UNIVERSIDADE ESTADUAL PAULISTA
Faculdade de Ciências - Bauru
Bacharelado em Ciência da Computação

Airton Estevão Barbosa

Desenvolvimento de Aplicações Web com Base em Modelagem 3D
Módulo Portal Web

UNESP
2012

Airton Estevão Barbosa

Desenvolvimento de Aplicações Web com Base em Modelagem 3D Módulo Portal Web

Orientador: Prof. Dr. José Remo Ferreira Brega

Monografia apresentada junto à disciplina Projeto e Implementação de Sistemas II, do curso de Bacharelado em Ciência da Computação, Faculdade de Ciências, Unesp, campus de Bauru, como parte do Trabalho de Conclusão de Curso.

UNESP

2012

Airton Estevão Barbosa

Desenvolvimento de Aplicações Web com Base em Modelagem 3D Módulo Portal Web

Monografia apresentada junto à disciplina Projeto e Implementação de Sistemas II, do curso de Bacharelado em Ciência da Computação, Faculdade de Ciências, Unesp, campus de Bauru, como parte do Trabalho de Conclusão de Curso.

BANCA EXAMINADORA

Prof. Dr. José Remo Ferreira Brega
DCo – FC - UNESP – Bauru
Orientador

Profª. Drª. Simone das Graças Domingues Prado
DCo – FC - UNESP – Bauru

Prof. Dr. Marco Antônio Rahal Sacoman
DCo – FC - UNESP – Bauru

Bauru, 29 de outubro de 2012.

AGRADECIMENTOS

Meus agradecimentos destinam-se primeiramente aos meus professores e em especial ao orientador deste projeto, José Remo, pelos conselhos, atenção e grande contribuição.

Em segundo lugar ao meu parceiro de projeto, Bruno Casella, pelo empenho e trabalho em equipe.

Mas principalmente à minha família. Meu pai, minha mãe e meu irmão, sem o apoio, o amor incondicional e acima de tudo a base moral com os quais fui presenteado, nada teria sentido. Eu os amo.

RESUMO

Este trabalho apresenta o desenvolvimento de uma aplicação *Web* chamada *CityFreedom* baseada em modelagem 3D. O sistema desenvolvido demonstra a utilização das mais revolucionárias e inovadoras técnicas para criação de portais *Web* junto à navegação por cenários 3D integradas às próprias páginas, sem que seja necessária utilização de *plug-ins* ou *softwares* externos. Tudo funciona com base nos navegadores compatíveis.

O *CityFreedom* visa proporcionar ao usuário a sensação de imersão em realidade virtual de tal maneira a conseguir interagir com uma cidade tridimensional no intuito de conhecer novos lugares, viajar por uma região da cidade que sempre pensou em conhecer ou mesmo analisar estabelecimentos muito antes de frequentá-los. É a liberdade de conhecer e viajar pela cidade de maneira simples e trivial. É uma nova tendência, o futuro de desenvolvimento de aplicações *Web*.

ABSTRACT

This paper presents the development of a Web application called *CityFreedom* based on 3D modeling. The developed system demonstrates the use of most revolutionary and innovative techniques to create Web portals with the integrated 3D navigation scenarios to their own pages, without requiring any kind of plug-ins or external software. Everything works on the basis of compatible browsers.

The CtyFreedom aims to give the user the feeling of immersion in virtual reality so get to interact with a three-dimensional city in order to see new places, traveling in an area of town that has always thought of knowing or even analyze establishments long before attend-them. It's the freedom to know and traveling around the city in a simple and trivial way. It is a new trend, the future of of Web systems development.

LISTA DE SIGLAS

XML – eXtensible Markup Language;

HTTP – Hipertext Transfer Protocol;

SOAP - *Simple Object Access Protocol;*

IIS - *Internet Information Services;*

JSON - JavaScript Object Notation;

GUI - Graphical User Interface;

VS – Visual Studio;

WebGL – API de desenvolvimento de modelagens 3D;

VRML - Virtual Reality Modeling Language;

HTML - HyperText Markup Language;

HTML5 - Hypertext Markup Language, versão 5;

GNU - General Public License.

LISTA DE QUADROS

LISTA DE FIGURAS

Sumário

1. INTRODUÇÃO

Um sistema deve ser chamado de "aplicação *Web*" muito além do limiar que compõe a forma de acessá-lo, ou seja, da necessidade de um navegador para a sua utilização. Em outras palavras, um sistema ou uma solução passa a ser uma Aplicação *Web* quando tudo é processado por um servidor, servidor este, recipiente para o código em si, que dali é distribuído a todos os clientes através de uma linha de comunicação comum, no caso a *internet* ou uma rede corporativa privada, a própria rede, sendo portanto algo mais complexo que a simples premissa de funcionamento do navegador (ARMSTRONG, et al. 2005).

A generalização e facilidade na obtenção da informação e uso dos aplicações *Web* só foram possíveis graças às grandes descobertas na área da computação nas últimas décadas, ou mesmo nos últimos anos. Descobertas ainda maiores continuam sendo apresentadas ao mundo e continuamente integradas ao "cotidiano" das pessoas, sem que na maioria das vezes boa parte destes usuários sequer tenha ideia das novas formas como os sistemas são construídos, das novas formas como suas informações (muitas vezes privadas) são armazenadas, ou mesmo como tais novidades facilitam e tornam sua experiência nos sistemas e Aplicações *Web* mais relevantes no sentido de eficiência, rapidez e consistência (BONIAT, et al. 2005).

No âmbito das descobertas, a preocupação de compatibilidade foi real por um bom tempo, e muitas alternativas a este entrave foram apresentadas. Todavia, algumas boas ideias esbarravam na guerra e disputa das grandes desenvolvedoras e distribuidoras de tecnologia. Cada uma tinha seu próprio caminho de programação, sua própria linguagem e parâmetros. Justamente por isso, o desenvolvimento de aplicações *Web* se viu num dilema. Investir em tecnologias próprias para facilitar a compatibilidade entre os sistemas e ver a *internet* e junto a ela, as Aplicações *Web* tornarem-se cada vez mais "heterogêneas", ou simplesmente pensar numa solução que fosse além do sentido inicial de "compatibilidade" imaginado para época? (BEHR, et al. 2009). Da necessidade de criar a compatibilidade universal de comunicação na *Web*, que fosse independente dos sistemas que precisassem se comunicar, surgiu uma ideia revolucionária, o conceito de *WebService*.

A novidade dos *WebServices* foi a introdução da linguagem XML, que é um conjunto de regras para definir uma linguagem de marcação que divide um documento em partes, identificando cada uma .

Independentemente da maneira como o sistema fosse construído, da linguagem da qual ele surgiu ou do sistema operacional a rodar por trás do navegador, as informações serão transmitidas em formato XML (HAROLD 1999).

Pensando na palavra base da nova tecnologia que deu origem aos *WebServices* e tomando-a como referência, cabe a pergunta: assim como "compatibilidade" associada à inovação é a essência dos novos alicerces dos grandes aplicações *Web*, o que ainda restringe o uso e a criação de soluções *Web* com base nas tecnologias 3D? Pode-se muito bem introduzir algo novo, trabalhar também com a compatibilidade para permitir que os *browsers*, através da *internet*, forneçam aos usuários a oportunidade única de interagir com novos sistemas baseados em modelagem 3D. Pode-se também tornar os elementos 3D, a modelagem e a navegação por ambientes criados virtualmente uma realidade não só para aplicações complexas como jogos, mas também para a navegação diária e simples. Muito além, apresentar e incorporar às aplicações *Web* a imersão no mundo 3D. Talvez seja o próximo passo a seguir, diante das tendências a preencher as lacunas de possibilidades dos sistemas de computador, sejam eles, aplicações *Web* ou não.

1.1. Objetivos do Trabalho

1.1.1 Objetivos Gerais

O objetivo principal foi desenvolver um sistema que tenha como base de criação a interação entre modelagem 3D de ambientes e as técnicas clássicas das aplicações *Web*. Com base nessa modelagem criar uma solução que corresponda a um serviço para qualquer pessoa que tenha acesso a *internet* no intuito de obter informações a respeito de lugares, estabelecimentos ou mesmo pontos de referência de uma cidade. Esta aplicação poderá ser acessada por qualquer usuário a fazer uso de um dos *browsers* compatíveis com as tecnologias abordadas, sem que seja necessário instalar qualquer outro *software* externo ou *plugin*. As modelagens 3D, bem como todos os recursos do aplicação *Web* dependerão única e exclusivamente de um *browser* compatível.

Ao sistema foi dado o nome de "*CityFreedom*", que representa a liberdade de poder navegar por uma cidade de maneira diferente e revolucionária. É sentir-se livre para, visitar bairros, conhecer estabelecimentos ou mesmo explorar aquela parte da cidade que a pessoa sempre quis conhecer. É a liberdade concedida pela modelagem 3D em aplicações *Web*.

1.1.1 Objetivos Específicos

Os objetivos específicos deste trabalho consistem na criação da aplicação *Web CityFreedom* estudando e resolvendo os problemas relacionados à compatibilidade de modelagens 3D com o ambiente *Web* como se conhece. Além disso preparar o sistema para a a integração com a segunda parte deste projeto, "Módulo *Android*", desenvolvido por Bruno Casella, coautor da aplicação.

1.2. Organização da Monografia

No Capítulo 1 deste trabalho são apresentados alguns conceitos introdutórios, os quais serão abordados e discutidos mais detalhadamente no decorrer da discussão acerca do projeto.

No Capítulo 2 é apresentada a fundamentação teórica, cujo conteúdo apresenta-se com base nas tecnologias utilizadas para o desenvolvimento do trabalho, abordando cada uma delas de maneira a estabelecer a relação que resultou no projeto apresentado.

No Capítulo 3 são apresentados os materiais e metodologia para o desenvolvimento do projeto.

O Capítulo 4 é destino à apresentação do sistema criado, o *CityFreedom*. Neste Capítulo são descritos seus recursos, regras de negócio, ferramentas e funcionamento de maneira mais detalhada.

O Capítulo 5 destina-se à apresentação dos resultados obtidos com o desenvolvimento do sistema, desde a descrição do sistema final até a elucidação dos problemas e desafios encontrados.

No Capítulo 6 é apresentada a conclusão obtida com o desenvolvimento deste trabalho.

Por fim, no Capítulo 7, são apresentados os trabalhos futuros que poderão ser realizados com base no projeto apresentado por este documento.

2. FUNDAMENTAÇÃO TEÓRICA

O objetivo deste Capítulo é elucidar os conceitos técnicos e teóricos para facilitar a compreensão das tecnologias *X3DOM*, *X3D*, *WebServices* e do desenvolvimento de aplicações *Web*, bem como das ferramentas utilizadas neste trabalho. Serão também mostrados nesse capitulo aspectos relativos à realidade virtual e aumentada.

2.1 Desenvolvimento de Aplicações Web

A escolha majoritária por aplicações desenvolvidas em plataforma *Web* se deu pelo fato desta tecnologia ser mais acessível no que diz respeito aos recursos do cliente, além da inegável possibilidade de interação e compatibilidade com diversas plataformas, a exemplo dos sistemas operacionais. Em outras palavras, desenvolver aplicações *Web* significa dar "liberdade" aos usuários. Poderão usufruir do trabalho, bem como explorar as funcionalidades oferecidas pela cidade virtual "*CityFreedom*" qualquer pessoa que tenha acesso a um *browser* compatível, seja um usuário de *Windows*, *Linux* ou mesmo *Android* (VALÉRIO; MACHADO, 2012).

A arquitetura dos aplicações *Web* se baseia na ideia de cliente/servidor, em que uma requisição proveniente do computador cliente chega ao servidor através de um protocolo de comunicação (seja este protocolo *HTTP* ou *SOAP*, por exemplo). Ao receber a requisição, o servidor dispara uma resposta com o pacote de dados específico para atender à necessidade inicialmente levantada (ARAUJO, 1997).

Para a aplicação "*CityFreedom*" a comunicação cliente – servidor será realizada por meio de protocolo *HTTP* através de um servidor IIS, no entanto, haverá este e outro servidor se comunicando de maneira mais diversificada. O primeiro comportará a aplicação em si, as modelagens 3D e a GUI (interface do usuário). Por outro lado, o banco de dados estará num servidor isolado *LINUX* que, mediante recepção de requisições codificadas e enviadas em formato *JSON*, retornará ao servidor da aplicação (no mesmo formato *JSON*) as informações necessárias para possibilitar a visualização dos dados no computador cliente. Essa técnica proporciona maior credibilidade na segurança das

informações ao fazer uso de pioneiras técnicas de programação *Web* (CASELLA, 2012).

No âmbito de desenvolvimento da aplicação em si, a nível de GUI, o projeto "*CityFreedom*" une todas as recentes técnicas de desenvolvimento *Web* mais utilizadas atualmente. As páginas do portal foram construídas segundo utilização de estilos em linguagem CSS3 e aplicação de *scripts* utilizando a biblioteca JQUERY. A composição e construção da aplicação se deu sob os moldes das "*Master Page*s", recurso oferecido principalmente pela sintaxe *Asp .Net* da *Microsoft* em conjunto com a utilização do *software* "Microsoft Visual Studio 2010".

Segundo o MSDN (2012, Microsoft): "O recurso de *Master Page*s, sem nenhuma dúvida é uma facilidade que todo desenvolvedor sonhava em ter nos projetos. Com uma *Master Page* você consegue desenvolver uma página padrão que será utilizada em todo o site, ou seja, é como se fosse uma página default contendo menus, cabeçalhos e rodapés."

2.2 ASP .NET

Ainda segundo a MSDN (2012, Microsoft): "A sintaxe ASP.NET define a estrutura, o layout e as configurações de uma página ASP.NET. Ela também define o layout para controles do servidor ASP.NET, código do aplicativo, configuração do aplicativo e *WebServices* XML. Qualquer recurso ASP.NET pode ser criado em arquivos texto com a correta extensão do nome do arquivo. Isto inclui arquivos de configuração do ASP.NET, arquivos do aplicativo (Global.asax), páginas ASP.NET e controles do servidor *Web*."

A aplicação proposta neste projeto foi desenvolvida com base na plataforma *ASP .NET* associada à linguagem *HTML* e *C#.*

Optou-se por fazer uso desta tecnologia pela experiência concentrada nela mas, sobretudo, pela interação e facilidade proporcionados para a construção de páginas mais semelhantes a sistemas em si, com base principalmente em funcionalidades como as citadas "*Master Page*s", além de formulários característicos e controle de contas de usuário triviais. Outro questão importante está relacionado às bibliotecas integradas no que diz respeito à interação e construção de *WebServices*. Referenciar e criar o vínculo de comunicação em aplicações com base em *ASP .NET* é extremamente simples e estruturado.

2.3 Realidade Virtual e Realidade Aumentada

A Realidade virtual (RV) é uma técnica avançada de interface onde o usuário pode interagir e navegar em um ambiente tridimensional fazendo uso de dispositivos não convencionais de entrada e saída para trabalhar de maneira única os sentidos humanos - visão, audição e tato (VALÉRIO; MACHADO, 2002).

Os dispositivos podem ser luvas eletrônicas, capacete de visualização, visão eletroscópica, dentro outros. De forma extremamente natural e equivalente ao que fariam se estivessem interagindo com o mundo real, a ideia é que uma pessoa munida de tais equipamentos possa interagir com o ambiente virtual de maneira tão natural e espontânea à ponto de literalmente sentir-se parte do cenário(ADAMS, 1994).

A realidade virtual ainda é considerada como a união de três pilares principais, os quais compõem a base da estruturação da tecnologia: imersão - sensação de estar dentro do ambiente -, interação - possibilidade do usuário interferir com o que acontece no ambiente, e vice-versa - e envolvimento - capacidade do ambiente motivar o usuário a participar (KIRNER; PINHO, 1996).

Tendo em vista que o objetivo deste trabalho é a criação de uma aplicação *Web* torna-se importante discorrer primeiramente a respeito da modelagem 3D em si, a qual será o foco do estudo de RV possibilitando ao usuário a identificação dos três pilares da realidade virtual (imersão, interação e envolvimento), mesmo que o único sentido envolvido seja a visão mediante restrições da tecnologia implantada, no caso um simples *browser* compatível com renderização *WebGL*.

Para tal atenta-se às ferramentas, técnicas e experiência. No que diz respeito à modelagem, aquilo que ficou muito claro no âmbito tecnológico nos últimos anos foi o surgimento de diversos sistemas de modelagem gratuitos. Neste sentido, a modelagem 3D passa de algo fantasioso e tido como complexo e inviável para algo cada vez mais palpável. Diante desta premissa alguns bons exemplos de modeladores e programas próprios para a criação de cenários 3D fornecem simplicidade e desenvolvimento intuitivo para quem se prontificar a gastar o mínimo de tempo no estudo das técnicas envolvidas nestes programas (KIRNER, 2011).

Neste quadro de inovações aliadas às práticas cada vez mais singelas surge uma das ferramentas amplamente utilizadas neste projeto, o *FluxStudio*, o qual nada mais é que uma ferramenta de modelagem 3D capaz de gerar arquivos nos formatos *VRML* e *X3D*. Apesar de o *FluxStudio* ser gratuito, ele evoluiu para outro sistema muito mais complexo e

enriquecido com diversas outras funcionalidades, chamado Vizard. O Vizard, no entanto, tem um custo na ordem de vários milhares de dólares, o que confirma a ideia de que a renderização mais profissional e detalhada envolve grande quantidade de recursos e investimentos em *software* e *hardware* (KIRNER, 2011).

2.4 Considerações Gerais Sobre a Tecnologia VRML

Não há como analisar ou estudar a linguagem *X3D*, a qual consolidou-se como a base para o desenvolvimento de modelagens tridimensionais para aplicações *Web*, sem antes discorrer a respeito de sua predecessora, a linguagem *VRML* (*Virtual Reality Modeling Language*).

O *VRML* é uma linguagem para criação de ambientes tridimensionais utilizado para projetar cenários tanto para *browser* (com a utilização de *plug-ins*) e aplicações *Web* quanto para sistemas *Desktop* (os quais possuem naturalmente maior poder de processamento e renderização). A linguagem *VRML* possibilita a criação de objetos, manipulação de cores, brilho e transparência além da aplicação de texturas. Um diferencial da linguagem *VRML* foi a introdução de sensores possibilitando ao usuário a sensação de interação com o ambiente modelado (BORLAND; FESTA, 2005).

A linguagem *VRML* não foi necessariamente abandonada. A linguagem *X3D*, sua sucessora possui todos os pontos positivos e os recursos anteriormente empregados, a diferença está na mudança de sintaxe, focada e vinculada na tecnologia XML, a qual por questões de compatibilidade e desempenho é uma alternativa muito mais interessante para a modelagem 3D para aplicações *Web*.

Exemplo da sintaxe *VRML* (Figura 1 e 2):

```
#VRML V2.0 utf8
#Color example: a pyramid
Shape {
        appearance Appearance{
                material Material { }
                }
        geometry IndexedFaceSet {
                coord Coordinate {
                    point [
                                # bottom
                                -1.0 -1.0 1.0,  #vertex 0
                                1.0 -1.0 1.0,   #vertex 1
                                1.0 -1.0 -1.0,  #vertex 2
                                -1.0 -1.0 -1.0, #vertex 3
                                # top
                                0.0 1.0 0.0             #vertex 4
                    ]
                }
        colorPerVertex FALSE #so each face will have one of the colors
        color Color {
                    color [
                        1.0 0.0 0.0,#color 0
                        0.0 1.0 0.0,#color 1
                        0.0 0.0 1.0,#color 2
                        1.0 1.0 0.0,#color 3
                        0.0 1.0 1.0,#color 4
                    ]
                }
        colorIndex [ 2 3 4 0 0 ] #the first face will have color 2, the 2nd color3...
                coordIndex [
                        #bottom square
                         0, 3, 2, 1, 0, -1,
                        #side1
                         0, 1, 4, -1,
                        #side2
                         1, 2, 4, -1,
                        #side3
                         2, 3, 4, -1,
                        #side1
                         3, 0, 4, -1,
                        ]

        }
}
```

Figura 1. Exemplo de código *VRML*

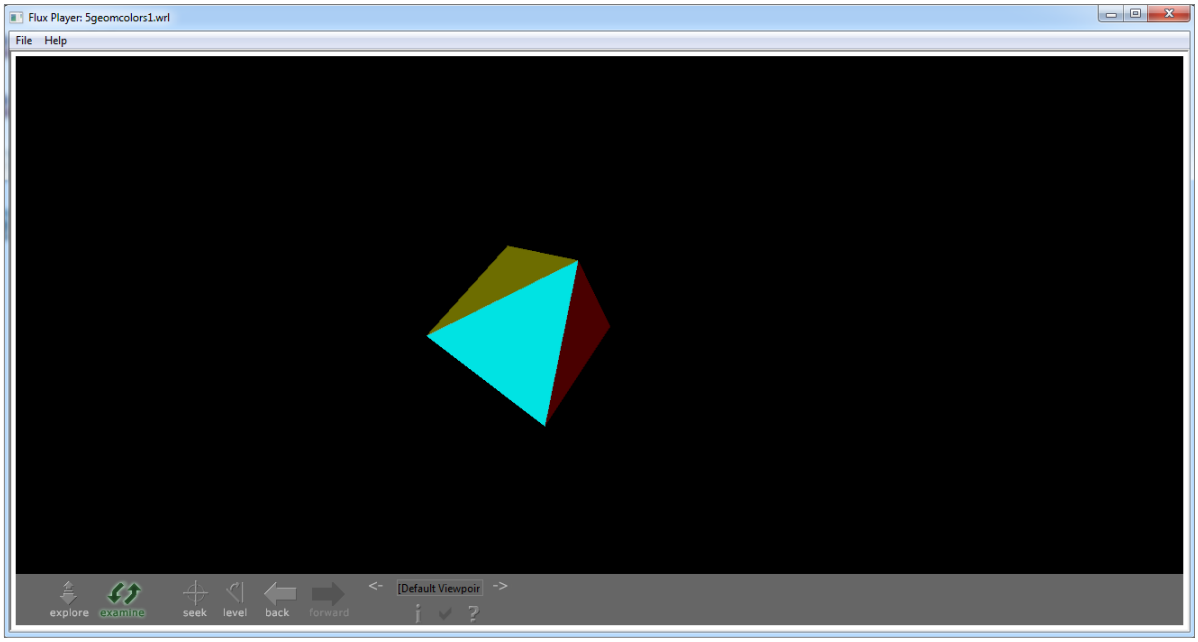

Figura 2. Representação do código *VRML* acima. Exemplo de construção 3D com sintaxe *VRML* no *plug-in* FluxPlayer.

2.5 Considerações Gerais Sobre a Tecnologia X3D

Surgindo da revisão da especificação do padrão *VRML*, é uma tecnologia de padrão aberto para distribuir conteúdo 3D, incorporando os avanços dos recursos disponíveis nos últimos dispositivos gráficos comerciais e também incorpora melhorias na sua arquitetura. É uma linguagem desenvolvida com a sintaxe *XML* capaz de representar e comunicar cenas 3D e objetos. A tecnologia *X3D* dá suporte para gráficos 3D, transformações de geometria, iluminação, materiais, texturas, mapeamento, *pixels*, vértices e aceleração de *hardware*. Permite animação com temporizadores e interpoladores de condução contínua. Permite também interação com o *mouse* e entradas de teclado. A navegação no ambiente acontece por meio do uso de pontos câmeras (*viewpoints*), com características de colisão, proximidade e visibilidade, detecção e vários tipos de iluminação (BRUTZMAN; DALY, 2007).

Para o foco deste projeto, convém elucidar o termo "*Browser X3D*". *Browser X3D* é aquele que possibilita não só a leitura e interpretação de arquivos ou código *X3D* como também possibilita a navegação por *viewpoints*, características de iluminação e animação anteriormente citadas. Os *browsers X3D* ainda interpretam os elementos *X3D* com base na linguagem *HTML* e nos novos elementos *HTML5*, ou seja, um arquivo *X3D* "puro" não necessariamente terá funcionamento amplo na interpretação dos *browsers* (como Mozilla *FireFox* ou *Google Chrome*). Para tanto é necessário converter o código *X3D* nativo em *tags HTML* possibilitando assim a renderização das cenas 3D de maneira a tornarem-se apresentáveis ao usuário final. (BRUTZMAN; DALY, 2007).

Exemplo da sintaxe *X3D* (Figuras 3 e 4):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN"
"http://www.web3d.org/specifications/x3d-3.0.dtd"/>
<X3D profile='Immersive' >
<head>
 <meta name='Vizthumbnail'
       content='Thumb_exemploMonografia_x3d393001351052894.jpg'/>
 <meta name='ExportTime' content='2:28:14'/>
 <meta name='ExportDate' content='10/24/2012'/>
 <meta name='FluxStudioVersion' content='659'/>
</head>
<Scene>
<WorldInfo
 title='Untitled'
 info='"This Web3D Content was created with Flux Studio, a Web3D authoring tool"
 "www.mediamachines.com"'/>
<Transform DEF='dad_Box1'
 translation='4.53006 0 2.26503'>
 <Shape DEF='Box1'
  containerField='children'>
  <Appearance
   containerField='appearance'>
   <Material DEF='Red'
    containerField='material'
    ambientIntensity='0.200'
    shininess='0.200'
    diffuseColor='1 0 0'/>
  </Appearance>
  <Box DEF='GeoBox1'
   containerField='geometry'
   size='1 1 1'/>
 </Shape>
</Transform>
<Transform DEF='dad_Sphere1'
 translation='4.57582 1.32699 2.31079'>
 <Shape DEF='Sphere1'
  containerField='children'>
  <Appearance
   containerField='appearance'>
   <Material
    containerField='material'
    USE='Red'/>
  </Appearance>
  <Sphere DEF='GeoSphere1'
   containerField='geometry'
   radius='1.000'/>
 </Shape>
</Transform>
</Scene>
</X3D>
```

Figura 3. Exemplo de código *X3D*.

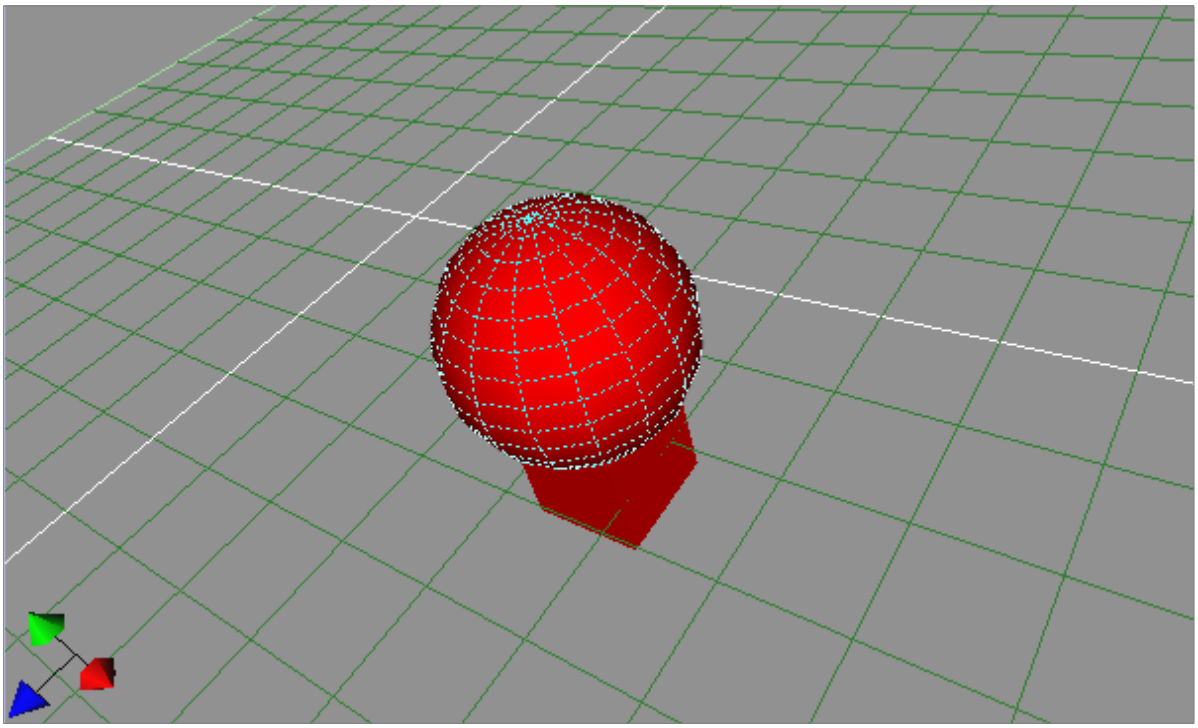

Figura 4.Representação do código *X3D* acima. Exemplo de construção 3D utilizando a sintaxe *X3D* no *FluxStudio*.

2.6 Ferramentas de Modelagem – FluxStudio e Blender

O *FluxStudio* é uma ferramenta de modelagem simples e intuitiva, que gera arquivos no formato *X3D*, os quais, mediante testes preliminares se apresentaram estáveis na leitura dos browsers dependendo do nível de detalhamento e da quantidade de objetos modelados. Por outro lado, esta ferramenta não possibilita o tratamento de modelagem necessário para criar objetos complexos o suficiente para uma exploração talvez mais abrangente da modelagem em si, caso as técnicas de aplicação de tais modelos sejam viáveis para um aplicação *Web*. Sendo assim, a tendência a trabalhar em conjunto utilizando a aplicação do *FluxStudio* e a aplicação *Blender* (também voltada para modelagem 3D) (Figura 5).

O *Blender* é um *software* gratuito para modelagem e trabalhos com gráficos 3D. Com ele é possível criar animações, jogos complexos e manipulação de vídeos. Ele foi concebido mediante a licença GNU/LINUX e está disponível também para Windows e *Mac*. Ao contrário do *FluxStudio*, o *Blender* é extremamente complexo e basicamente não é possível criar cenários 3D utilizando-o sem prévio conhecimento a respeito das funcionalidades da ferramenta. Por outro lado, ao utilizá-lo a possibilidade de criação de objetos cada vez mais complexos e ao mesmo tempo "leves" o bastante para não

comprometer o desempenho da aplicação torna-se extremamente viável (JAMES, 2010).

Com o *Blender* os modelos e objetos ficarão naturalmente mais condizentes com o nível de tecnologia atual, resta saber se os modelos são complexos o bastante a ponto de não permitir o desenvolvimento plausível de uma aplicação voltada à *Web*. Se for este o caso, a modelagem terá de ser restrita ao *FluxStudio*, o qual, como dito, possui uma modelagem mais simples e intuitiva cujos objetos podem ser facilmente criados usando um sólido qualquer preenchido com uma textura (Figura 6).

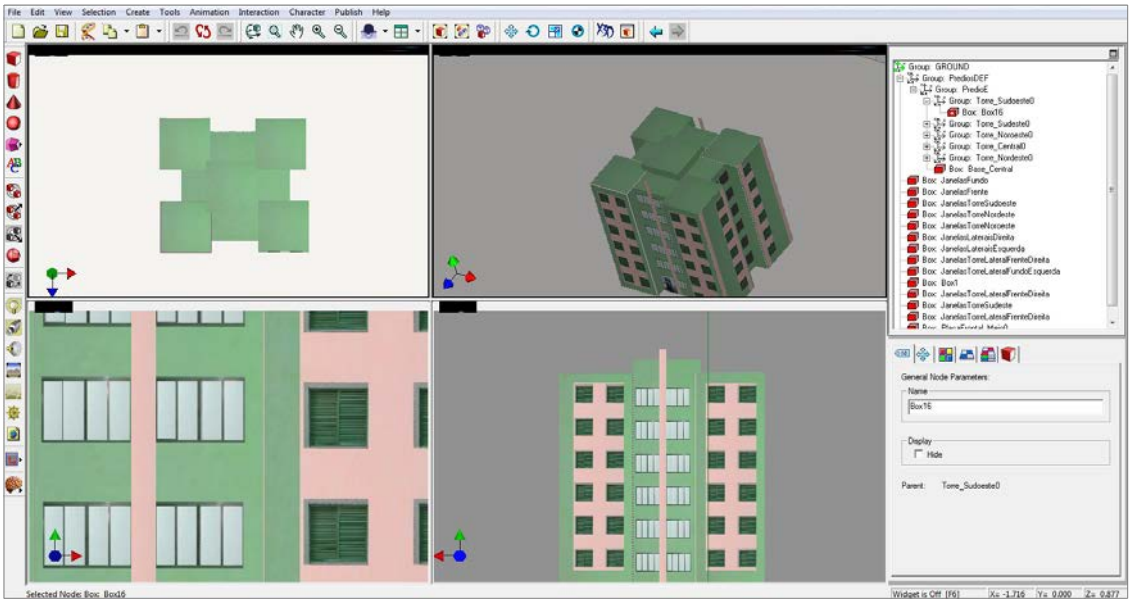

Figura 5. *FluxStudio*. Exemplo de prédio modelado para a aplicação *CityFreedom*.

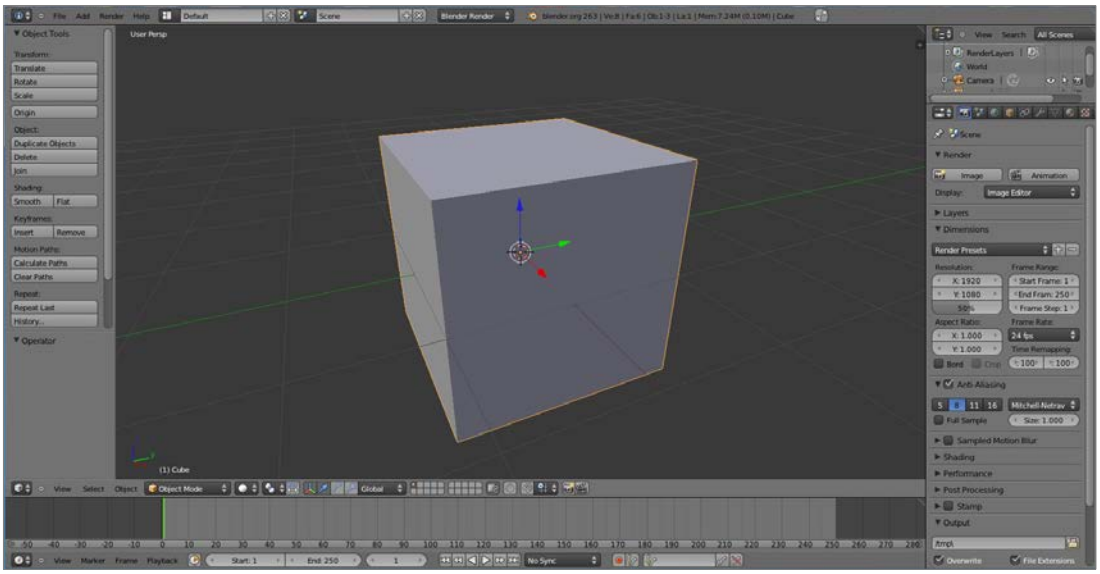

Figura 6. *Blender*. Exemplo da tela inicial.

2.7 Considerações Gerais sobre a Tecnologia WebGL

De nada adianta criar cenários completos em linguagem *X3D* organizados perfeitamente em nós XML se o *browser* não for capaz de interpretá-los. Diante deste problema surgiu em 2011 uma API concebida com base em *JavaScript* a partir do elemento *Canvas* do HTML5. Esta API era capaz de aplicar os recursos da modelagem 3D aos *browsers* e fazer com que os cenários e objetivos modelados segundo a linguagem *X3D* se tornassem parte integrante do desenvolvimento de aplicações *Web*. Sem a necessidade de *plug-ins* ou *softwares* de terceiros para que isso fosse possível (DELILLO, 2009).

A API *WebGL* tornou-se a base para o desenvolvimento de inúmeras outras ferramentas de criação de cenários 3D nesse período de pouco mais de um ano de seu lançamento. Dela surgiu, por exemplo, o *X3D*OM que é uma biblioteca que maximiza a utilização de *WebGL* em aplicações *Web* tornando as renderizações mais eficientes e dinâmicas, no contínuo e incessante esforço de melhorar cada vez mais o desempenho das modelagens numa plataforma teoricamente tão limitada, que são os *browsers* (*X3DOM*, 2012). Este projeto fará uso da biblioteca *X3DOM* para a criação de seus respectivos cenários e modelagens 3D visando a possibilidade de aproximar-se o máximo possível da realidade dos objetos criados, para que os conceitos de navegação e reconhecimento da cidade virtual tenham para o usuário o que há de melhor no que diz respeito a imersão no universo 3D.

2.8 Considerações Gerais Sobre a Tecnologia X3DOM

X3DOM é uma tecnologia, ainda em fase experimental, *opensource* que têm como objetivo dar suporte a discussões de como integrar as tecnologias como *HTML5* e um ambiente *Web* 3D declarativo, e permite incluir elementos *X3D* como parte do código *HTML5* de uma página *Web* (X3DOM, 2012).

As implementações atuais do *X3DOM* funcionam em browers que tem implementados *WebGL*, como o Firefox 4.0 para *desktop* ou mobile, e as ultimas versões do Opera para *desktop* e *Android*.

Na atual versão é possível colocar conteúdo *X3D* direto nas *tags* de *HTML5* através de uma biblioteca *Javascript*, o que auxilia no desenvolvimento.

3. MATERIAIS E MÉTODOS

3.1 Material

- *Notebook*;
- *Desktop* com *hardware* preparado para testes de desempenho das modelagens (Placa de vídeo ZOTAC GeForce 560GTX, Intel Core i5, 8Gb memória);
- *Tablet* para testes da aplicação em sistemas móveis;
- Windows 7 Professional Edition;
- Microsoft Visual Studio 2010 Premium;
- Microsoft IIS7.5 (*internet information services*);
- *Software* de modelagem 3D *FluxStudio* 2.1;
- *Software* de modelagem 3D *Blender* 2.63;
- Biblioteca *JQUERY* 1.7+;
- *API X3DOM*;
- Navegadores Google Chrome, Mozilla FireFox e Opera (Não há suporte à tecnologia tratada neste trabalho para o *browser Internet Explorer* e para o Safari os recursos funcionam apenas mediante uso de *plug-ins*);
- Câmeras (para obter fotos base dos ambientes reais a serem modelados);
- *Software* conversor *X3D*, *HTML5* (x3dom.org).

3.2 Metodologia

O desenvolvimento deste trabalho contou com as seguintes etapas:

- Estudo a respeito das técnicas de modelagem
- Desenvolvimento da aplicação *Web* em paralelo com a criação de modelagens exemplo;
- Integração das modelagens criadas com a aplicação "*CityFreedom*" em si;

- Integração com o *WebServer* e Banco de Dados referente à parte desenvolvida por Bruno Casella, a qual complementa este trabalho.

A primeira parte foi dividida em:

- A partir de exemplos simples modelados, trabalhar a renderização das modelagens nos *browsers*;
- Em um primeiro momento essa interação era feita através de *plug-ins* como o "Octaga Player". O esforço foi para encontrar uma maneira de incluir as modelagens 3D na aplicação sem o uso de *softwares* externos;
- Encontrar maneiras para melhorar o desempenho da aplicação mediante modelagens mais complexas;
- Estudar as ferramentas de modelagem para criar cenários cada vez melhores e mais eficientes.

A segunda parte foi dividida em:

- Criação da *master page* base para a aplicação;
- Criação das páginas referentes a cada funcionalidade do sistema;

A terceira parte foi dividida em:

- Buscar alternativas para transformar os arquivos *X3D* das modelagens de cenários 3D em formato HTML5;
- Estruturar o código gerado para a página ASP.NET;
- Criar interações do usuário com o cenário apenas utilizando codificação HTML5;
- Garantir a funcionalidade das interações do usuário com o cenário;

A quarta parte foi dividida em:

- Preparar as telas da aplicação que necessitarão as informações do banco de dados;
- Trabalhar em conjunto com Bruno Casella (co-autor do projeto) para unir o *WebServer* criado por ele à aplicação "*CityFreedom*".

4. SOFTWARES DESENVOLVIDOS (PORTAL "CITYFREEDOM")

A aplicação foco dos estudos deste projeto e na qual foi investido todo o conhecimento e a experiência adquiridos durante o processo de desenvolvimento é um aplicação *Web* nos moldes clássicos, com controle de conta de usuário, navegação instintiva, interface de sistema (diferente do padrão atual das páginas em estilo "blog") e acesso a uma base de dados para obtenção das principais informações. Este aplicação *Web* cujo nome faz referência à liberdade tem sua ideia ideia e objetivo principal fundamentados nesta palavra (Figura 7).

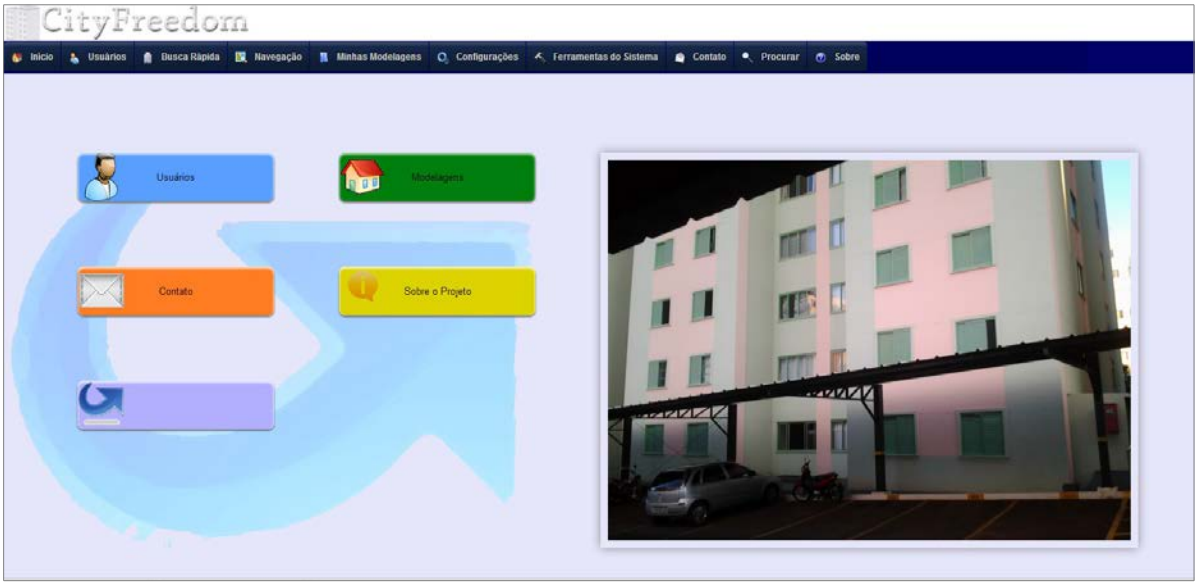

Figura 7. Página inicial do protótipo *CityFreedom*.

A intenção de fornecer ao usuário a possibilidade de navegar livremente por uma cidade para que possa conhecer lugares, estabelecimentos, pontos turísticos ou mesmo localidades de referência, antes mesmo de estar presente em tais lugares fisicamente está ligada intrinsecamente ao fato e à necessidade das pessoas, na sociedade atual, de (pelas razões mais distintas possíveis) de conhecer novos lugares constantemente. Com base nisso pensou-se em como seria agradável e útil proporcionar tal possibilidade de familiarização com um lugar mesmo antes de conhecê-lo de fato. O sistema *CityFreedom* fornece por meio de utilização e integração das mais avançadas técnicas de desenvolvimento de aplicações *Web* este sentimento de liberdade ao usuário. Todavia, muito além disso, uma aplicação que envolve e une tantas tendências busca também, em caráter mais implícito,

explorar e divulgar as técnicas de desenvolvimento *Web* com base em modelagem 3D no intuito de mostrar às pessoas que a criação de aplicações *Web* com navegação em cenários tridimensionais já não é mais um sonho distante. Da mesma maneira como é comum e cotidiano a alguém abrir seu *browser* pela manhã e ler as notícias do dia sustentadas por textos, imagens e até animações concebidas através de scripts, será comum e cotidiano, em breve, navegar pela *internet* sob a imersão do universo da RV.

O processo de criação da aplicação teve inicio com a montagem das telas utilizando o Visual Studio 2010 com base numa *Master Page*. Além disso foram criados scripts e arquivos de estilo (CSS) para a identificação visual do sistema. Em paralelo a este desenvolvimento modelagens 3D mais simples foram criadas para estudar o impacto em uma aplicação e as técnicas necessárias para o bom funcionamento na plataforma base para aplicações *Web*, o *browser*.

A primeira dificuldade encontrada para dar continuidade ao desenvolvimento do sistema foi encontrar uma maneira de desvincular as modelagens 3D de *plug-ins* ou aplicações de terceiros, pois além de comprometer a ideia inicial do projeto, sustentada pela palavra "liberdade", ainda restringia a integração com os outros elementos da interface do usuário o que prejudicaria entre outras funcionalidades a navegação. O problema se dava pois, ao criar um cenário 3D tem-se como resultado um arquivo no formato *X3D*. Imaginou-se inicialmente que haveria alguma forma de utilizar a codificação nativa *X3D* diretamente na codificação *HTML*, o que de fato não foi possível. Os *plug-ins* resolviam tal impasse ao criarem a partir do *browser* uma interface própria que renderiza o arquivo X3D.

Optou-se então por analisar mais a fundo uma tecnologia pioneira, cujo surgimento oficial data de 2011. Tal tecnologia era a API *WebGL*. A partir da *API WebGL* e mais adiante, de uma de suas vertentes mais promissoras, foi possível traduzir o código nativo *X3D* em linguagem *HTML5* mediante criação de nós (no estilo *XML*) já suportados por *browsers* como *Google Chrome* e *Mozilla FireFox*. No entanto, tal conversão não era tão simples de ser feita.

Algumas modelagens criadas de edifícios ou casas geram arquivos *X3D* nativos de centenas de linhas de código, tornando-se inviável a tradução para *HTML5*. Diante deste problema uma ferramenta oferecida pela própria desenvolvedora da biblioteca *X3DOM* oferece a solução para a conversão rápida e simples de uma arquivo *X3D* nativo para a codificação HTML5 sob o seguinte revés: modelagens complexas que envolvem muitos

vértices e possuem dezenas de coordenadas não são traduzidas corretamente na maioria dos casos. Assim, para um protótipo inicial, a criação das modelagens se deu com base em manipulação de texturas aplicadas a objetos simples, no intuito de resolver tal impasse.

Abaixo, o Quadro 1 demonstra o comparativo entre diversas modelagens feitas para o condomínio Vila Verde. Para os testes foi utilizado um *Notebook* com processar Intel Corei5, Radeon 4500HD com 8Gb de memória. Percebe-se que mesmo com um *hardware* teoricamente "robusto" dependendo do tamanho da modelagem em quantidade de nós, o computador simplesmente para de funcionar. A tabela ainda demonstra como foi importante o trabalho de otimização das modelagens por meio de trabalho com texturas (Exemplo "Cubos Dançantes" na Figura 13).

Quadro 1. Comparativo entre algumas modelagens criadas em linhas de código e tamanho dos arquivos (apenas com variação no número de objetos).

	Linhas (*X3D*)	Linhas (*HTML5*)	Tamanho dos Arquivos	Resultado
Cubos Dançantes	259	129	7kb / 7kb	Funcionamento sem problemas
Vila Verde 1 Prédio (Sem Otimização)	3100	1372	79kb / 67kb	Lentidão
Vila Verde 6 Prédios (Sem Otimização)	18402	8117	493kb / 414kb	Travamento do computador
Vila Verde 6 prédios otimizado	2592	1180	72kb / 61kb	Funcionamento sem problemas

Com a utilização das *API WebGL* e *X3DOM* foi possível atribuir à aplicação as modelagens 3D como qualquer outro componente da mesma. No entanto, o desafio passou a ser a criação de possibilidade de navegação do usuário através destas modelagens.

Os *browsers*, por padrão, apresentam recursos de navegação para as modelagens 3D criadas a partir da codificação HTML5 obtida através das bibliotecas *WebGL*. Tal navegação consiste em, ao posicionar o mouse sobre o cenário e pressionar o botão esquerdo ao mesmo tempo em que movimentá-lo para qualquer direção, percebe-se que a visão da cena muda, como se o usuário estivesse andando pela "casca" de uma esfera oca e sua visão está sempre concentrada no centro da esfera, mudando no entanto a posição pela

qual o centro é visualizado (Figura 8). O centro da esfera é o *viewpoint* atual do usuário. Ao pressionar o botão esquerdo do mouse e movimentá-lo a visão será deslocado segundo as arcos da esfera sempre apontando a visão para o centro.

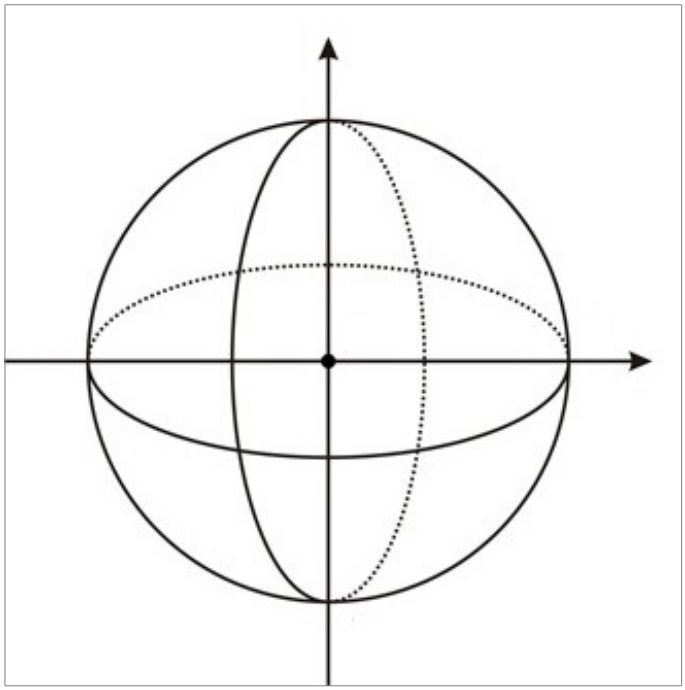

Figura 8. Representação de uma esfera de visão das modelagens 3D em *browser*.

Ao pressionar o botão direito e ao mesmo tempo movimentar o mouse, o usuário movimenta-se para a esquerda, direita, topo ou rodapé da modelagem segundo o plano atual de visão. Por fim, a aproximação (*zoom*) da modelagem se dá através do botão de rolagem (disponível na maioria dos mouses). Esta navegação trivial não é o suficiente para permitir a sensação de liberdade almejada, mesmo porque, na prática é muito difícil de se acostumar com ela. Sendo assim, buscou-se alternativas para manipular programaticamente a navegação pelos cenários 3D utilizando apenas codificação *HTML5*.

A partir da criação das modelagens em nós *HTML* através das bibliotecas *WebGL* e *X3DOM* torna-se possível manipular cada elemento 3D como um elemento *HTML* qualquer. Sendo assim, para proporcionar melhores condições de navegação, bastou criar *scripts* (baseando-se na biblioteca *JQUERY*) que manipulassem tais objetos. Para esta questão da navegação a manipulação dos *scripts* focou-se nos nós "*ViewPoint*" das modelagens (Figura 9). Por exemplo, quando o usuário clica em um botão, o *script* específico é disparado e a partir dele o *ViewPoint* do cenário muda, dando a impressão de

navegação, de acordo com a opção selecionada. Como as modelagens passaram a ser construídas segundo nós HTML, os botões contidos no cenário também são construídos dessa maneira, com suporte a scripts, possibilitando a navegação pelo cenário apenas com um clique.

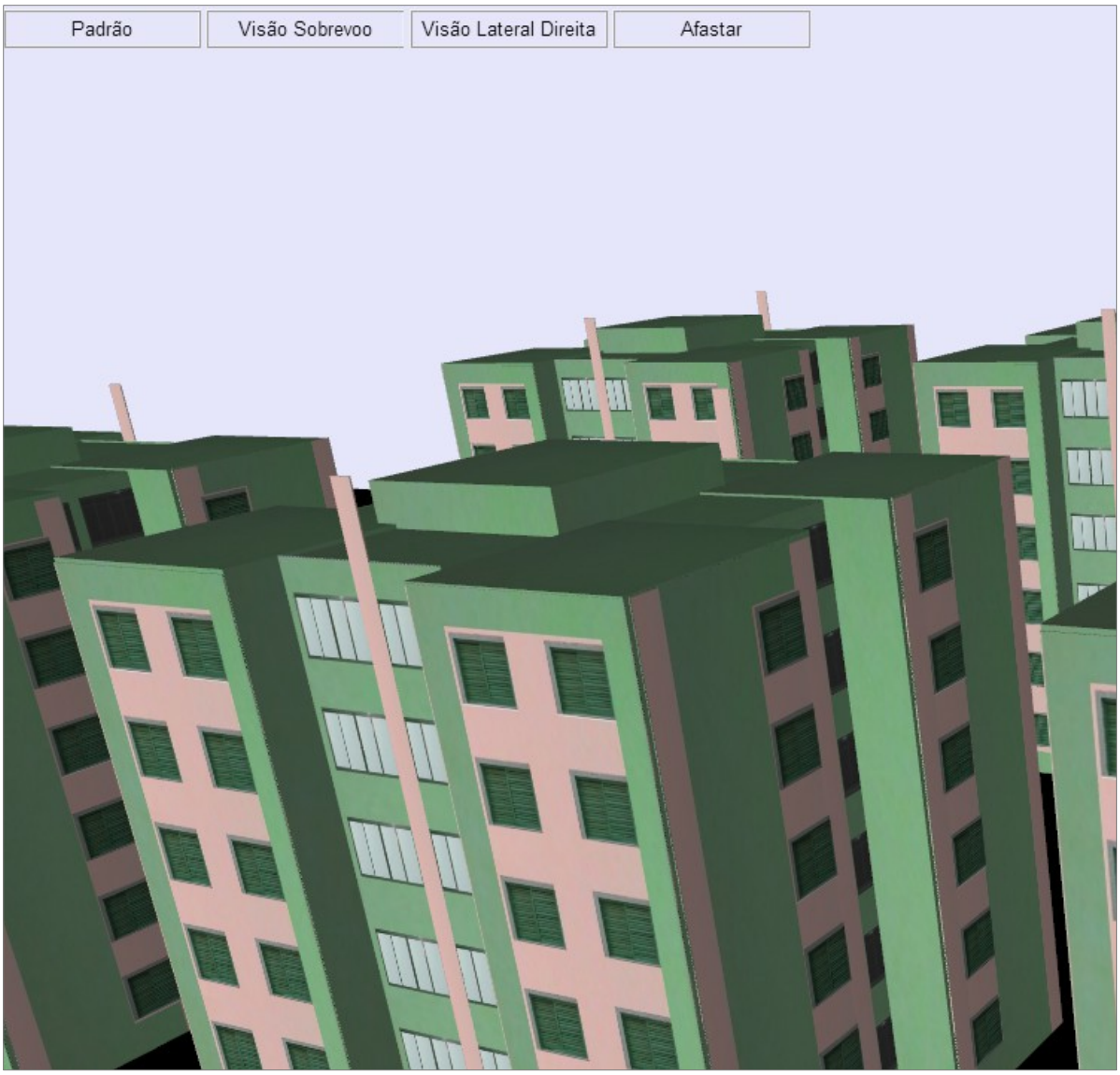

Figura 9. Representação dos botões de navegação no cenário tridimensional.

Para finalizar as pretensões iniciais quanto a interação do usuário com os elementos e cenário 3D buscou-se a possibilidade de interagir elementos HTML tradicionais (botões, *links* e elementos de página) com os elementos 3D (como um único cubo, um único prédio modelado, ou mesmo um cenário completo). Para tanto, um cuidado deve ser tomado no início do processo de modelagem.

Quando o elemento ainda está sendo criado por meio do *FluxStudio* ou *Blender* convém dar-lhe nomes ou IDs específicos. Isso faz com que seja simples mapear o código HTML da modelagem (o qual geralmente é bem grande, mesmo com aplicação de técnicas para melhorar o desempenho e reduzir o tamanho dos arquivos) e localizar o elemento específico ao qual deseja-se aplicar a integração da aplicação.

Um exemplo muito claro disso, é a associação de *links* de informações a respeito de um estabelecimento à um elemento ou objeto 3D específico como a janela de um prédio, ou o edifício todo (elemento grupo) (Figura 10). Para que isso seja possível, aplica-se um *script* que redireciona a aplicação para um página de descrição se, e somente se, o objeto 3D vinculado a tal *script* for selecionado (código *Script*).

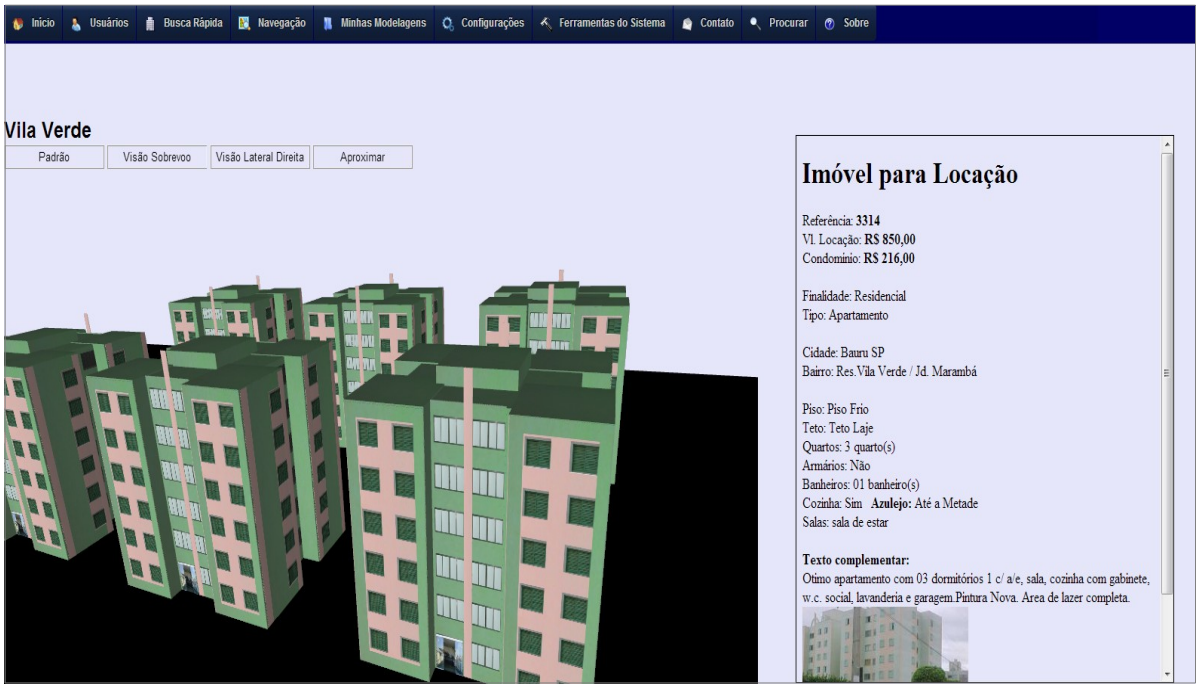

Figura 10. Exemplo de página de descrição aberta ao clicar no pilar central da primeira torre ao fundo e à esquerda.

Existem dois tipos de usuário no sistema *CityFreedom*. O primeiro é o usuário comum, que apenas navegará através da cidade virtual, podendo acessar qualquer modelagem apenas no âmbito de visualização. Por outro lado há o usuário que possuirá uma conta vinculada às "suas modelagens". Um cliente solicita a modelagem de sua estabelecimento para que outras pessoas possam conhecê-lo pela *Web*, fazendo uso do sistemas proposto este projeto. Com isso uma conta de usuário será criada e vinculada às informações da modelagem solicitada. Assim o usuário poderá controlar e fazer

configurações para a maneira como sua estabelecimento será mostrado no portal da aplicação (esta funcionalidade completa envolve os trabalhos futuros e não é tratada completamente nesta proposta).

Com a manipulação das modelagens 3D em mãos a ponto de fornecer ao usuário navegação mais intuitiva e diversificada, começou-se a pensar na estruturação do sistema quanto às camadas de navegação. Obviamente é inviável modelar a cidade toda e renderizá-la de uma vez só, o arquivo *X3D* teria um tamanho na ordem de centenas de *Megabytes* (considerando o a grandeza do município de Bauru e o tamanho da modelagem de um condomínio, usado como exemplo para o protótipo do *CityFreedom*). Para resolver este problema foi criado um sistema de navegação por níveis. Primeiramente tem-se mapa do município de Bauru (poderia ser outra cidade qualquer). Este mapa é dividido em regiões, e cada região pode ser "aproximada" dando acesso a mais regiões como num efeito de *zoom*.

Depois de três ou quatro telas, o usuário se deparará com as modelagens em si, mas para que isso seja possível ele terá de navegar num primeiro momento através destes mapas. Estas telas especiais de mapas foram chamadas de "Mosaicos". Cada mosaico tem um ID de identificação no banco de dados do sistema *CityFreedom* e as páginas ASP correspondentes a cada mosaico serão carregadas de acordo com tais IDs. Os IDs são atribuídos a partir do primeiro mosaico, por exemplo: Se o primeiro mosaico chama-se "Mosaico1" e o usuário selecionado o quadro da segunda coluna e segunda linha do mesmo, a página ASP invocada a partir de uma requisição ao banco de dados será a página "Mosaico122". Seguindo o mesmo exemplo, se o usuário selecionar o mosaico da terceira linha e terceira coluna a partir do já citado, será invocada a página "Mosaico12233". Sendo assim, de acordo com o quadrante do mosaico que o usuário selecionar, uma requisição será enviada ao *WebServer* para colher informações da página de acordo com a montagem dos mosaicos (Figura 11). Reparemos que as áreas que podem ser acessadas possuem uma textura diferente, indicando que ao clicar nela um novo nível de mosaico será mostrado.

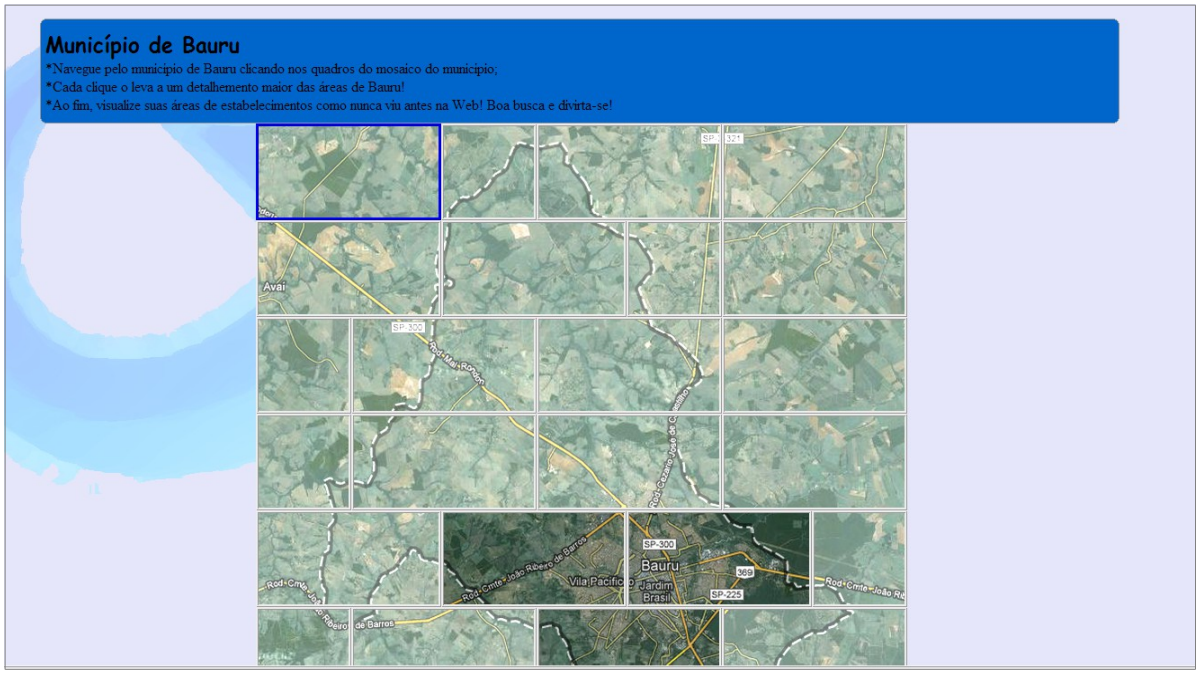

Figura 11. Mosaico inicial para o mapa do município de Bauru.

Com base na ideia de mosaicos citada acima, o usuário navega através de regiões devidamente caracterizadas e identificadas, até chegar à nível de bairro. Quando for possível selecionar um quadrante do mosaico referente a um bairro, a modelagem 3D daquele bairro é mostrada no *browser*. A partir desta modelagem de bairro (a qual naturalmente será mais simplificada) é possível selecionar um objeto 3D e a partir dele será aberta uma nova página envolvendo a modelagem específica do estabelecimento ou ponto de referência (Figura 12). Ao clicar no conjunto de prédios ao fundo o sistema abrirá a modelagem específica do condomínio "Vila Verde" anteriormente citado.

Finalmente, têm-se os recursos de tratamento de contas de usuário e pesquisa. As pesquisas serão feitas com base em filtros bem específicos, os quais também serão controlados por requisições ao banco de dados. Em outras palavras, a ser concebida uma página com modelagem de algum estabelecimento, por exemplo um bar, uma tabela no banco de dados referente ao tipo e informações da modelagem será preenchida com informações a respeito deste bar, entre elas um ID de identificação de tipo de modelagem.

Assim quando o usuário buscar por algo específico dentro do sistema *CityFreedom*, as informações do banco de dados serão cruciais para proporcionar melhor navegação e menor perda de tempo procurando por lugares pertencentes a um mesmo grupo, tipo ou classe de estabelecimento.

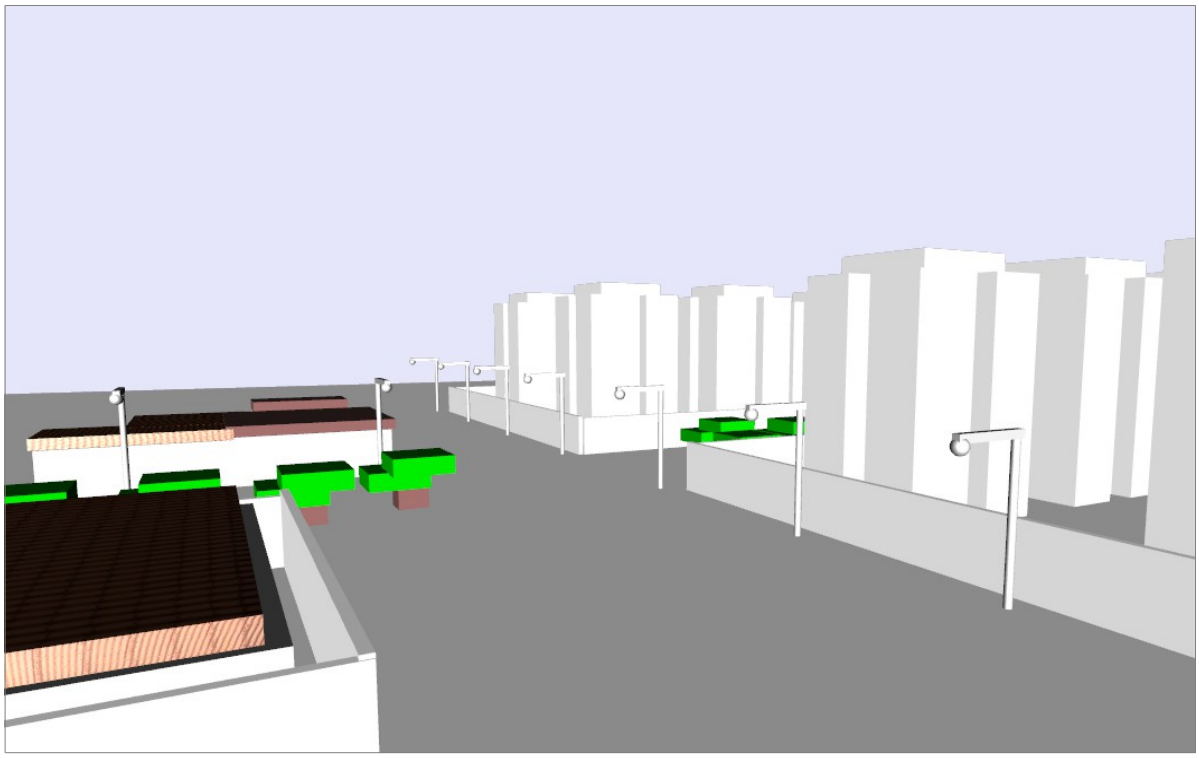

Figura 12. Representação de um bairro de Bauru (mais especificamente o Jardim Auri Verde).

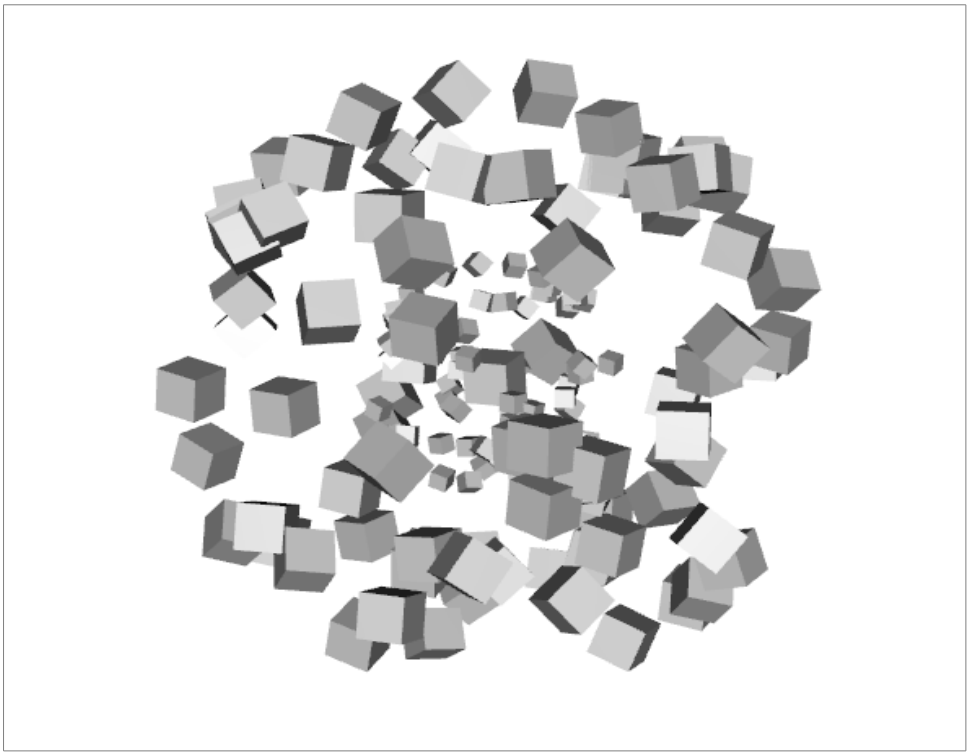

Figura 13. Cubos Dançantes. Exemplo de modelagem com movimentação. Utilizada em testes de performance.

5. RESULTADOS OBTIDOS

5.1 Desvinculação de Plug-Ins – escolha das APIs WebGL e X3DOM

A ideia inicial do projeto consistia em possibilitar integração entra aplicação *Web* e modelagem tridimensional de tal maneira que possibilitasse sentimento de liberdade ao usuário e a sensação de estar de fato conhecendo algum lugar antes mesmo de ir até ele no mundo real.

Muitas soluções foram estudadas para desvincular a aplicação *Web* dos *plug-ins* de renderização de modelagens 3D.

No fim, os mais eficientes e que de certa maneira, mais se enquadraram na ideia proposta para o projeto foram as já citadas e discutidas *APIs WebGL* e *X3DOM*. Com elas foi possível associar os objetos modelados a nós *HTML5* transformando-os em componentes da página, como qualquer outro botão ou *div*, por exemplo. Isso permitiu manipular o cenário 3D dentro da aplicação *Web* possibilitando a capacidade de navegação inicialmente imaginada.

5.2 Técnicas de Texturização, Reduzindo o Tamanho dos Arquivos X3D

Testes preliminares mostraram que a tradução da codificação *X3D* para *HTML5* gerava arquivos muito grandes, e dependendo do nível de detalhamento dos cenários, o *browser* poderia até mesmo fazer com que o computador do cliente parasse de funcionar. Para resolver este problema utilizou-se técnicas de texturização e junção de objetos para aumentar o desempenho e reduzir consideravelmente o tamanho dos arquivos.

Um exemplo disso foi a modelagem do condomínio “Vila Verde”. Num primeiro momento cada elemento do prédio foi modelado separadamente, ou seja, cada janela era um objeto 3D, por exemplo. Ao converter a codificação do condomínio da linguagem *X3D* para HTML5, o arquivo resultante era extremamente extenso e causava problemas ao ser interpretado pelos *browsers*, chegando algumas vezes a fazer o computador parar de funcionar (como citado).

Ao construir um elemento que comporte muitos outros de uma vez, e atribuindo um trabalho de montagem de texturas que substitui em apenas uma imagem o que antes era mostrado por muitas, o desempenho da aplicação aumentou consideravelmente, quase extinguindo-se os problemas anteriormente observados.

5.3 CityFreedom

As bases do projeto *CityFreedom* foram construídas de acordo com o planejado. A aplicação *Web* foi criado, os problemas quanto a união de modelagem 3D às páginas HTML foram sanado e um ambiente de navegação intuitivo e interessante ao usuário foi concebido.

Como o foco deste trabalho foi criar o sistema e garantir o seu funcionamento, pode-se considerar pouco o trabalho de modelagem comparado à grandeza de Bauru. No entanto, com os alicerces prontos basta complementar, aos poucos, com a criação de novos cenários, a base de navegação do *CityFreedom*. A ideia do projeto é muito ambiciosa e almeja fronteiras muito mais distantes, mas com certeza, os primeiros passos para que isso venha a se tornar possível fazendo uso de tantos técnicas inovadoras, já foram dados.

5.4 Integração com Servidor de Banco de Dados Através de WebServer e Requisições em Formato JSON

O projeto CirtyFreedom é um trabalho dividido em duas partes. A complementação deste projeto foi criada por Bruno Casella (CASELLA, 2012). O trabalho de Bruno Casella consiste em criar um servidor de banco de dados que se comunicará com a aplicação *CityFreedom* por meio de requisições HTTP em formato JSON (semelhante ao XML, porém possibilita maior compactação de informações).

Além disso, Bruno Casella trabalhou no conceito de criação de um *browser* próprio, com base no código aberto do *Fennec* (distribuição do *Mozilla Firefox* para dispositivos móveis) no intuito de possibilitar o melhor funcionamento do sistema em plataformas como o *Android* por exemplo. Isso possibilita a expansão da ideia de liberdade, palavra-chave e inspiração deste projeto (Figura 14).

A integração das duas partes do projeto proporciona a criação de um aplicação *Web* completo, no que diz respeito à aplicação de todas as técnicas revolucionárias e pioneiras

de programação oriundas dos estudos dos últimos anos, tudo isso unido à possibilidade oferecida de navegação e imersão em um universo 3D bem ao alcance de um simples *browser*.

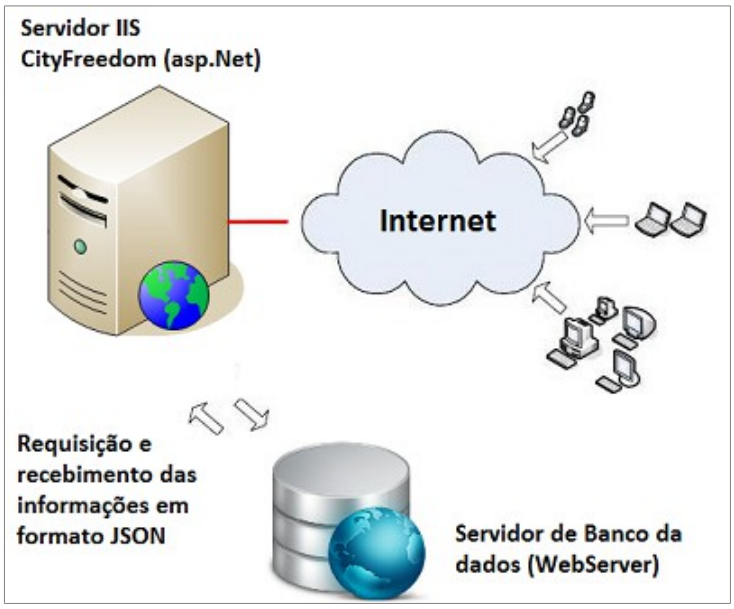

Figura 14. Estruturação do sistema. Servidor de banco de dados é também responsável pelo controle das requisições de WebService.

6. CONCLUSÕES

Este trabalho apresenta uma visão geral das novas tecnologias relacionadas ao desenvolvimento de aplicações *Web* com base em modelagem 3D e a partir deste pressuposto propõe o desenvolvimento de um projeto sob tais moldes. Nele são discutidos os mais diversos aspectos referentes aos passos, estratégias e estudos que guiaram o desenvolvimento desta aplicação. Mediante seu processo de desenvolvimento diversos novos aspectos foram comprovados ou descobertos no que diz respeito à interação entre *X3DOM*, modelagens e *HTML5*.

As técnicas aplicadas ao desempenho do sistema tiveram de ser acrescentadas, tendo em vista que o único "motor" para o funcionamento do mesmo é o *browser*. Texturização e renderização cada vez mais eficientes são necessárias para uma boa experiência do usuário ao se deparar com um projeto que une tantas técnicas novas de programação em uma só aplicação.

A utilização de modelos 3D para o uso contínuo e comum no desenvolvimento *Web* caminha a passos largos para uma revolução na maneira como os portais ou sites são vistos. Logo, com a utilização e o crescimento das técnicas aqui estudadas, será tão simples navegar por um cenário 3D através do *browser* quanto é "cotidiano" ler textos em blogs ou visualizar imagens das mais diversas.

Para o futuro, os resultados obtidos com a criação do protótipo "*CityFreedom*" expõem necessidades e problemas no desenvolvimento dos novos projetos *Web* com base em modelagens 3D e é essencial para a difusão do conhecimento, assim como a consequente e inevitável tendência de em pouco tempo algo tão revolucionário poder sofrer o mesmo processo discutido e atribuído aos *WebServices*, tornando-se um marco na criação de aplicações *Web*.

7. TRABALHOS FUTUROS

7.1. Banco de Dados e Controle de Usuários

Há melhorias a serem feitas na tratamento das informações de usuários e elas são relacionadas às modelagens de cada um destes usuários. Num primeiro momento, foi preparada a base de dados para que fosse possível demonstrar o funcionamento e a associação de um usuário ao sistema. No entanto resta criar as telas e vincular os usuários às modelagens especificas, o que possibilitará recursos como configurações e maneira de visualização dos objetos e cenários.

7.1. Banco de Dados e Controle de Usuários

Como citado anteriormente, o projeto *CityFreedom* é ambicioso e extremamente grande, se pensar-se numa conclusão total em termos de modelagem. Criar o município de Bauru inteiro com base em modelagem 3D é um trabalho árduo mas que pode ser feito através de módulos.

Entenda-se por "módulos de desenvolvimento" criar a modelagem de uma região ou bairro e incrementá-la a solução do *CityFreedom*. Cada nova modelagem dará origem a uma nova página, a qual poderá ser acessada da mesma maneira que os exemplos criados e valendo da mesmas base, dos mesmos pilares construídos nesta proposta para o sistema *CityFreedom*.

REFERÊNCIAS BIBLIOGRÁFICAS

ADAMS, L. **Virtualização e Realidade Virtual**, São Paulo, Ed Makron Books, p. 255-259.

ARAUJO, J. G. R. O **Desenvolvimento de Aplicações Web.** 1997. Disponível em: http://www1.rnp.br/newsgen/9710/n5-3.html. Acesso em: Out. 2012.

ARMSTRONG, E.; BALL, J.; BODOFF, S.; CARSON, D. B.; EVANS, I.; Dale, G.; HAASE, K.; JENDROCK, E. **ORACLE, Getting Started with Web Applications.** 2005. Disponível em: http://docs.oracle.com/javaee/1.4/tutorial/doc. Acesso em Jul. 2012.

BEHR , J. E. SCHLER , P. J. UNG, Y. A. Z. OLLNER , M. **X3DOM – A DOM-based HTML5/ X3D integration model.** 2009. Disponível em: http://www.Web3d.org/x3d/wiki/images/3/30/X3dom-Web3d2009-paper.pdf. Acesso em Jun. 2012.

BONIAT, B. B.; PADOIN, E. L. **Web services como middlewares para interoperabilidade em sistemas.** 2005. Disponível em: http://www.urcamp.tche.br/site/ccei/revista/revista12.pdf. Acesso em Jul. 2012.

BRUTZMAN, D.; DALY, L. **X3D, Extensive 3D Graphics for Web Authors.** 1ª Ed. [S.I.] Editora Morgan Kaufmann Publishes. 2007, p. 20-30.

CASELLA, B. F. **Desenvolvimento de Aplicações Web com Base em Modelagem 3D – Módulo Android.** Bauru**,** 2012, p. 11-25.

DANIEL, J. **Crafting Digital Media**. 1ª Ed. [S.I.] Editora Apress. Parte 2, p. 145-147.

DELILLO, B. **Khronnon Group. WebGL – Open GL ES 2.0 for the Web. 2009.** Disponível em: http://bjartr.blogspot.com.br/2009/10/more-Webgl-progress-now-with-video.html. Acesso em Jul. 2012.

FESTA, P; BORLAND, J. **Is a 3D Web more than just empty promises?** 2005. Disponível em www.zdnet.co.uk. Acesso em abril de 2012.

HAROLD, R. **XML Bible**. 1st Ed. [S.I.]: Editora Wiley. 1999.

KIRNER, C; PINHO, M. **Uma Introdução à Realidade Virtual**. **Jornada de Autalização em Informática. Congresso Nacional da Sociedade Brasileira de Computação. ANAIS.** Recife, Agosto 1996.

KIRNER, C. 2011. **Realidade Virtual e Aumentada: Aplicações e Tendências.** 2011. Disponível em: http://www.ckirner.com/realidadevirtual/?PUBLICA%C7%D5ES. Acesso em Jun. 2012.

MICROSOFT (MSDN). **Tecnologia ASP NET**. 2012. Disponível em: http://msdn.microsoft.com/pt-br/library/cc580600.aspx e http://msdn.microsoft.com-/pt-br/library/9k6k3k4a(VS.85).aspx. Acesso em Out. 2012.

VALÉRIO, A; MACHADO, L. **Realidade Virtual – Definições, Dispositivos e Aplicações**. 2002. Disponível em: http://www.de.ufpb.br/~labteve/publi/2002_reic.pdf. Acesso em Jun, 2012.

X3DOM Group. **API and Examples**. 2012. Disponível em: http://x3dom.org/docs/dev/. Acesso em Jun. 2012.