UMA ESTAÇÃO DE TESTE DE APLICAÇÕES INTERATIVAS DE TV DIGITAL BASEADA EM SOFTWARE LIVRE

Lailson NOGUEIRA (1); Cidcley SOUZA (2)

(1) Centro Federal de Educação Tecnológica do Ceará (CEFET-CE), Rua: Monsenhor Vicente Martins nº 1022 Bairro: Henrique Jorge, (085)3233-1877, lailson.lima@gmail.com
(2) Centro Federal de Educação Tecnológica do Ceará (CEFET-CE), cidcley@gmail.com

RESUMO

Desde o dia 2 de dezembro de 2007, começaram as transmissões das primeiras imagens de TV Digital no Brasil. Esta tecnologia vem aliada à promessa de democratização do acesso à informação à população brasileira em geral. Entretanto, os reais benefícios da TV Digital só serão observados com o aumento da produção de conteúdo de alta definição e, principalmente, com o desenvolvimento de aplicações interativas. Entretanto uma das grandes dificuldades dos produtores de conteúdo são os custos associados ao teste de aplicações. De fato, o processo de transmissão e recepção de aplicações em TVDI é bastante complexo e requer um conjunto de equipamentos específicos. Assim, como mecanismo alternativo, propomos nesse artigo um ambiente de simulação da cadeia de transmissão e recepção de aplicações interativas de TVDI, denominado S-TVDI. Este ambiente, que é totalmente baseado em soluções de software livre, possibilita a realização de testes de aplicações interativas com características muito próximas aos ambientes reais de difusão de TVDI.

Palavras-chave: TV digital, testes de aplicações, simulação, software livre

1. INTRODUÇÃO

Desde o dia 2 de dezembro de 2007, começaram as transmissões das primeiras imagens de TV Digital no Brasil. Depois dessa data histórica, temos uma nova realidade na televisão brasileira. O telespectador, antes passivo na era da TV analógica, passará a poder interagir com os programas, além de assisti-los numa imagem bem mais nítida.

Com essa tecnologia serão disponibilizadas imagens com alta definição, som digital, interatividade, múltiplos programas simultaneamente, venda de produtos via televisão, jogos, entre outras funcionalidades. Nesse sentido, a disseminação da televisão aberta digital é de importância estratégica para o Brasil, uma vez que beneficiará quase toda a população nos próximos anos, proporcionando crescente democratização do acesso à informação.

Entretanto, os reais benefícios da TV Digital só serão observados com o aumento da produção de conteúdo de alta definição e com o desenvolvimento de aplicações de interatividade. O Brasil, porém, precisa vencer diversos problemas para que possa se beneficiar dessa novidade. Esses problemas podem ser resumidos em três tópicos principais:

- Falta de profissionais capacitados no Brasil para dar suporte à produção de software para TV Digital;
- Escassez de métodos e técnicas que possam sistematizar a produção de software para esse novo mercado;
- Elevados custos das ferramentas de desenvolvimento e teste de aplicações em TV Digital.

O tratamento urgente de todos esses pontos é fundamental para que o Brasil possa se firmar, em um futuro muito próximo, como uma nação independente nesse mercado tão promissor. Todavia, antes de se pensar em capacitar profissionais ou produzir técnicas novas para a construção de softwares interativos, é necessária a implementação de uma infra-estrutura para que as aplicações possam ser desenvolvidas e testadas. Portanto esse é um problema básico que necessita ser tratado urgentemente.

A opção de se utilizar software para simular as etapas de transmissão e recepção é realista e necessária, uma vez que equipamentos reais que façam esse processo são muito caros e proibitivos à maioria das empresas brasileiras. Contudo os softwares disponíveis atualmente também têm um custo bastante elevado e não contemplam a cadeia completa de transmissão e recepção, sendo sempre necessária a aquisição de placas moduladoras, por exemplo, as quais possuem custos elevados.

A grande desvantagem da utilização de apenas parte do processo de recepção, a qual é uma técnica adotada por simuladores como XleTView (XLETVIEW, 2004), Espial (ESPIAL GROUP, 2006) ou OpenMHP (OPENMHPPROJECT, 2004), é que isso não é suficiente para garantir o funcionamento de uma aplicação, visto que, ao utilizarmos esses simuladores parciais, apenas vemos a nossa aplicação funcionando em um programa que simula a TVDI. Porém, no mundo real, essa aplicação passará por processos como a multiplexação, a modulação e a demultiplexação. Só depois dessas etapas, ela chega aos televisores domésticos. Uma vez que não simulamos todas as etapas de transmissão e recepção, não podemos ter certeza se a aplicação realmente funcionará na casa dos telespectadores se somente usarmos simuladores parciais.

Outro grande problema é que esses programas oferecem um ambiente menos restritivo que os set-top-box (STB) reais, ou seja, o desenvolvedor de aplicações pode programar a aplicação utilizando técnicas que estejam dependentes desses softwares inviabilizando assim seu funcionamento nos STBs.

Devemos ter em mente também que os custos das ferramentas que simulam as etapas de transmissão e recepção de TVDI também são significativos. Assim, para se atingir a meta de inclusão social do SBTVD, os custos da produção de software devem ter um impacto controlado nos produtos finais.

Nesse sentido, apresentamos nesse trabalho o ambiente de simulação S-TVDI. Esse ambiente trata de todas as etapas de transmissão e recepção de conteúdos utilizando apenas software livre. A intenção desse trabalho é fornecer um mecanismo eficiente e barato para permitir que desenvolvedores de software brasileiros possam de fato entrar nesse novo mercado de desenvolvimento que deve ser formar muito em breve por conta dessa nova tecnologia que está sendo implementada no Brasil. Visto que esse ambiente de simulação é baseado nas especificações técnicas fornecidas pelas normas da ABNT para a TV digital brasileira (ABNT, 2006);

Para apresentar esse ambiente, esse trabalho está dividido da seguinte forma: na seção 2 descrevemos todas as etapas relacionadas à transmissão e a recepção de aplicações em TVDI, mostrando em detalhes cada um dos mecanismos necessários para realizar a tarefa de difusão de sinal digital assim como os detalhes do padrão MPEG-2. Na seção 3 descrevemos alguns aspectos sobre trabalhos relacionados ao tema. Na seção 4, apresentamos a arquitetura do ambiente de simulação de transmissão e recepção que estamos propondo, assim como a forma que o implementamos. Na seção 5 mostramos os passos necessários para simular aplicações interativas. Finalizamos, na seção 6, com algumas conclusões e trabalhos futuros.

2. TRANSMISSÃO E RECEPÇÃO DO SINAL DA TV DIGITAL INTERATIVA

2.1 Arquitetura

A implantação de um ambiente de testes de aplicações de TVDI está relacionado ao entendimento do processo de difusão e recepção das aplicações. Inicialmente o sinal a ser difundido deve ser construído. Essa construção é dividida em uma série de etapas. A Figura 3 ilustra um modelo simplificado desse processo. A primeira etapa é a codificação e compressão da informação seguindo um padrão digital bem definido (FERNANDES, 2004). A saída de cada codificador é um fluxo de dados denominado fluxo elementar. Quando vários fluxos elementares são relacionados entre si, como seqüências de vídeo com seqüências de áudio e possivelmente também com seqüências de dados, eles formam um serviço.

Na TVDI, todos os serviços são multiplexados em uma seqüência de dados denominada de fluxo de transporte. Essa tarefa é geralmente implementada por um sistema encarregado de multiplexar, além dos fluxos de vídeo, áudio e dados, tabelas que descrevem apropriadamente o conjunto de serviços transportados em um fluxo de transporte. Essas tabelas são conhecidas como tabelas SI (*Service Information*). Na próxima etapa, o fluxo de transporte é então modulado em uma onda portadora e difundido via satélite, cabo ou radiodifusão.

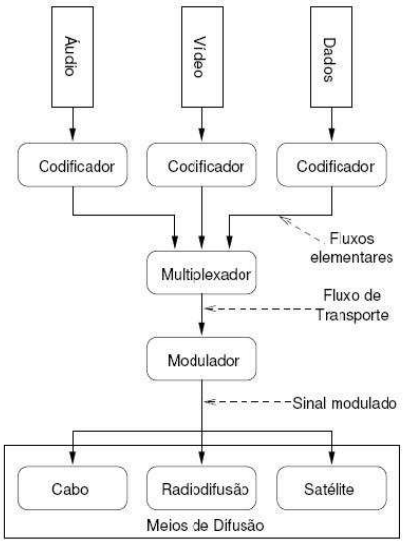

Figura 1 - Transmissão em TVDI

Já na fase de recepção, o equipamento de recepção pode ser integrado a uma televisão, a televisão digital, ou ser um equipamento à parte. Esse é conhecido industrialmente como set-top-box (STB). A idéia básica desse dispositivo, quando não embutido em uma televisão digital, é o de uma pequena caixa com a função de processar e converter os sinais digitais para o formato analógico utilizado pelas televisões atuais. As principais etapas de processamento do sinal em um terminal de acesso são ilustradas na Figura 4.

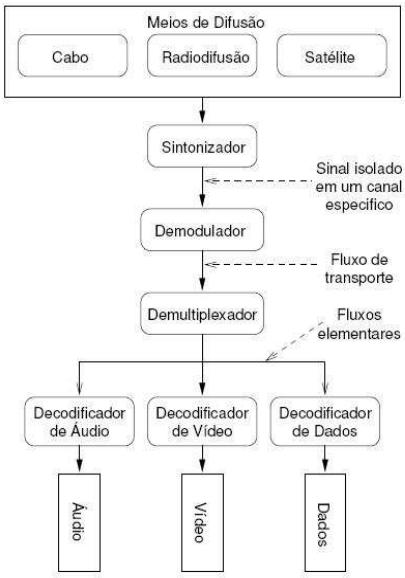

Figura 2 - Recepção em TVDI

Antes de ser processado pelo STB, o sinal difundido é captado por uma antena específica para a tecnologia, no caso da difusão via satélite ou radiodifusão, e isolado em determinado canal pelo sintonizador. O sinal passa então pelo demodulador, que restaura o fluxo de transporte difundido. O demultiplexador é encarregado de extrair os fluxos elementares de determinado serviço, alimentando os respectivos decodificadores. Geralmente existe um decodificador para áudio e um para vídeo, e o correto tratamento dos outros dados depende da implementação do sistema. O áudio decodificado é encaminhado para a saída de áudio do terminal de acesso, assim como o vídeo é direcionado para a tela de exibição. Outros tipos de dados são processados, executados e exibidos se necessários. Dependendo da implementação, o usuário pode interagir com alguns serviços.

Como pudemos observar, os processos de transmissão e recepção do sinal de televisão digital abrangem muitos elementos complexos. Assim, se quisermos testar nossa aplicação em um ambiente real, teríamos que gastar alguns milhares de dólares para a compra dos equipamentos que fazem esses processos. Torna-se evidente, então, que a simulação dessas etapas via software é bem mais viável economicamente.

2.1 O Padrão MPEG-2

Para melhor entender a cadeia de transmissão e recepção de TVDI é necessário também, conhecer um pouco do padrão MPEG-2, visto que os padrões de TVDI atualmente usam fluxos (streams) MPEG-2 para a transmissão do sinal de TVDI. Cada fluxo, denominado Fluxo de Transporte (*Transport Stream*), ou simplesmente TS, é transmitido através de um canal e tem tipicamente uma taxa de dados em torno de 40 Mbit/s (para satélite e cabo, e de 25 Mbit/s para a transmissão terrestre). Esta taxa da transmissão é suficiente para fornecer de sete a oito canais de TV separados. Um canal é geralmente uma faixa de freqüência, que é transmitida usando satélite, cabo ou transmissão terrestre.

Um TS consiste em um conjunto de sub-fluxos chamados Fluxos Elementares (*elementary streams*). Cada fluxo elementar pode conter áudio MPEG-2 codificado, vídeo MPEG-2 codificado ou dados encapsulados. Um TS é formado por pacotes que carregam esses fluxos elementares multiplexados no tempo. Para identificar que parte do fluxo de transporte pertence a que fluxo elementar, cada pacote contém um identificador do pacote (PID) de seu fluxo elementar. A Figura 1 mostra uma visão esquemática de um fluxo de transporte.

Um STB que recebe estes fluxos necessita de informações suplementares para decodificá-los da maneira correta, de modo a responder questões como: que fluxo contém que tipo de dados e que fluxos devem permanecer juntos. Esses dados são fornecidos pelas informações de serviço do fluxo.

A tabela da associação de programas (*Program Association Table* - PAT) é a tabela fundamental para as informações de serviço. Um fluxo de transporte contém somente uma PAT, que contém os PIDs da PMT (*Program Map Table*) de cada serviço. A PAT é transmitida com PID fixo 0.

O conjunto de alguns fluxos elementares é chamado de serviço (TEKTRONIX, 2002). Um serviço pode ser um canal de TV, um canal de rádio, ou um canal de dados. A PMT descreve de quais fluxos um serviço consiste e de que tipo eles são. Há somente uma PMT por serviço em um fluxo de transporte. A relação entre os fluxos elementares, entre a PAT e as PMTs pode ser visualizada no exemplo da Figura 2. O exemplo mostra uma parte do fluxo de transporte e como as tabelas de informação de serviço são usadas para a decodificação do fluxo.

As setas na Figura 2 indicam os dados de um canal de TVDI específico. Primeiramente, o STB recebe a PAT, localizada no PID 0. A PAT aponta para o PID 200, onde a PMT do serviço 1 é encontrado. O receptor procura os tipos e PIDs dos fluxos elementares que formam o serviço. O PID 100 carrega o vídeo, o PID 102 o áudio, e o PID 105 uma aplicação de TVDI. O receptor pode agora montar os diferentes fluxos elementares e decodificar o vídeo, o áudio e os dados.

As informações de serviço dos fluxos MPEG incluem mais tabelas do que a PAT e a PMT, contudo essas tabelas estão relacionadas ao sistema de TVDI específico.

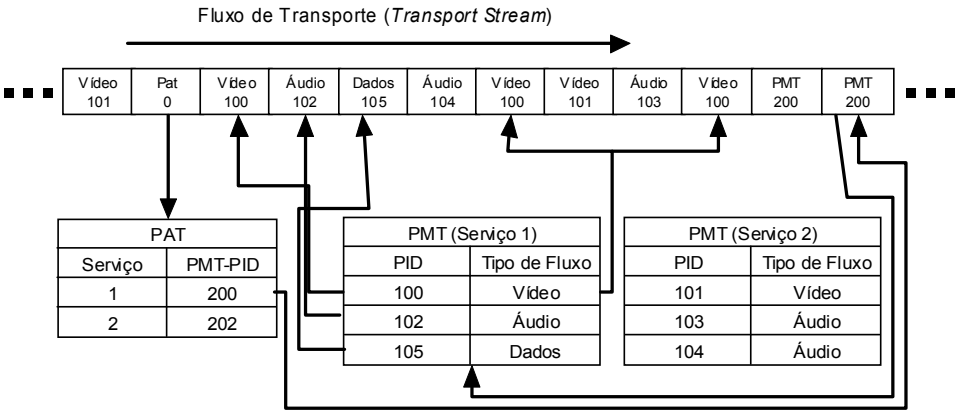

Figura 3 - Tabelas PAT e PMT

3. TRABALHOS RELACIONADOS

Existem diferentes tipos de implementação para um ambiente de simulação de aplicações para TV digital. Cada qual tem sua vantagem/desvantagem, fazendo com que esteja nas mãos do cliente a decisão de qual tipo de implementação usar (PYTELKA, 2007).

O ambiente de simulação mais próximo da realidade só é possível de ser implementado se usarmos um hardware para fazer a modulação do fluxo de transporte. Esse tipo de hardware conhecido como placa moduladora é extremamente caro.

A modulação é necessária, pois a maioria dos sinais, da forma como são fornecidos pelo transdutor, não podem ser enviados diretamente através dos canais de transmissão. Conseqüentemente, uma onda portadora cujas propriedades são mais convenientes aos meios de transmissão, é modificada para representar a mensagem a ser enviada. A modulação é a alteração sistemática de uma onda portadora de acordo com a mensagem (sinal modulante), e pode incluir também uma codificação (CAMPOS, 2002).

Percebemos então que um ambiente de simulação com modulação é inviável para a maioria dos provedores de conteúdo para TVDI brasileiros. A idéia desse artigo é justamente fornecer um software de simulação próximo ao real, flexível para futuras atualizações e que seja principalmente de baixo custo.

Por fim, temos também a óbvia possibilidade de testar nossas aplicações diretamente no Set-Top-Box, porém perderíamos muitos detalhes da transmissão do sinal como, por exemplo, a sincronização do conteúdo e a geração do carrossel de objetos. E também poucos STBs possuem suporte a esse tipo de estratégia.

4. O AMBIENTE S-TVDI

4.1 Arquitetura

Na Figura 5, apresentamos a estrutura proposta para o nosso ambiente de testes integrado, a qual é composta pelos seguintes componentes:

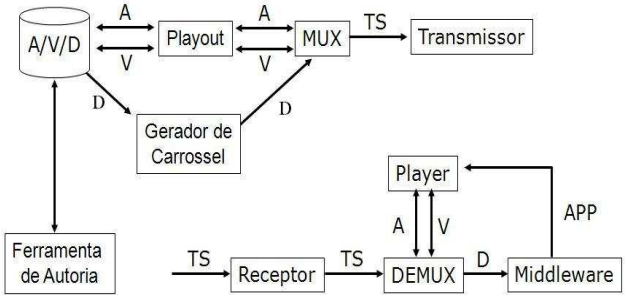

Figura 4 – Arquitetura do S-TVDI

- **Ferramenta de Autoria**: Obviamente, antes de iniciar qualquer teste, a aplicação deverá primeiro ser produzida. Futuramente será integrada à estação de testes uma ferramenta de autoria que também está sendo desenvolvida;
- **A/V/D**: Refere-se a áudio/vídeo/dados. Depois da aplicação estar pronta, obtemos o áudio/vídeo e possivelmente também dados dela. Áudio e vídeo são enviados ao Playout, e, se houver dados, eles serão enviados ao Gerador de Carrossel;
- **Playout**: O Playout tem a função de entregar o áudio e o vídeo (nessa etapa, chamados de fluxos elementares) ao multiplexador;
- **Gerador de Carrossel**: Será utilizado se a aplicação possuir dados para serem transmitidos. Essa etapa do processo é a responsável por criar um carrossel de objetos necessários para a transmissão dos dados da aplicação. O carrossel de objetos tem a função de possibilitar a transmissão de dados, de forma periódica, aos STBs. A idéia fundamental desse protocolo é de módulos de dados transmitidos de forma cíclica, de modo que, quando o receptor precisar de determinado módulo, deve apenas esperar o momento de sua próxima repetição no fluxo de dados;
- **Mux**: Referente ao multiplexador. Aqui todos os fluxos elementares são multiplexados para a geração do fluxo de transporte (*Transport Stream*);
- **Transmissor/Receptor**: Para efeito de testes, a transmissão e recepção do fluxo de transporte dar-se-á de um computador para outro via rede, pois torna-se inviável a aquisição de placas moduladores e antenas para realizar essa etapa;
- **DEMUX**: Referente à tarefa de demultiplexação, ou seja, obtenção dos fluxos elementares a partir do fluxo de transporte.
- **Player**: Caso a aplicação só possuir áudio e vídeo, eles serão finalmente entregues para um player de vídeo que tocará esse vídeo;
- **Middleware**: É através do middleware que os dados serão tratados independente da plataforma de hardware dos fabricantes de STBs. Caso a aplicação possua dados, eles deverão ser entregues primeiro ao Middleware para que ele possa finalmente entregar ao player de vídeo e encerrar assim a simulação da transmissão/recepção.

Após a integração de todas essas etapas, temos formado o nosso ambiente de testes. Podemos perceber que a união de todas essas etapas contempla toda a cadeia de transmissão e recepção do sinal de TVDI, logo a nossa abordagem está bem coerente com o objetivo proposto pelo artigo.

4.2 Implementação

Para iniciar a construção deste ambiente de testes, inicialmente foram levantadas as ferramentas de software disponíveis atualmente que sejam capazes de atuar em algumas das etapas do processo de transmissão e recepção de TVDI. Esses softwares foram encontrados na Web, sendo que foram priorizadas soluções de código aberto, que pudessem ser adaptadas para a produção da estação de teste de aplicações.

Listamos algumas dessas ferramentas abaixo, sendo que todas são de código livre e compatíveis com o sistema operacional Linux.

- FFmpeg (FFMPEG, 2008): Conversor de áudio/vídeo. Útil para podermos converter os fluxos elementares para os mais diversos formatos, inclusive os usados pela TV digital, como o MPEG-2.
- Mplex13818 (LINUXTV PROJECT, 2003): Programa multiplexador de fluxos elementares. Recebe fluxos elementares como entrada e devolve um fluxo de transporte como saída.
- VLC media player (VIDEOLAN, 2008): Tocador multimídia de código aberto e poderoso. Ele pode ler diversos formatos de áudio e vídeo, assim como ler vários protocolos de transmissão (streaming). Também tem a opção de ser usado como servidor para vídeos.
- TS2PES (LINUXTV PROJECT, 2003): Ferramenta demultiplexadora. Útil para simularmos uma parte do processo de recepção do sinal da TV digital em que o fluxo de transporte entra nesse programa e ele devolve os fluxos elementares.
- XletView: Simula uma aplicação (xlet) funcionando em uma televisão, porém não simula todo o processo de transmissão e recepção do sinal da TVDI.
- Dsmcc-mhp-tools (LINUXTV PROJECT, 2004): Gerador de carrossel. Serve para simular a transmissão dos dados da TV digital.

Foi decidido que o ambiente precisa ser feito utilizando a linguagem de programação Java, uma vez que o Java é uma das poucas linguagens com APIs (Application Programming Interface) especialmente criadas para o trabalho com a TV Digital. A Java TV é um bom exemplo de API que foi feita com o intuito de ajudar os desenvolvedores de aplicações a produzir aplicativos para a TVDI.

Uma das grandes dificuldades sentidas na realização deste trabalho se deve ao fato de que grande parte dessas ferramentas foram desenvolvidas utilizando as linguagens de programação C/C++, e o nosso ambiente está sendo feito em Java. Reescrever esses programas para a linguagem Java não é uma alternativa simples nem rápida, uma vez que o código fonte de cada uma dessas ferramentas é enorme e bastante complexo.

Inicialmente foi pensado em utilizar JNI (Java Native Interface) para poder fazer essa integração, pois a JNI foi criada justamente com esse objetivo de fazer a integração entre o Java e outras linguagens como o C++, por exemplo. Entretanto, após a realização de alguns protótipos, a JNI mostrou-se de difícil implementação e pouco funcional perante os objetivos pretendidos.

A solução que está sendo adotada e que parece ser a mais simples para realizar essa integração foi a de utilizar sockets e shell scripts. Sockets são pontos-finais (*endpoints*) para a comunicação ponto-a-ponto entre sistemas. Eles escondem do programador alguns detalhes da rede que são irrelevantes para ele. Os sockets são implementados em três formas: Stream (Interface para TCP – *Transmission Control Protocol*), Datagram (Interface para UDP – *User Datagram Protocol*), Raw (Interface para ICMP – *Internet Control Message Protocol*).

Dessas três formas de implementação, a escolhida para ser utilizada no ambiente foi a stream, pois ela utiliza o protocolo TCP que é orientado à conexão e possui um transporte de dados confiável, controle de fluxo e de seqüênciamento de pacotes. Um ônus dessa implementação é a velocidade mais baixa de transferência de dados em relação aos sockets que usam o protocolo UDP.

Por sua vez, um shell script é um script escrito para o shell, ou interpretador da linha de comando, de um sistema operacional, no nosso caso Linux. Operações típicas realizadas por shell scripts incluem manipulação de arquivos, execução de programas e escrita de texto.

Com estes conceitos definidos, o primeiro passo para a construção do S-TVDI foi realizar a integração entre o multiplexador (escrito em C) e o nosso programa principal em Java. Para isso procedemos da seguinte

forma: o programa principal recebe os fluxos elementares de entrada inseridos pelo usuário e então envia uma requisição para que o multiplexador gere o fluxo de transporte a partir dos fluxos elementares inseridos.

O programa principal então executa um shell script que recebe como argumento uma string contendo todos os fluxos elementares. O shell script está escrito de tal forma que apenas pega essa string e executa o multiplexador com os fluxos elementares desejados passados como argumentos.

Contudo, antes de iniciar qualquer implementação, precisávamos conseguir fluxos elementares para fazer os testes necessários. Uma boa alternativa foi a de utilizar um programa demultiplexador em alguns vídeos para obtê-los. O programa TS2PES realiza essa tarefa facilmente. Construímos, assim, um pequeno banco de fluxos elementares para testes. Esse processo foi necessário, uma vez que a aquisição de câmeras de alta definição para a obtenção desses fluxos implicaria um custo extremamente elevado.

De posse do fluxo de transporte, ele deverá ser enviado para outro computador usando o protocolo de transmissão UDP, pois quando trabalhamos com a transmissão contínua de vídeos o protocolo UDP é mais eficiente que o TCP, uma vez que ele transmite os pacotes bem mais rapidamente. A única ressalva fica por conta de que alguns pacotes podem se perder durante a transmissão, pois ao contrário do protocolo TCP, não temos garantia de entrega de todos os pacotes. Após o término dessas etapas, consideramos como concluída a transmissão do sinal da TV digital que foi proposta na nossa abordagem.

Para fazer a recepção do fluxo de transporte em outro computador podemos usar algum player de vídeo com a capacidade de recepção de vídeos pela rede. O programa VLC media player tem exatamente esta característica. Basta deixá-lo na rede "escutando" por novas conexões e ele receberá e "tocará" o vídeo. Isso representa finalmente a recepção do sinal de TV digital. Uma alternativa a isso é integrar um player próprio à nossa interface gráfica, abordagem essa que será explicada na próxima subseção.

4.3 Interface Gráfica

Para que a estação de testes seja utilizável pelo usuário, é importante que ela possua uma interface gráfica simples e intuitiva. Segundo essa idéia é apresentada, na Figura 5, a versão atual do STVD-I, desenvolvida em Java e utilizando os componentes da classe Swing.

Figura 5 – Ambiente S-TVDI: Transmissor

A nossa interface é constituída de dois ambientes: o do transmissor e o do receptor. Quando o usuário clica na opção "Arquivo" ele pode escolher qual ambiente quer usar, tornando assim desnecessária a instalação de outro software.

Para o correto funcionamento do software, precisamos deixar o ambiente receptor em um computador escutando por conexões e depois em outro computador executamos o ambiente transmissor e nele inserimos os fluxos elementares desejados, bem como o IP do qual o ambiente receptor está esperando e a porta utilizada para a conexão.

Depois de configurarmos tudo, o usuário clica em "Iniciar simulação" e então o programa começa a mostrar o log de operações na área de texto da interface. Após a criação do fluxo de transporte o programa envia o vídeo para o receptor e também o toca localmente.

Para que o vídeo seja "tocado" dentro da nossa interface utilizamos a API JMF (*Java Media Framework*) que permite que áudio, vídeo e demais mídias baseadas no tempo sejam adicionadas em aplicações Java.

5. SIMULAÇÃO DE APLICAÇÕES INTERATIVAS

A abordagem acima é eficiente para simular aplicações que contenham áudio e vídeo, porém quando falamos em aplicações interativas, outra abordagem deve ser adotada, pois a inclusão de dados na aplicação torna o processo bem mais complexo.

Inicialmente, ao colocarmos dados na simulação, temos que obrigatoriamente enviá-los ao programa dsmcc-mhp-tools, para que ele crie o carrossel de objetos. Como mencionado anteriormente, o carrossel de objetos tem a função de possibilitar a transmissão de dados, de forma periódica, aos STBs.

Juntamos então o áudio, vídeo e o carrossel de objetos no programa multiplexador para criar o fluxo de transporte que agora possui também o carrossel de objetos dentro dele. De forma análoga às aplicações não-interativas, o sinal é então transmitido para outro computador.

Porém, torna-se clara então a necessidade de que áudio, vídeo e dados sejam demultiplexados. Novamente chamamos o programa TS2PES para que seja feita a obtenção dos fluxos elementares a partir do fluxo de transporte recebido.

Após a demultiplexação, os fluxos de áudio e vídeo serão entregues ao player, os dados porém deverão ser entregues ao middleware para que ele possa fazer o tratamento adequado deles e só então entregá-los ao player.

Notamos agora a importante relação entre o middleware brasileiro Ginga e a interatividade na TV Digital. Não há interatividade sem um middleware, pois ele é o responsável por tratar dos dados como legendas, tabelas, menus e tudo mais que possa trazer interatividade para o telespectador. Então, fica latente a necessidade de que o Ginga e seu código fonte seja disponibilizado publica e rapidamente para que a comunidade brasileira possa trabalhar construindo aplicações ou no nosso caso, construindo ferramentas para o teste de aplicações.

Além de ainda não podermos contar com o Ginga, outro grande problema para a simulação de aplicações interativas se dá pelo fato de ainda não termos um player de vídeo próprio para esse tipo de aplicação. A construção de um player que satisfaça as nossas necessidades está em desenvolvimento.

Enquanto o player não fica pronto, a saída provisória pensada foi de, enviarmos o áudio e o vídeo para alguma aplicação de simulação parcial do sinal de TVDI. Um exemplo seria o envio desses fluxos ao programa XletView que simula aplicações sendo executadas em uma televisão. Nesse caso, o vídeo e o áudio dessa aplicação seriam os fluxos elementares recebidos.

Na nossa implementação, ainda não está sendo considerada nenhuma aplicação com interatividade, uma vez que nossa prioridade inicial é testar aplicações com apenas áudio e vídeo. Porém já está sendo estudado como um programa gerador de carrossel poderá ser integrado ao S-TVDI.

6. CONCLUSÕES

Pudemos perceber que o desenvolvimento de um ambiente de testes de aplicações de TV digital interativa com código livre e totalmente funcional é algo que deve ser feito com urgência se o Brasil deseja realmente se sobressair nessa área perante outras nações.

Os benefícios com a conclusão desse trabalho serão imensos tendo em vista o imediato surgimento de inúmeros desenvolvedores de aplicações nessa área e a falta de softwares disponíveis na Internet que façam esse processo de simulação.

Este ambiente que estamos desenvolvendo ainda está incompleto, pois como a cadeia de transmissão e recepção é bastante complexa, muitos detalhes precisaram ser estudados a fundo. O lançamento do Ginga assim como o desenvolvimento de um player de aplicações interativas próprio para o S-TVDI são importantíssimos para que a nossa Estação seja totalmente concluída.

Porém já temos resultados que estão mostrando que as expectativas quando à funcionalidade dessas ferramentas trabalharem de forma integrada estão realmente sendo confirmadas.

O S-TVDI estará disponível para download assim que uma versão estável estiver pronta. A comunidade interessada poderá fazer o download dessa ferramenta e instalar no seu computador, além de contribuir incrementando funcionalidades do ambiente.

REFERENCIAS

ABNT. **A abnt e as normas para TV Digital**. Disponível em: <http://www.abnt.org.br/m3.asp?cod_pagina=1249>. Acesso em: Agosto de 2008.

CAMPOS, S. **O que é modulação e que modos são utilizados**. Disponível em <http://paginas.terra.com.br/arte/sarmentocampos/Modulacao.htm>. Acesso em Julho de 2008.

ESPIAL GROUP. **Espial**. Disponível em: <http://www.espial.com>. Acesso em: Agosto de 2008.

FERNANDES, J., LEMOS, G. E SILVEIRA, G. **Introdução à televisão digital interativa**: arquitetura, protocolos, padrões e práticas. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 24., 2004.

FFMPEG. Disponível em: <http://ffmpeg.mplayerhq.hu/>. Acesso em: Agosto de 2008.

LINUXTV PROJECT. **Dsmcc-mhp-tools**. Disponível em: <http://linuxtv. org/cgi-bin/viewcvs.cgi/dsmcc-mhp-tools/>. Acesso em: Agosto de 2008.

LINUXTV PROJECT. **Mplex13818**. Disponível em: < http://www.scara.com/~schirmer/o/mplex13818/ >. Acesso em: Agosto de 2008.

LINUXTV PROJECT. **Ts2pes**. Disponível em: < http://www.scara.com/~schirmer/o/mplex13818/>. Acesso em: Agosto de 2008.

OPEN MHP PROJECT. **Openmhp.** Disponível em: <http://www.openmhp.org>. Acesso em: Fevereiro de 2008.

PYTELKA, T. **Interaction in digital television environment**. Disponível em: <https://dip.felk.cvut.cz/browse/pdfcache/pytelt1_2007bach.pdf> . Acesso em Agosto de 2008.

TEKTRONIX. **A guide to mpeg fundamentals and protocol analysis, 2002**. Disponível em: <http://www.tek.com/Measurement/programs/mpeg_fundamentals/>. Acesso em: Agosto de 2008.
VIDEOLAN. **Vlc media player**. Disponível em: <http://www. videolan.org/vlc/>. Acesso em: Agosto de 2008.

XLETVIEW. Disponível em: <http://xletview.sourceforge.net/>. Acesso em: Agosto de 2008.