
TECNOLOGIA
TECNOLOGIA
TECNO LOG
OLOGIA

GAMETV: UM *FRAMEWORK* DE JOGOS DIGITAIS 2D PARA TV DIGITAL

Lady Daiana O. Maia[1],
José P. Queiroz-Neto[2],
Vicente F. Lucena Jr.[3]

RESUMO

A TV Digital ainda está iniciando a sua popularização no Brasil e são poucas as experiências consolidadas no desenvolvimento de aplicações interativas para o modelo de TV digital brasileira, principalmente no que diz respeito aos Jogos Digitais, que configuram um desafio para pesquisadores, técnicos e empresários, pois a indústria de jogos cresceu no último ano 31% nos diversos tipos de plataforma existentes. Neste cenário, a proposta desse trabalho é desenvolver um aplicativo de auxílio ao desenvolvimento de jogos, um *framework* denominado Game TV, reduzindo as dificuldades encontradas na programação em ambientes com recursos limitados, contribuindo de maneira positiva para os desenvolvedores de jogos e provedores de conteúdo televisivo.

Palavras-Chave: TV Digital, Jogos Digitais, *Framework*

ABSTRACT

The Digital TV is just starting its popularity in Brazil. Thus, the development of interactive applications for the Brazilian digital TV standard is not consolidated, especially regarding Digital Games, which represent a challenge for researchers, technicians and entrepreneurs, due to growth of 31% in last year's gaming industry in several types of platform. Hence, the goal of this research is to produce a framework to support the development of games, called Game TV, which reduces the adversities to programming in a limited resources environment and provides a positive contribution to game designers and TV content developers.

Keywords: Digital TV, Digital Games, *Framework*

[1] Mestre em Ciência da Computação. Professora do Campus Itacoatiara da UFAM, e-mail: ladydaiana@ufam.edu.br
[2] Doutor em Ciência da Computação. Professor e pesquisador do IFAM/Campus Distrito Industrial, e- mail: pinheiro@ifam.edu.br, autor para correspondência.
[3] Doutor em Ciência da Computação. Professor e pesquisador do IFAM/Campus Distrito Industrial, e-mail: vicente@ufam.edu.br

1. INTRODUÇÃO

Atualmente no Brasil, a Televisão Digital Interativa (TVDI) aberta está se tornando uma realidade, pois o governo tem dado o apoio à infraestrutura necessária para que até o ano de 2016 esta tecnologia esteja funcionando em todos os estados. Por enquanto, apenas alguns estados estão transmitindo o sinal digital e o prazo estipulado para o término da implantação da televisão digital vai até 2016, conforme disposto na Portaria do Ministério das Comunicações no. 652 de 10 de outubro de 2006. (DTV, 2010)

A TVDI visa proporcionar a democratização do acesso à informação para a população nos próximos anos, por meio da interatividade das aplicações desenvolvidas, fazendo uso da convergência tecnológica de acordo com o Sistema Brasileiro de TV Digital. No entanto, apesar de completar um ano do início da implantação, são pequenas as iniciativas relacionadas ao desenvolvimento de aplicações para compor esse novo cenário.

Em se tratando de jogos para TV digital, Ferreira & Souza (2008) afirmam que as iniciativas são menores ainda, pois em um jogo eletrônico é necessário uma série de recursos computacionais para que seja possível atender as expectativas do usuário e, além disso, possui algumas limitações impostas pelo ambiente, como por exemplo, a *interface* com o jogo (controle remoto), a escassez de ferramentas para auxiliar o desenvolvimento e ainda a falta de uma estratégia para projeto de jogos.

Uma das maneiras interessantes de se utilizar a TV digital para fins educacionais é a utilização de jogos, que deve ser considerado como uma ferramenta auxiliar ao ensino. Ao se pensar na utilização dos jogos educacionais é necessário primeiramente defini-lo como uma ferramenta que facilita o aprendizado e aumenta a capacidade de retenção do que foi ensinado. A sua utilização possui o intuito de despertar o interesse no aprendizado, ou seja, se tornar um elemento motivador no processo de ensino-aprendizagem em crianças e jovens.

Nesse contexto, observa-se a necessidade da criação de recursos para suporte ao desenvolvimento dessas aplicações, pois além dos jogos exercerem um poder atrativo e motivador, pode possibilitar a inclusão social, que é um dos objetivos do Sistema de TV digital do Brasil, através de aplicações *T-learning, T-commerce e T-government*, que são aplicativos em TVDI voltados à educação, comércio e ações do governo, respectivamente.

Neste trabalho, será apresentada a tecnologia de TV Digital brasileira e os requisitos necessários para o desenvolvimento de jogos para esse ambiente, levando em consideração as especificações do *middleware* Ginga, a fim de propor uma arquitetura para um *framework* de jogos digitais capaz de gerar jogos para serem executados em sistemas televisivos digitais.

2.TECNOLOGIAS DE DESENVOLVIMENTO DE JOGOS PARA TV DIGITAL BRASILEIRA

A convergência da televisão com o universo dos jogos digitais pode aumentar ainda mais os índices das indústrias de jogos e das emissoras de TVs, especialmente pelas vantagens que a TV Digital oferece. O Reino Unido é líder em penetração de TV digital e desenvolvimento de serviços para TV interativa, apresentou em março de 2003, um índice de 1,3 milhões de utilizadores de jogos disponíveis na *Sky Digital*, desse índice 40% eram mulheres. O canal de jogos *PlayJam* da *Sky Digital* consta entre o 8º e o 15º lugar dos canais mais vistos do serviço da *Sky*, de acordo com o *Broadcast Audience Research Board* -BARB, o órgão oficial no Reino Unido para medição de audiências.

Um *framework* de jogos é na verdade uma plataforma computacional, um aplicativo, que tem como objetivo auxiliar a criar padrões de projeto típicos de jogos, facilitando a construção da *interface* gráfica, tratamento de eventos e o controle da navegação. No

desenvolvimento de um *framework* busca-se pela característica generalidade em relação aos conceitos e funcionalidades do domínio, pois um *framework* é uma abstração de um domínio de aplicações. Além disso, são necessárias as características de alterabilidade (alterar as funcionalidades presentes) e extensibilidade (ampliar a funcionalidade presente) para o projeto de um *framework* (ASSIS & SUZANO, 2003).

Para a construção de um *framework* deve-se produzir uma estrutura de classes tendo a capacidade de adaptar-se a um conjunto de aplicações diferentes, porém no mesmo domínio, tendo em vista que é necessário dispor de modelagens de um conjunto significativo de aplicações do domínio.

Mais especificamente, a utilização de *framework* de jogos permite fornecer um conjunto de componentes projetados especialmente para estes dispositivos, visando a reusabilidade de código e ainda a diminuição do tempo de desenvolvimento do jogo. Logo, os componentes comuns do jogo estão prontos no *framework*, permitindo a reutilização dos mesmos e com isso um desenvolvimento mais rápido do jogo.

No entanto, existem algumas desvantagens na utilização do *framework*. Segundo (SILVA, 2000), uma das grandes desvantagens no uso dos *framework*s ocorre devido ao fato de que os jogos podem ser completamente diferentes uns dos outros. Deste modo, utilizar uma arquitetura muito geral, que busque atender a todos os tipos de jogos, fatalmente irá causar um overhead que poderá inviabilizar a construção de jogos, ou ainda criar uma arquitetura especialista servirá apenas para um determinado tipo de jogo, o que foge ao propósito do *framework* de ser reusável. Portanto, é preciso atenção no momento do projeto do *framework*, evitando-se construir algo muito geral ou específico.

O sistema de televisão é composto pelo estúdio, radiodifusão e sistema de recepção, responsáveis pelo processo utilizado na TV digital, que consiste respectivamente na informação gerada pelo estúdio, a transmissão dessas informações e a recepção dos sinais na televisão de maneira digital.

O Sistema Brasileiro de Televisão Digital (ISDTV-T) é baseado no padrão japonês (ISDB), é uma tecnologia que reúne os recursos de mobilidade, portabilidade, alta definição, transmissão de dados e segmentação que é a subdivisão do canal permitindo a transmissão simultânea de programas no mesmo canal.

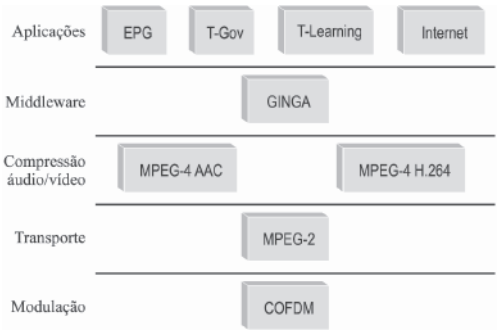


Figura 1: Camadas de Tecnologia do ISDTV-T

De acordo com a Figura 1, o ISDTV-T é formado por um conjunto de componentes relacionados à aplicação, codificação do sinal, multiplexação e transmissão, comum ao padrão ISDB. Em (PINTO et. al., 2007) são descritas as camadas de aplicação, *middleware*, compressão, transporte e modulação. A camada de aplicação é onde acontecem as entradas de informações de vídeo, áudio e novos serviços interativos, representados pelos dados. Na camada de codificação do sinal, os sinais de áudio e vídeo são compactados por meio dos sistemas MPEG-4 ACC e o MPEG-4 H.264 respectivamente. A *interface* responsável pela implementação de interatividade e novos serviços interativos da ISDTV, chamada de *middleware*, é o GINGA (LEMOS et. al., 2006). Este *middleware* foi elaborado pelo desenvolvimento de projetos de pesquisa coordenados pelos laboratórios Telemídia da PUC-Rio e LAViD da UFPB. Na camada de multiplexação ocorre à transformação da união dos sinais em um feixe, o qual será transmitido pela camada de radiodifusão utilizando a modulação BST-OFDM.

O desenvolvimento de aplicações para TV digital concetra-se na camada do *middleware* . Na Figura 2 está representado o modelo de referência do *middleware* Ginga, onde a arquitetura do Ginga pode ser dividida em três módulos principais: Ginga-CC(Ginga Common Core) é o subsistema lógico que provê todas as funcionalidades comuns ao suporte dos ambientes declarativo, Ginga-NCL, e imperativo, Ginga-J.; os módulos Ginga-NCL e Ginga-J que compõem a camada de Serviços Específicos do Ginga.

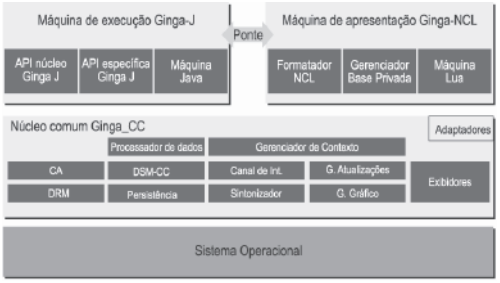


Figura 2: (A) modo correto para secagem. (B) modo incorreto para a secagem.

Para Soares (2007), "Ginga-NCL é o subsistema Lógico do *middleware* Ginga que processa documentos NCL". O formatador NCL recebe um documento NCL e controla a sua apresentação, sincronizando os objetos de mídia, fazendo com que elas sejam apresentadas no momento programado.

As linguagens declarativas seguem um alto nível de abstração, normalmente ligada a um domínio ou objetivo específico. Neste modelo, o deverão ser executadas, não se preocupando com os detalhes de como o interpretador, compilador, máquina real ou virtual fará este procedimento. Dentre as linguagens declarativas mais comuns para a TV Digital, pode-se citar a NCL e o XHTML (BML-ARIB, DVB-HTML e ACAP-X) (ZANCANARO et. al.,2009).

Em relação às linguagens procedurais, Zancanaro et. al. (2009) diz que é onde o programador tem maior poder sobre o código, informando cada passo o que será executado. A linguagem utilizada para desenvolver sistemas de TVD nesse paradigma é a linguagem Java. (BARBOSA & SOARES, 2008).

A plataforma Ginga-J foi projetada para suprir todas as funcionalidades necessárias para a implementação de aplicativos para televisão digital utilizando API's, *Application Programming Interface* (ou *Interface* de Programação de Aplicativos) que é um conjunto de rotinas e padrões estabelecidos por um *software* para a utilização das suas funcionalidades por programas aplicativos que não querem envolver-se em detalhes da implementação do *software*. As API's são utilizadas para o processamento de classes java compiladas, que são consideradas como componentes, cada uma definida para um tipo de serviço, conforme descrito em (SILVA, 2004).

As aplicações do Java TV, que são desenvolvidas na linguagem Java para a TV Digital, são denominadas *Xlets*. Analisando analogicamente, um *Xlets* é similar a um *applet* (*web*) ou *midlet* (celular), e possuí um ciclo de vida que é composto por quatro estados, conforme (LEMOS et. al., 2004): Carregado (*loaded*), Pausado (*paused*), Ativo (*active*) e Destruído (*destroyed*).

Para o gerenciamento do ciclo de vida de um *Xlet*, o Java TV utiliza um gerente de aplicação (*application manager*), que pode modificar o seu estado, assim como também o próprio **Xlet**, que notifica o gerente de aplicação quando muda de estado.

O ciclo de vida de um *Xlet* se inicia quando instanciado pelo gerente de aplicação usando o método *intXlet( )*; Nesse estado *Loaded*, o gerente passa ao *Xlet* o objeto *XletContext* (canal de comunicação entre o *Xlet* e o gerenciador de aplicação) que define o contexto de execução, podendo enviá-lo novamente para o gerenciador notificando sobre transição do seu estado.

Após a inicialização, passa para o estado *paused*, que pode ser modificado para o estado ativo, através do método *startXlet( )*; que por sua vez pode voltar ao estado *paused* utilizando o método *pauseXlet( )*. e por fim vai para o estado *destroyd*, usando o método *destroyXlet( )*.

3. PROJETO DO *FRAMEWORK* GAMETV

Esta seção relaciona-se com as etapas necessárias para a criação do GameTV, sendo definida pelo levantamento da arquitetura, análise e modelagem, implementação do *framework* e as atividades de cada fase respectivamente. Tal etapa é fundamental para o desenvolvimento do trabalho, pois se trata de forma conceitual da modelagem do *framework*, requisitos e ferramentas utilizadas no seu desenvolvimento, como apresentado na Figura 3.

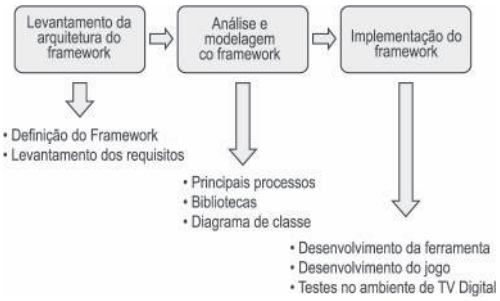


Figura 3: Desenvolvimento do *Framework* GameTV

O *framework GameTv* é uma ambiente de trabalho para geração de jogos, consiste em um conjunto de classes básicas que permitem a criação de novos jogos, apenas sendo necessário ao usuário adicionar os objetos e cenários para a criação dos jogos, através de uma *interface* gráfica intuitiva.

As principais funcionalidades oferecidas pelo *GameTV* é a exibição de imagens, utilizando mecanismo de visualização em 2D, reprodução de música e som, tratamento de eventos gerados pelo controle remoto e o processamento da lógica do jogo através de ações. As ações representam modificações sob os objetos do jogo. O foco do *Framework* é facilitar a criação de *interfaces* gráficas dos jogos e prover suporte para execução de jogos na TVD. Além disso, não é necessário que o usuário conheça alguma linguagem de programação, o que facilita a sua utilização.

A Figura 4 demonstra a caracterização do *Framework* apresentando os dois blocos responsáveis para atender o objetivo proposto. Sendo o primeiro bloco o *framework*, que consiste na ferramenta gráfica de criação de jogos e a *Xlet* gerada, que pode ser utilizada em uma plataforma computacional, e o segundo bloco representado pelo padrão de TV digital, com as camadas de aplicações, *middleware*, sistema operacional e *hardware*. Isso significa que o suário poderá gerar o *Xlet* em um computador utilizando a ferramenta do *framework*, transmitir o código Xlet pelo protocolo IPTV (*Internet Protocol TV*) e executá-lo no *Set-top-box* (STB).

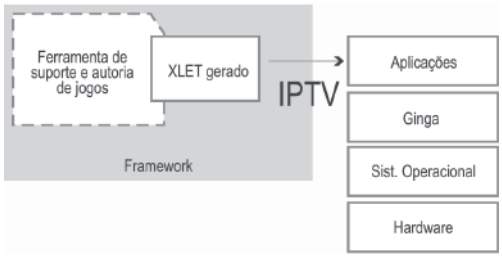


Figura 4: Diagrama de blocos do *Framework GameTV*

A elaboração do projeto da arquitetura do *framework* para jogos em TV digital consistiu no estudo dos requisitos necessários e as características que um *framework* deve oferecer, dentro das limitações impostas pelo ambiente da TV digital. Através desse levantamento foi possível identificar características a serem compostas e propor uma arquitetura conceitual para o *GameTV*.

Os requisitos definidos para o *framework* consistem em:

• Prover suporte a implementação de jogos a serem executados na TV digital ou STB;

• Desenvolver o *framework* utilizando o paradigma de orientação a objetos e técnicas de engenharia de desenvolvimento de *software*;

• Possuir uma licença de *software* livre, para compartilhar os conceitos e experiências com a comunidade;

• O *framework* funcionará inicialmente na plataforma *Windows*, mas poderá ser ampliado para suporte ao sistema operacional *GNU/Linux*;

• A implementação do *framework* foi na Linguagem Java, devido a mesma oferecer portabilidade, alto desempenho, orientada objetos e dinâmica.

O ciclo de vida de um jogo, segundo (ROCHA, 2003) consiste de um laço continuo que executa uma determinada lógica e apresenta o resultado ao usuário, conforme apresentado na Figura 5.

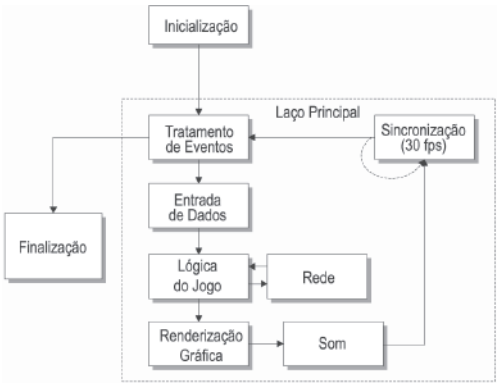


Figura 5: Ciclo de vida de um jogo.

O ciclo de vida possui a inicialização como responsável pela preparação para executar o jogo, o laço é responsável pelo gerenciamento e execução dos componentes da aplicação, como o tratamento de eventos, a entrada de dados, lógica do jogo e renderização gráfica, até a finalização do jogo, onde são liberados todos os recursos que foram empregados.

A arquitetura definida para o *GameTV* baseouse em uma arquitetura de *framework* de jogos para dispositivos móveis, o qual também possuí algumas limitações sobre a plataforma. Observou-se a necessidade de definição dos módulos e a interação entre eles. Esses módulos são responsáveis por gerenciar todos os eventos que acontecem durante o jogo, o conjunto dos módulos forma a arquitetura do *framework*.

A arquitetura definida para o *framework GameTV* está explicita na Figura 6, composta pelos módulos e uma breve descrição da responsabilidade de cada um.

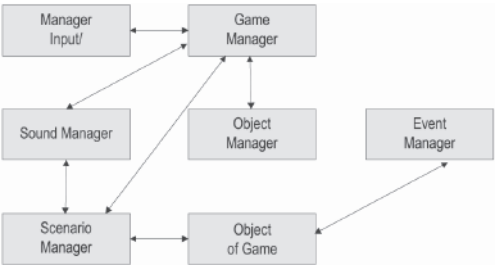


Figura 6: Modelo da arquitetura do *Framework*.

Os módulos sugeridos possuem as seguintes funcionalidades:

• *Game Manager:* O módulo gerenciador do é o modulo principal sendo responsável por criar, gerenciar, destruir os demais módulos e as exigências ao computador. E ainda gerar o ciclo de vida do *Xlet* para a execução no STB;

• *Manager Input/Graphic:* O módulo é responsável por identificar os eventos do controle remoto e ainda permitir uma representação gráfica, desenhando os elementos gráficos do jogo na tela da TV, permitindo as imagens 2D;

• *Sound Manager:* É o módulo responsável pela a reprodução dos sons e efeitos sonoros;

• *Event Manager:* Possui a responsabilidade de gerenciar os eventos que estão relacionados aos objetos do jogo;

• *Object Manager:* Responsável por gerenciar os objetos do jogo e controlar o seu ciclo de vida e a detecção de colisão;

• *Object of game:* Contém os atributos dos objetos pertinentes ao jogo, como posição, velocidade, etc;

• *Scenarios Manager:* Responsável pelo cenário do jogo e contém os personagens e os objetos de cada fase.

Uma das principais diferenças entre

o desenvolvedor dos jogos para PC e jogos para a TV digital é a forma de lidar com a plataforma gráfica. Desenvolvedores de ambientes de TVD têm de estarem conscientes dos diferentes modelos gráficos, as questões envolvidas na configuração dos dispositivos de visualização e integração de vídeo em suas aplicações, assim como a necessidade de conceber uma *interface* que funcione bem dadas às limitações de uma tela de TV (MORRIS & SMITH-CHAIGNEAU, 2005).

A *interface* é fundamental para que um jogo possa ser aceito, por isso há necessidade de considerar alguns pontos relevantes pra o desenvolvimento de *interfaces* gráficas relacionas a TV (MORRIS & SMITH-CHAIGNEAU, 2005):

Pontos na tela: As imagens exibidas na TV são formadas por pixel enquanto as geradas pela API gráfica são pontos quadrados, ocasionando uma variação de dimensão e uma mudança na resolução do receptor;

Variação da exibição: O tamanho de exibição em uma TVDI pode variar entre o tamanho padrão 4:3 e o widescreen 16:9 ou ainda 14:9, gerando alterações no tamanho dos objetos gráficos;

Transparência: A utilização de gráficos transparentes para que o usuário veja o que está por trás;

Cores: O espaço de cor do sinal digital é o espaço YUV enquanto gerados pela API gráfica é o espaço RGB.

A escolha da API gráfica para a criação das *interfaces* foi determinada pela a análise dos problemas citados acima e possíveis soluções utilizando o conjunto das API's Havi, que é responsável pela criação e gerência do uso de *interface* gráfica no ambiente de TV digital.

É importante frisar que na TV digital, devido limitações de recursos computacionais, os jogos são desenvolvidos em tecnologia 2D, onde o jogo possui objetos com duas dimensões, tendo assim largura e comprimento. Sobre a tecnologia de jogos 2D é possível especificar texturas de textos e imagens representadas em jogos através de objetos do tipo *sprite*.

Os *sprites* consistem em uma série de imagens que ao serem exibidas rapidamente em sequência produzem efeitos de animação. Tais *sprites* podem representar os personagens do jogo que interagem uns com os outros e com o cenário, podendo inserir ações de pular, voar, nadar, lutar e atirar. Geralmente, o jogo fornece um *sprite* que representa o personagem principal controlado pelo jogador. Os outros *sprites* podem ser monstros e inimigos que perseguem o jogador, ou objetos que o jogador pode pegar, tal como moedas e armas (ALMEIDA, 2004).

Os jogos que utilizam *sprites* também são conhecidos como jogos de plataforma, que é o nome dado a um gênero de jogos de *video-game* em que o jogador corre e pula entre plataformas e obstáculos, enfrentando inimigos e coletando objetos bônus. Entre os jogos de plataforma podemos destacar o *Pac-Man* e o Super Mario, bastante conhecidos dos usuários.

Com o intuito de auxiliar ao processo de desenvolvimento da arquitetura foi realizado o levantamento dos principais processos em relação as funcionalidade no *framework*. Tais processos apresentam uma simplificação da funcionalidade do sistema, sendo que cada processo poderá envolver uma ou mais classes de controle de acordo com análise realizada. Na fase de análise definiram-se quatro processos básicos:

• Processo principal: É o processo de inicialização do sistema e coordenador dos demais processos, responsável pela criação do ciclo de vida do *Xlet*.

• Processo de adicionar objetos: é o processo responsável pela criação de objetos como texto, imagem, som. Permitindo o carregamento dos objetos e manipulação em memória.

• Processo de tratamento de eventos: responsável por todos os eventos de entrada e saída do sistema, pela interação do usuário com o jogo. Ele que permite colocar ações aos objetos.

• Processo de conversão do jogo para TV digital: é o processo fundamental, ele que permite a inserção de cada objeto e realiza o processamento deles no *Xlet* para execução do jogo no STB.

Para o desenvolvimento das aplicações *Xlets* realizou-se um levantamento sobre as bibliotecas necessárias para o desenvolvimento de aplicações para TVD, as quais funcionariam de acordo com as especificações do Ginga-J. Tais especificações são utilizadas para garantir a compatibilidade com o padrão GEM (*Globally Executable* MHP). O qual trata de uma série de APIs comuns aos *middlewares* existentes mundialmente, sendo possível uma interoperabilidade entre os sistemas (*middleware* GEM).

O Ginga-J foi especificado de forma compatível com o padrão GEM, e possui um conjunto de APIs específicas para o ISDTV-T. Desta forma é possível a construção de aplicações que possam ser executadas em qualquer *middleware*.

4 GAMETV: *FRAMEWORK* DE JOGOS DE PLATAFORMA PARA TV DIGITAL

Nesta seção é apresentado o desenvolvimento do *framework GameTv* para a criação de jogos para TV digital, conforme os principais processos funcionais correspondentes aos módulos citados na seção anterior (Figura 6). A implementação ocorreu através do desenvolvimento dos seguintes módulos: *Game Manager, Manager Graphics, Sound Manager, Event Manager, Object Manager e Scenarios Manager.*

O processo principal envolve a classe Principal necessária para a inicialização do sistema. Isso significa que, através dela todas as demais classes são instanciadas. Nesse processo também é utilizada a classe que é responsável pela *interface New Project*, apresentada na Figura 7, e por criar logicamente e fisicamente o novo projeto no diretório escolhido.

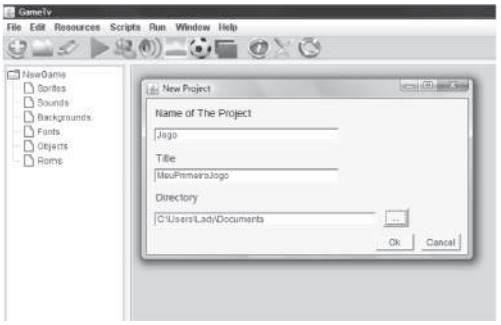


Figura 7: *Interface* Principal e a *New Project*

Na *interface "New Project",* ao iniciar um novo projeto deverá ser inserido o nome do projeto, título e o diretório onde será armazenada a estrutura de pastas semelhante à árvore apresentada do lado esquerdo da Figura 7. O nome do projeto é utilizado para o nome da pasta que o sistema criará para armazenar todos os objetos do jogo, e o título é dado à raiz da árvore de diretórios do aplicativo e também nomeado ao *Xlet* que será construído pelo sistema. Caso o usuário não digite o nome do projeto, ele utilizara uma estrutura de pastas padrão, de acordo com a Figura 8, onde se tem a pasta *temp* que armazenará a pasta *Sprites, Sounds, Backgrounds e Xlet*, contendo os *sprites,* sons, *backgrounds* e o *Xlet* do jogo, respectivamente.

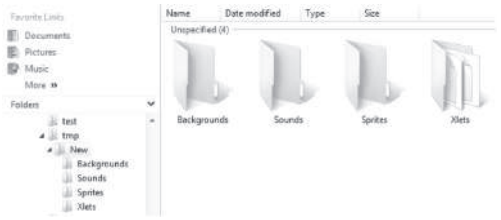


Figura 8: Estrutura de pastas default geradas.

O processo principal está relacionado com o módulo *GameManager* da aquitetura proposta, o qual é encarregado de lidar com a instanciação e destruição dos

demais módulos. Sendo responsável pelo gerenciamento do código do *Xlet*. Através do *GameManager* é possível acessar todos os outros módulos do *Framework*, com isso essa é a principal *interface* entre o usuário e o jogo em si.

O Processo que adiciona todos os objetos para a criação do jogo é responsável por permitir que as imagens sejam desenhadas na tela do computador, carregando as imagens e sons e manipulando em memórias. Esse processo é representado pelas classes *createSprite*, *createSound, createBackground, createObjeto*s e *createRoom* que correspondem respectivamente a inserir *sprites*, sons, *backgrounds*, objetos e cenários.

Esse processo envolve o módulo *Manager Graphics* para inserir os *sprites*, *backgrounds* no cenário do jogo, gerando uma representação gráfica, e ainda o módulo *Sound Manager* para inserir os arquivos de áudio. Além disso, no cenário pode-se inserir textos e objetos criados pelos *sprites*.

A Figura 9 representa a *interface Create Sprites*, com ela é possível adicionar as imagens com formato .jpg, .jpeg, .png e .gif que poderão ser utilizadas no jogo. Após a inserção da imagem escolhida, a mesma aparecera do lado esquerdo do aplicativo na estrutura de pasta correspondente ao diretório *sprites*.

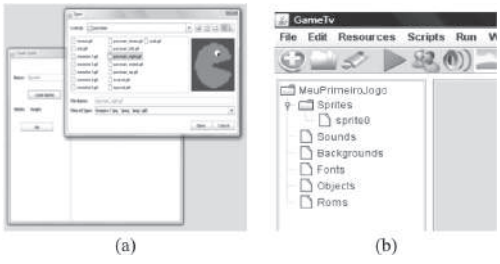


Figura 9: Processo de inserir um *sprite*. Em (a) Adicionando uma imagem como *sprite*, e em (b) *sprite* inserido e pode ser visualizado na árvore a esquerda.

Para atualizar a imagem adicionada basta um duplo clique no nome dela e aparecerá novamente o formulário para mudar o *sprite*. A imagem será adicionada fisicamente no diretório *sprite*.

Com a *interface Create Sound* é possível inserir os arquivos sonoros, a classe *createSound* é responsável por permitir a leitura e reprodução dos sons, conforme mostrado na Figura 10. Além disso, controla o parâmetro volume e é representado pelo módulo *Sound Manager*.

A *interface Create Room* é utilizada para definir os níveis do seu jogo ou os cenários diferentes. No cenário é possível ter um fundo, conhecido como *background*, que pode ser uma cor ou uma figura. A figura pode ser carrega a partir de um arquivo na *interface Create Background*. Ao criar um cenário é necessário posicionar os objetos na tela, correspondente aos objetos do jogo. Por exemplo, adicionar um objeto "pedra" e usá-lo em vários locais. E Ainda ter várias instâncias de um objeto "nave" e todas elas terão o mesmo comportamento.

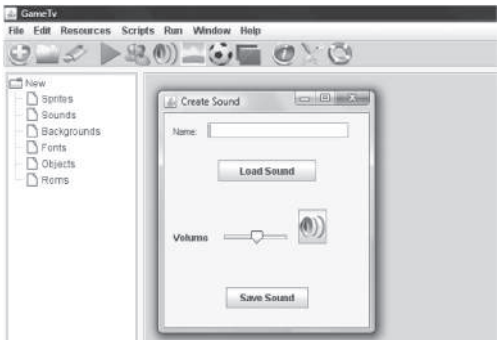


Figura 10: *Interface Create Sound.*

Os cenários são as entidades onde o jogo acontece. Todo jogo necessita de pelo menos um cenário e são colocados os objetos iniciais de um jogo e o plano de fundo. Além disso, podem-se adicionar textos no cenário. Na janela *Create Rom,* correspondente a Figura 11, temos as propriedades da *Room*, onde são visualiza-

das as abas *Settings, Objects, Background e Titles,* sendo que em *Settings* é a parte de configuração da janela, em *Objects* escolhe-se os objetos a serem adicionados, *Background* para inserir um fundo ou cor de fundo e os *Titles* que permitem inserir textos e posicioná-los.

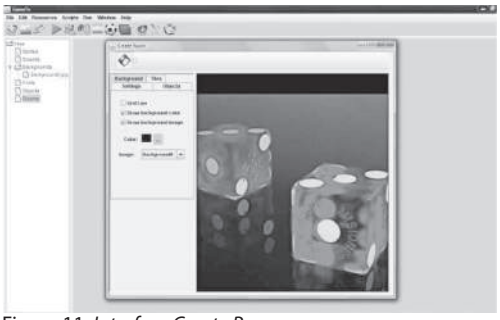


Figura 11: *Interface Create Room.*

À medida que vai modificando o cenário, por exemplo, uma configuração de tela, inserção de uma cor de *background* ou a inserção de um *object*, o painel ao lado vai recebendo as modificações simulando uma tela. Na Figura 11, observamos a aba *Background* com as opções de utilizar um *background* já existente ou uma cor de fundo desejada, conforme a escolha são marcadas as opções de "D*raw background color*" ou "*Draw background image*".

Do ponto de vista do jogo, os objetos são recursos que permitem os movimentos aos *sprites*, com a *interface Create Object* é possível escolher qual o *sprite* que se deseja adicionar os eventos e ações, como mostrado na Figura 12, ou seja, é possível programar o comportamento do objeto, tendo a percepção de animação e dinamismo. A diferença entre os objetos e os *sprites* é que estes últimos são apenas imagens animadas ou não, os objetos possuem comportamentos e são representados pelos *sprites*. Os jogos são formados pelos objetos, sendo que um objeto descreve uma entidade, e a partir dele são criadas as instâncias que executam ações do jogo.



Figura 12: *Interface Create Object.*

A ideia de adicionar objetos no jogo é para que haja interação, para isso é necessário indicar quais eventos a ser adicionado ao objeto e que ações devem ser tomadas após este evento. As opções de eventos serão apresentadas em uma nova janela, demonstrada na Figura 13, onde se tem os seguintes eventos:

• *Colision* - ocorre quando uma instância colide com uma outra instância de objeto.

• *Create* - acontece quando uma instância é criada.

• *Destroy* – sucede um pouco antes de uma instância ser destruída.

• *Key Press* – ao se pressionar uma tecla do controle remoto.



Figura 13: Opções de Eventos

As ações são executadas de acordo com os eventos. Por exemplo, o evento creation é emitido quando uma instância do objeto é criada. Quando o objeto for criado, é possível atribuir a ele a ação de mover-se. Quando dois objetos se chocam, acontece o evento *collision*, e neste caso é possível exe-

cutar a ação de parar ou mudar a direção do movimento. Além disso, é possível ainda executar a ação de "emitir um som" por ocasião de um evento *collision*. Outro exemplo de ação é quando o jogador pressiona uma tecla e o evento *key Pressed* pode executar a ação de o personagem mover-se em uma direção de acordo com a tecla pressionada. Para cada objeto é possível indicar ações para diversos eventos e, assim, definir o comportamento do objeto, sendo possível inserir as seguintes ações: movimento livre ou direcional; parar; pular; tocar som.

Uma vez definido o jogo e ações, um *framework* deve ser capaz de gerar o código-fonte para o jogo em ambiente de TV digital. Neste trabalho, o código é gerado utilizando as classes e métodos do *Framework GameTV*, o que torna o processo mais simples e reduz a quantidade de código fonte que deve ser gerado. A criação do código é feito de maneira automática em tempo de execução e é possível simular o ambiente de TV digital com o *XletView*.

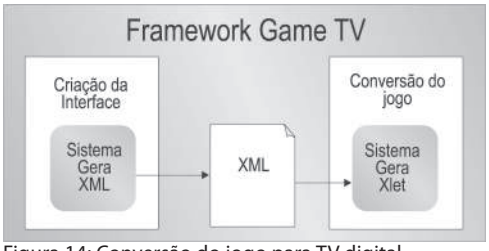


Figura 14: Conversão do jogo para TV digital.

A ideia é criar o jogo no computador através da *interface* gráfica, tendo uma usabilidade amigável ao usuário e o *GameTV* se encarregará de gerar o código para a TV digital. Para isso é necessário inserir os dados em arquivos XML, onde cada cenário possui um arquivo XML para que possa ser consultado e montado o jogo para o ambiente de TV digital, conforme Figura 14.

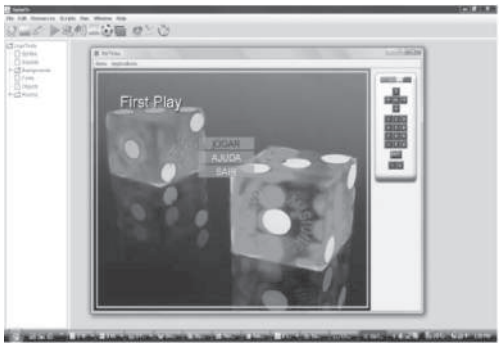


Figura 15: *Interface* criada com o *GameTV.*

Após o código fonte criado, o sistema compila o código utilizando o Java TV e suas bibliotecas, executando o jogo no ambiente da TV digital. Devido o Ginga-J ainda estar em fase final de desenvolvimento, o resultado do jogo criado é apresentado apenas no emulador, através do *Xletwiew*. Vide tela do jogo criado na Figura 15.

5. CONCLUSÕES

O *framework* GameTV proposto neste trabalho permite contribuir com o aumento da demanda da criação de jogos para TV digital no Brasil, facilitando e incentivando os desenvolvedores de *software* e provedores de conteúdo televisivo na criação de jogos para essa nova plataforma, com a disponibilização de rotinas funcionais necessárias nos jogos digitais, conforme apresentado.

Sem a intenção de ser uma solução definitiva, o *TVGame* ainda necessita de adaptações à tecnologia de TV Digital, que ainda se encontra em implantação, bem como padronizações que sociabilizem sua utilização.

6. REFERÊNCIAS

ALMEIDA, *M.* R. G. *Programação de Jogos 2D usando o MRDX e Fundamentos de Jogos 3D. 2004. 73f.* Monografia (Bacharelado em Ciência da Computação) – Centro de Ciências Exatas e Naturais, Universidade Federal do Pará, Belém. 2004.

ASSIS, *S.* P.; SUZANO, R. *Framework:* Conceitos e Aplicações. CienteFico,. Salvador, ano III, v.2, p. 1-21, Salvador, Jun./Dez. 2003.

BARBOSA *S.D.J.;* SOARES, L.F.G. *TV Digital interativa no Brasil se faz com Ginga:* Fundamentos, Padrões, Autoria Declarativa e Usabilidade. Em T. Kowaltowski & K.Breitman (orgs.) Atualizações em Informática 2008. Rio de Janeiro, Rj: Editora PUC-Rio, 2008. pp.105-174.

DTV. *Site* Oficial da TV Digital Brasileira. Disponível em <http://www.dtv.org.br/materias.a sp?menuid=3&id=11>. Acesso em 09/02/ 2010.

FERREIRA, *D.* A; SOUZA, C. T. *TuGA:* Um *Middleware* para o Suporte ao Desenvolvimento de Jogos em TV Digital Interativa. NASH – Núcleo Avançado em Engenharia de *Software* Distribuído e Sistemas Hipermídia. Centro Federal de Educação Tecnológica do Ceará. Infobrasil2008. Fortaleza, 2008.

LEMOS, *G.;* FERNANDES, J.; SILVEIRA, G. *Introdução à Televisão Digital Interativa:* Arquitetura, Protocolos, Padrões e Práticas. In: Jornada de Atualização em Informática do Congresso da Sociedade Brasileira de Comunicação, 2004, Salvador. Anais. Salvador: [s.e.], 2004.

MORRIS, *S.;* SMITH-CHAIGNEAU, A. *Interactive TV Standards. Burlington:* Ed. Focal. Press, 2005. 585 p. NBR15290.

PINTO, *L.* D. O, QUEIROZ-NETO, J. P., SILVA, K. N. *Tecnologias utilizadas para o desenvolvimento de aplicações educacionais através da TV Digital.* In CONNEPI, 2007.

ROCHA, J. T. S., 2003. *Forge 16V: Um Framework para Desenvolvimento de Jogos Isométricos.* 2003. 135f Dissertação (Mestrado em Ciências da Computação) - Centro de Informática. Universidade Federal de Pernambuco. Pernambuco, 2003.

SILVA, J. Q. *TV Digital Interativa.* 2004. 78f. Monografia (Especialização em Redes de Computadores) - Centro de Ciências Exatas e Tecnológicas, Universidade do Vale do Rio dos Sinos. São Leopoldo: UVRS, 2004.

SILVA, *R.* P. *Suporte ao desenvolvimento e uso de frameworks e componentes.* Tese (Doutorado em Ciência da Computação) – Departamento de Ciência da Computação, Universidade Federal do Rio Grande do Sul. Porto Alegre, 2000.

SOARES, *L.* F. G., 2007. *Ambiente para desenvolvimento de aplicações declarativas para a TV digital brasileira.* TV digital: qualidade e interatividade / IEL.NC.– Brasília: IEL/NC, 2007.

ZANCANARO, *A.;* SANTOS, P.; TODESCO, J. L., 2009. *Ginga-J ou Ginga-NCL: características das linguagens de desenvolvimento de recursos interativos para a TV Digital.* 1º Simpósio Internacional de Televisão Digital (SIMTVD), novembro, Bauru/SP. 2009.