de um carregador de arquivo, o qual se incumbirá de verificar se os artefatos de software necessários estão devidamente instalados e, no caso de não estarem, instalá-los, respeitando a versão e o sistema operacional em que a aplicação será executada. Esse carregador, a exemplo da primeira técnica, é especificado no descritor de conteúdo do arquivo JAR em que a aplicação é empacotada.

A vantagem dessa técnica é a possibilidade da instalação de artefatos externos a JVM, tais como imagens, *cookies*, sons, etc.

4.2. Restrição de acesso aos recursos do sistema

Por questões de segurança, a JVM policia as operações executadas pelas Applets Java e impõe a essas aplicações um conjunto bastante extenso de restrições. Destas, se destacam o impedimento à carga e a definição de métodos nativos, a leitura e a escrita em arquivos e a ativação de programas no *host* em que está sendo executada, a criação de conexões de rede (exceto com o *host* de onde foi baixada), leitura de certas propriedades do sistema e acesso a funções de baixo nível do sistema (funções de hardware). [Horstmann 2001]

A *toolkit* Xj3D utilizada para a confecção dos ambientes 3D faz uso de recursos de baixo nível sistema. Dos quais, um bom exemplo é a OpenGL, biblioteca gráfica implementada em hardware usada pela Xj3D para a finalização (*rendering*) de ambientes gráficos tridimensionais.

Em condições convencionais, tais restrições implicam na impossibilidade de uso da aplicação que necessita desse tipo de recurso. Porém, esse quadro pode ser revertido se medidas para a obtenção de autorização do usuário forem tomadas. A via prevista na JVM para a realização dessa tarefa é a da autorização por certificação digital.

Certificado digital é um comprovante fornecido por um indivíduo ou organização para a comprovação da sua identidade. Após o seu registro, a sua veracidade pode ser comprovada por uma autoridade certificadora. O uso deste recurso para a solução da questão de restrição de acesso a recursos do sistema é analisado por Pistoia e outros [2001], por Gong e outros [2001] e por Knudsen [1998].

A solução consiste na execução de 5 passos básicos: compilar a aplicação (usando o compilador javac), empacotar a aplicação em um arquivo JAR (usando a ferramenta jar), gerar as chaves criptográficas (por meio da ferramenta keytool), assinar a aplicação empacotada (usando a ferramenta

jarsigner) e exportar o certificado para registro na autoridade certificadora.

4.3. Utilização de dispositivos de entrada não-convencionais

Jogos de computador, assim como algumas aplicações de desenho auxiliado por computador, ferramentas de diagnóstico médico e de visualização científica podem fazer uso de dispositivos de entrada de dados não convencionais. A manipulação dos dados enviados por tais dispositivos é uma tarefa que até recentemente demandava bastante esforço, sobre tudo em aplicações cuja execução pode ser realizada em máquinas com as mais variadas plataformas de software.

Por essa razão, o grupo de desenvolvimento de tecnologias para jogos da Sun (Sun' s Game Technologies Group) gerou a Java Input Controller API (Jinput), interface de programação para a manipulação de dispositivos de entrada, que é capaz de tratar por meio de uma abstração comum dispositivos de entrada convencionais (teclado e mouse) e não-convencionais (*joysticks* e *game pads*). [Twilleager et al 2004]

Contida naturalmente na *toolkit* Xj3D, seu uso para a manipulação de *game pads* é explicado em detalhes por Davison em "Killer Game Programming in Java" [2005]. O método consiste na obtenção por *pooling* dos dados fornecidos pelos denominados controladores e sub-controladores. Nomes dados dentro da abstração da API aos dispositivos de entrada e seus elementos. O que pode ser realizado via leitura periódica através dos temporizadores (*timers*) existentes na aplicação.

5. Atual status de desenvolvimento da pesquisa

Seguindo uma metodologia de desenvolvimento em espiral proposta por Boehm [Sommerville 2003], a implementação do Edugraph tem sido orientada pela passagem cíclica por quatro etapas: definição de objetivos, avaliação e redução de riscos, desenvolvimento e validação e planejamento.

O objetivo estabelecido para o primeiro ciclo é resolução das questões mais recorrentes no desenvolvimento do jogo nesse primeiro laço do ciclo de desenvolvimento. Para tanto, foi realizada a divisão dos pontos a serem resolvidos em categorias, as quais foram abordadas isoladamente dando origem a um conjunto de protótipos especializados.

Os conhecimentos obtidos na confecção de cada protótipo subsidiarão a definição do objetivo do próximo laço do ciclo de desenvolvimento. O qual deverá ser focado na confecção da versão inicial da aplicação, a qual contará com os recursos explorados em todas as categorias investigadas no ciclo anterior do processo.

5.1. Resultados parciais obtidos

O protótipo a respeito da criação de ambientes tridimensionais em Applets Java usando a *toolkit* Xj3D já foi completamente desenvolvido e os seus resultados são consistentes. O mesmo acontece com os protótipos a respeito da carga e manipulação de cenas e adição de interatividade (manipulação dos dispositivos de entrada de dados), dos quais, a exemplo do anteriormente citado, foram obtidas informações para a confecção desse artigo.

Os protótipos especializados no controle/comutação das animações do personagem e na detecção de colisão já possuem resultados preliminares, contudo, ainda carecem de maior elaboração. Em ambos os protótipos a questão preponderante está relacionada à via mais adequada para a manipulação dos conteúdos tridimensionais. O X3D (a exemplo da VRML) possui duas interfaces de programação, uma interna e outra externa.

O uso da interface externa para a manipulação das cenas tridimensionais, além de mais simples, facilita o gerenciamento dos elementos da cena. Isto porque torna mais direta a associação entre a cena e as demais funcionalidades da aplicação. Contudo, o uso da interface interna permite uma melhor modularização dos algoritmos, possibilitando a criação de camadas de funcionalidades.

A camada criada usando a interface de programação interna poderia ser responsável pelo controle da cena tridimensional, enquanto que a criada para manipular as demais funcionalidades da aplicação seria responsável pela interface com o usuário. Uma terceira camada ainda poderia ser implementada para o gerenciamento da utilização da aplicação (camada de negócio), a qual poderia contar com comunicação com um servidor para *login* de usuários e demais funcionalidades administrativas recorrentes em jogos *on-line*.

No que diz respeito ao protótipo especializado na inserção e manipulação de áudio, poucos avanços foram obtidos. A utilização da *Java Audio Library* (JAL) tem sido considerada para a solução desse aspecto da aplicação, mas poucos experimentos foram realizados neste contexto.

Agradecimentos

Nossos especiais agradecimentos ao CNPq pela financiamento da pesquisa ora em curso.

5. Conclusão

Este artigo apresentou as informações preliminares obtidas em uma pesquisa de mestrado atualmente em curso, a respeito da criação de aplicações gráficas interativas baseadas na Web. Este emergente tipo de aplicação apresenta pesquisas ainda em estágio inicial, o que confere a este trabalho um certo nível de ineditismo.

Os resultados deste trabalho podem ser inteiramente empregados na construção de jogos baseados na Web. Categoria de jogos cujo alcance tende a crescer nos próximos anos.

Referências

ABSTRACT SPECIFICATION, 2006. *Web 3D Consortium* [online] Available from: www.web3d.org/x3d/specifications/ISO-IEC-19775-X3DAbstractSpecification/Part01/Architecture.html [Accessed 04 Apr 2006].

BARRILLEAUX, J. A., 2000. 3D User Interfaces with Java 3D. *Manning Publications Co.*,

BATTAIOLA, A. L. et al., 2005. Implementação das Fases 2D do Jogo Educacional Edugraph. Proceedings Simpósio Brasileiro para Jogos de Computador e Entretenimento Digital, Anais GameArt, Editora da Sociedade Brasileira de Computação, 78-83.

BATTAIOLA, A. L. et al., 2005. Tutorial ActionScript para Designers.. Proceedings Simpósio Brasileiro para Jogos de Computador e Entretenimento Digital, Anais GameArt, Editora da Sociedade Brasileira de Computação, 78-83.

BOGAARD, D. S. et al., 2004. SVG for Educational Simulations. SIGITE' 04 conference *ACM Conference On Information Technology Education*, 43-49.

BULTERMAN, D., 2005. *Synchronized Multimedia Integration Language (SMIL 2.1).* [online] World Wide Web Consortium (W3C). Available from: www.w3.org/TR/2005/CR-SMIL2-20050513/ [Accessed 04 Apr 2006].

BOUVIER, D J.., 2000. Getting Started with the Java 3D™ API. [online] Sun Microsystems, Inc.. Available from: java.sun.com/developer/onlineTraining/java3d/j3d_tutorial_ch1.pdf [Accessed 04 Apr 2006].

BOWLER, J., 2001. *Scalable Vector Graphics (SVG) 1.0 Specification* [online] World Wide Web Consortium (W3C). Available from: www.w3.org/TR/2001/REC-SVG-20010904/ [Accessed 04 Apr 2006].

DALGARNO, B., 2001. Technologies Supporting Highly Interactive Learning Resources on the Web: A Analysis. *Jornal of Interactive Learning Research*, 153-171.

DAVISON, A., 2005. Killer Game Programming in Java,

O'REILLY, 1ED.

FIELDING, R. T. AND TAYLOR, R. N., 2002. Principled Design of the Modern Web Architecture. *ACM Transactions on Internet Technology*, 2 (2), 115-150.

FLAMMIA, G., 1998. SMIL makes Web Applications Multimodal. *IEEE Intelligent Systems*, 12-13.

FLANAGAN, D., 1998. JavaScript: The Definitive Guide. *O'Reilly and Associates*. 3ed.

FLASH 8 DOCUMENTATION, 2006. *Adobe Systems Incorporated* [online] Adobe Systems Incorporated. Available from: livedocs.macromedia.com/flash/8/main/Part7_Extending.html [Accessed 04 Apr 2006].

FLASH 8 DOCUMENTATION, 2006. *Adobe Systems Incorporated* [online] Adobe Systems Incorporated. Available from: livedocs.macromedia.com/flex/20beta1/docs/00002555.html [Accessed 04 Apr 2006].

FRATERNALI, P., 1999. Tools and Approaches for Developing Data-Intensive Web Applications: A Survey. *ACM Computing Surveys*, 31 (3), 227-263.

GONG, L. et al. 2001. Inside Java™ 2 Platform Security: Architecture, API Design, and Implementation, *Addison-Wesley*, 2ed.

GOODMAN, D., 1998. JavaScript Bible. *IDG Books Worldwide*.

HORSTMANN, C, S., 2001. Core Java™ 2: Volume II–Advanced Features. *Prentice Hall PTR*.

JAVA™ MEDIA FRAMEWORK API GUIDE, 1999. *Sun Microsystems, Inc.* [online] Sun Microsystems. Available from: java.sun.com/products/java-media/jmf/2.1.1/guide/index.html [Accessed 04 Apr 2006].

KNUDSEN, J. B., 1998. Java Cryptography, O'REILLY, 1ED.

LAU, R. W. H. AND KUNII, T. L., 2003. Web Graphics. *IEEE Computer Graphics and Applications*, 26-27.

LAU, R. W. H. et al.., 2003. Emerging Web Graphics Standards and Techonologies. *IEEE Computer Graphics and Applications*, 66-75.

LEHMAN, J. A., 1986. Business Graphics: A Taxonomy for Information Systems Managers. *Data Base Fall*, 24-31.

MAGALHÃES, A., 2006. *O que esperar da internet nos próximos anos?* [online] IBOPE Inteligência, IBOPE//NetRatings, Internet, Notícias 2006. Available from: www.ibope.com.br [Accessed 04 Apr 2006].

MOZILLA SVG PROJECT, 2006. *SVG* [online] Mozzila Project. Available from: www.mozilla.org/projects/svg/ [Accessed 04 Apr 2006].

MURRAY, C. S. AND CHURCH, J. E., 2003. Macromedia Flash MX Game Programming. *Premier Press*.

PISTOIA, M. et al. 2001. Core Java™ 2: Volume II–Advanced Features. *Prentice Hall PTR*.

PROBETS, S. et al., 2001. Vector Graphics: From PostScript and Flash to SVG. Proceedings ACM Symposium on Document Engineering (DocEng ' 01), 135-143.

PROJETO DE PESQUISA LUDICLEARNING, 2006. *Laboratório de Animação Interativa (LAI)* [online] Universidade Federal do Paraná - UFPR. Available from: www.design.ufpr.br/lai/Projetos/LudicLearning/ludiclearning.html [Accessed 04 Apr 2006].

ROSENBLUM, L. AND MACEDONIA, M., 1999. The Java 3D API and Virtual Reality. *IEEE Computer Graphics and Applications,* 12-15.

RUTLEDGE, L., 2001. SMIL 2.0: XML for Web Multimedia. *IEEE Internet Computing Archieve*, 5 (5), 78-84.

SAVE AS WEB PAGE BROWSER AND OUTPUT FORMAT COMPATIBILITY. 2006. *Microsoft Internet Explorer (MSIE)*.. [online] Microsoft Inc. Available from: office.microsoft.com/en-us/assistance/HP010501811033.aspx?mode=print/ [Accessed 04 Apr 2006].

SELMAN, D., 2004. Java 3D Programming. *Manning Publications Co.*,

SOMMERVILLE, I., 2003. Engenharia de Software, Tradução André Maurício de Andrade Ribeiro. *Pearson Addison Wesley*, 6 ed., 44-46.

STANDARD ECMA-262 ECMASCRIPT LANGUAGE SPECIFICATION, 1999. *ECMA General Assembly* [online] ECMA Resouce Group. Available from: www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf [Accessed 04 Apr 2006].

STEINMEYER, P., 2005. *Development Platforms for Casual Games.* [online] Portal Gamasutra. Available from: www.gamasutra.com/features/20050324/steinmeyer_01.shtml [Accessed 04 Apr 2006].

TERRAZAS, A. et al.., 2002. Java™ Media APIs: Cross-Platform Imaging, Media, and Visualization. *Sams Publishing*,

TWILLEAGER, D. et al., 2004. Java™ Technologies for Games. ACM Computers in Entertainment, 2 (2), 1-9.

X3D OVERVIEW, 2006. *Web 3D Consortium* [online] Available from: www.web3d.org/x3d/overview.html [Accessed 04 Apr 2006].

XJ3D TOOLKIT, 2006. *Web 3D Consortium* [online] Available from: www.xj3d.org/ [Accessed 04 Apr 2006].

XJ3D - JAVA BASED X3D TOOLKIT AND X3D BROWSER , 2006. *Web 3D Consortium* [online] Available from: www.web3d.org/x3d/xj3d / [Accessed 04 Apr 2006].

WANG, L. et al.., 2002. A Java 3D-Enabled Cyber Workspace. *Communications of the ACM*, 45 (11), 45-49.

WIJKMAN, P. A. I. ET AL.., 2003. Transparent Java standard extensions with native libraries on multiple platforms. Internation, *Conference on Principles and Practice of Programming in Java (PPPJ'03). Computer Science Press, Inc*., 41-43.