FCOM: um *framework* de comunicação para o desenvolvimento de aplicações de TV digital interativa

José Daniel Pereira Ribeiro Filho, Rafael Fernandes Lopes e Omar Andrés Carmona Cortes

***Resumo*— Com a criação do SBTVD (Sistema Brasileiro de Televisão Digital), um novo modelo tecnológico de televisão aberta e gratuita emerge no País. Além de melhorias significativas na definição de imagem e som das transmissões televisivas, o SBTVD propôs ainda a disponibilização de serviços baseados na interatividade com o telespectador, criando novas oportunidades de negócios e abrindo novas fronteiras para a inclusão social. No SBTVD os serviços de interatividade executam sobre o *middleware* Ginga.**

Este trabalho propõe o desenvolvimento de um *framework* de comunicação de alto nível que permita a implementação de aplicações para TV digital interativa de uma maneira facilitada. Estas aplicações poderão executar sobre o *middleware* Ginga. Dessa forma, o *framework* proposto permitirá às aplicações fazer uso do canal de interatividade para realizar a comunicação das mesmas com servidores remotos, provendo os mais variados tipos de serviços.

***Palavras-Chave*— *Framework*, Ginga, NCL, LUA, TV Digital.**

***Abstract*— With the creation of the SBTVD (Digital Television Brazilian System), a new technological model for free and open brazilian television arises. Beyond significant improvements in image and sound definition of the television transmissions, the SBTVD proposes the deployment of services based on interaction with the televiewer, creating new business opportunities and opening new frontiers for social inclusion. In SBTVD, the interactive services run over Ginga middleware.**

This work proposes the development of a high level communication framework which allows the implementation of applications for interactive digital TV in an easier way. These applications will run over Ginga middleware. Thus, the proposed framework allows applications to make use of the interactive channel for communicating with remote servers, providing several types of services.

***Keywords*— Framework, Ginga, NCL, LUA, Digital TV.**

I. INTRODUÇÃO

Com a criação do SBTVD (Sistema Brasileiro de Televisão Digital), um novo modelo tecnológico de televisão aberta e gratuita emerge no País. Além de melhorias significativas na definição de imagem e som das transmissões televisivas [1], o SBTVD propôs ainda a disponibilização de serviços baseados na interatividade com o telespectador, criando novas oportunidades de negócios e abrindo novas fronteiras para a inclusão social.

José Daniel Pereira Ribeiro Filho, Rafael Fernandes Lopes e Omar Andrés Carmona Cortes, Departamento Acadêmico de Informática, Instituto Federal do Maranhão, São Luís, Brasil, E-mails: jdanielprf@gmail.com, rafaelf@ifma.edu.br, omar.carmona@gmail.com. Este trabalho foi financiado pelo CNPq através do programa PIBIC/CNPq/IFMA.

A TV digital interativa (ou *Interactive Digital Television* - IDTV) expande o modelo tradicional de transmissão televisiva ao adicionar o canal de interatividade [2]. É este canal que permite ao telespectador realizar o envio e recebimento de dados de e para a emissora de televisão ou mesmo para servidores remotos disponíveis na Internet.

Através da TVDI, o telespectador abandona seu papel passivo e passa poder interagir com os programas televisivos e elaborar sua própria programação [3]. A possibilidade de executar aplicações computacionais cria um novo leque de oportunidades, permitindo que o usuário interaja com as diversas mídias apresentadas através da TV. Programas interativos, comércio eletrônico, seleção de programas exclusivos, acesso à navegação web são alguns dos novos serviços possíveis a partir do da implantação do SBTVD [4].

O canal de interatividade é o mecanismo que permite o transporte do tráfego de dados entre o aplicativo que executa na TV digital (ou *set-top box*) e os servidores remotos. Este canal poderá utilizar diversas tecnologias de transmissão de dados, como a telefonia celular, telefonia fixa, ADSL (*Assymetrical Digital Subscriber Line*), tecnologias de transmissão via rádio (e.g. Wi-Fi, WiMax), PLC (*Power Line Communications*), entre outras.

Do ponto de vista das aplicações, a interatividade da TV digital pode ser classificada em três categorias (de acordo com o grau de interação do usuário): (a) "local", (b) "intermitente" e (c) "permanente" [2] [3].

A interatividade "local" é considerada a mais básica das categorias. Nesse tipo de interatividade não há a necessidade de se utilizar o canal de interatividade, uma vez que toda a interação ocorre entre o usuário e a aplicação que executa localmente no *set-top box*. Em geral, essa classe de aplicações é baixada para o *set-top box* por meio da própria transmissão *broadcast* (carrossel de dados), via o protocolo DSM-CC (*Digital storage media command and control*) [5] [6] Como exemplo de aplicação desta categoria pode-se citar os EPGs (*Electronic Program Guides*) ou guias de programação da TV.

Na interatividade "intermitente" o aplicativo envia informações a um servidor remoto utilizando o canal de interatividade. No entanto, este servidor não envia respostas de volta ao telespectador. Por conta desta característica, diz-se que na interatividade "intermitente" o canal de retorno é considerado não-dedicado [3]. Por ser unidirecional, este tipo de comunicação pode ser utilizada em aplicações como votações e pesquisas de opinião.

Por fim, a interatividade "permanente" atua como uma

evolução da interatividade "intermitente". Nesta categoria, o canal de retorno deixa de ser unidirecional e passa a ser bidirecional, ou seja, passa a ser um canal dedicado. Assim, é possível utilizar-se de diversos serviços, como a navegação na Internet, serviços de compras, e-mail, chat, home-banking, educação a distância, entre outros.

A infraestrutura tecnológica do SBTVD foi baseada no padrão de japonês ISDB-T [7] (*Integrated Services Digital Broadcasting*) e no *middleware* Ginga [8], [9]. Esta infraestrutura deu origem a um novo padrão de televisão digital terrestre, chamado ISDTV-T (*International Standard for Digital Television Terrestrial*).

As aplicações desenvolvidas para o *middleware* Ginga já trazem suporte à interatividade local. No entanto, para desenvolver mecanismos avançados de interatividade intermitente e permanente, o desenvolvedor necessita de conhecimentos sobre protocolos de aplicação e programação em rede.

Dessa forma, com vistas a facilitar o processo de desenvolvimento desse tipo de aplicações, torna-se necessária a existência de mecanismos de comunicação que possam ser facilmente acoplados às aplicações (clientes e servidores), simplificando a implementação das funcionalidades de comunicação remota.

Nesse contexto, este artigo apresenta um *framework* de comunicação de alto nível chamado FCOM, que permite o desenvolvimento de aplicações para TV digital de uma maneira mais fácil e rápida. Este *framework* fará uso de um canal de interatividade para realizar a comunicação da aplicação com um servidor remoto, fornecendo recursos de interatividade intermitente e permanente.

Este artigo está organizado como a seguir: a Seção II apresenta o *middleware* Ginga; já a Seção III traz informações sobre a arquitetura do *framework* FCOM; a Seção IV apresenta a implementação do *framework* FCOM, bem como são apresentadas algumas aplicações construídas com o uso deste; por fim, na Seção V são apresentadas as conclusões deste trabalho e as perspectivas futuras.

II. O MIDDLEWARE GINGA

O Ginga é uma camada de software intermediário (*middleware*) que permite o desenvolvimento de aplicações interativas para a TV Digital de forma independente da plataforma de hardware dos fabricantes de terminais de acesso (set-top boxes). Resultado de pesquisas lideradas pela PUC-Rio e pela UFPB, o Ginga reúne um conjunto de tecnologias e inovações brasileiras que o tornam a especificação do middleware mais avançada e, ao mesmo tempo, mais adequada à realidade do país [10]. O middleware Ginga, por sua vez, encontra-se dividido em dois subsistemas principais: o Ginga-J e o Ginga-NCL [11], conforme ilustrado na Figura 1.

O Ginga-NCL permite o desenvolvimento de aplicações declarativas com a linguagem NCL (*Nested Context Language*) [11]. A linguagem NCL é uma linguagem declarativa baseada em XML (*eXtensible Markup Language*) [12] para autoria de documentos hipermídia que foi desenvolvida utilizando uma estrutura modular, seguindo os princípios adotados pelo W3C [13].

A programação declarativa em NCL possui vantagens como a especificação de aspectos de interatividade, sincronismo

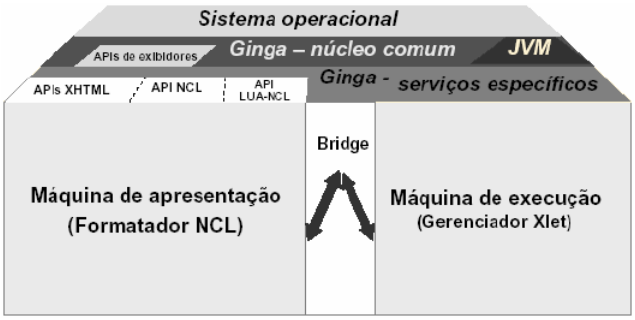

Fig. 1. Arquitetura do *middleware* Ginga

espaço-temporal entre objetos de mídia, adaptabilidade, suporte a múltiplos dispositivos, buscando a convergência digital, e suporte à produção ao vivo de programas interativos não-lineares [13]. Para estender as funcionalidades declarativas básicas da linguagem NCL é possível fazer uso de scripts escritos com a linguagem Lua [14], combinada à linguagem NCL.

O Ginga-NCL, por conta de sua característica declarativa, restringe o grau de interação dos usuários à interatividade local. Para o desenvolvimento de programas com interatividade intermitente ou permanente torna-se necessário escrever componentes em código Lua.

Já o Ginga-J é a porção do *middleware* responsável pela execução de código procedural no Ginga. Este módulo tornará possível a utilização da linguagem Java no desenvolvimento de software para televisão digital. O Ginga-J provê o suporte ao desenvolvimento de aplicações escritas com as APIs Java TV, Davic e HAVi (APIs para o desenvolvimento de aplicações de TV digital com Java). As aplicações desenvolvidas com o Ginga-J podem ainda ser invocadas a partir de aplicações declarativas, utilizando-se a "ponte" (*bridge*) do *middleware*.

Após uma série de problemas relacionados ao licenciamento da linguagem Java e de sua máquina virtual (JVM), o Fórum SBTVD e a Sun fecharam um acordo que culminou com a geração da especificação Java DTV pela Sun e sua adoção pelo Ginga-J. O Java DTV é um conjunto de APIs, que pode ser considerado uma única API dentro do Ginga-J. No entanto, até o presente momento não está disponível nenhuma implementação funcional desta especificação.

A combinação deste conjunto de tecnologias converge para um modelo de TV digital capaz de propiciar novas experiências aos telespectadores. Estas experiências, no entanto, dependem da disponibilização de mais e novos programas interativos e da popularização de técnicas e tecnologias envolvidos na construção destes.

III. FRAMEWORK DE COMUNICAÇÃO FCOM

Por conta da característica declarativa da linguagem NCL, torna-se necessária a existência de mecanismos que possam ser incorporados às aplicações NCL com vistas a facilitar a construção de aplicações de interatividade intermitente e permanente. Estes mecanismos devem poder ser facilmente acoplados às aplicações interativas, tanto do lado cliente quanto do lado servidor, de forma a prover funcionalidades de comunicação com servidores remotos.

O *framework* FCOM[1] foi desenvolvido com a linguagem de programação Lua e permite o desenvolvimento de aplicações para TV digital com interatividade intermitente e permanente de uma maneira mais fácil e rápida. Ele utiliza o paradigma cliente-servidor para a comunicação dos aplicativos em execução no *set-top box* com servidores remotos. Este *framework* foi desenvolvido em duas partes: uma parte a ser acoplada ao cliente e outra ao servidor. Dessa forma, os desenvolvedores de aplicações interativas poderão se concentrar apenas no desenvolvimento da lógica de negócio da aplicação.

Atualmente o *framework* FCOM utiliza uma interface de comunicação baseada em *sockets* (utilizando a biblioteca LuaSockets [15]) para a transmissão de mensagens em um formato XML (*eXtensible Markup Language*) [12] bastante compacto. Entre as razões que motivaram essa decisão de projeto em relação à adoção de outras tecnologias (e.g. serviços web), estão: (a) o fato do mecanismo de comunicação remota baseado em *sockets* consumir poucos recursos computacionais (que é um requisito dos *set-top* boxes); e (b) as limitações com relação ao consumo de recursos de transmissão, devido aos custos potencialmente envolvidos neste processo, o que requer formatos de comunicação reduzidos. Futuramente pretende-se investigar a adoção de outros formatos além do XML, como o formato JSON [16].

Diversas funcionalidades estão atualmente disponíveis no *framework* FCOM. Entre estas destacam-se:

- **Mecanismo de autenticação remota:** este mecanismo fornece um serviço de autenticação cujo protocolo de aplicação é bem definido. Recursos de criptografia serão futuramente incorporados à implementação;
- **Gerenciamento da comunicação com usuários e grupos:** este serviço permite que o FCOM gerencie a troca de informações entre clientes e grupos, facilitando o controle da comunicação do servidor com os clientes, de acordo com as especificidades do serviço sendo provido;
- **Gerenciamento de *black-lists*:** funcionalidade que permite ao servidor criar uma lista negra baseada em usuários ou mesmo em endereços IP, evitando que usuários "mal comportados" consigam se comunicar com o servidor;
- **Transmissão de mensagens de texto:** o mecanismo mais simples de troca de informações entre cliente e servidor é a transmissão de mensagens de texto. Através deste mecanismo mensagens de texto poderão ser trocadas e um protocolo de aplicação próprio pode ser definido pelo implementador da aplicação interativa;
- **Transmissão de "pacotes de informação":** os pacotes de informação são abstrações de conjuntos de pares "[variável, valor]" que podem ser utilizados para troca de parâmetros ou argumentos entre o cliente e o servidor. O FCOM disponibiliza uma API que permite a troca destes pacotes de informação de uma maneira bastante simples;
- **Chamada remota a procedimento:** este recurso permite que uma função ou procedimento seja invocado remotamente, executado no servidor e tenha seu resultado devolvido novamente ao cliente. Utiliza conceitos de SOA (*Service Oriented Architecture*) de forma similar a um serviço web;
- **Acesso a um banco de dados remoto:** funcionalidade que permite ao cliente realizar a conexão a um determinado banco de dados e executar consultas SQL (com posterior obtenção de seus resultados) através do servidor da aplicação. Recurso extremamente útil no desenvolvimento de uma larga gama de aplicações interativas (e.g. comércio eletrônico através da TV – chamado *t-commerce*);
- **Serviço de *logging*:** este mecanismo cria uma sucessão de informações de *log* que podem ser utilizadas para depurar a aplicação durante o processo de desenvolvimento ou mesmo realizar uma auditoria sobre as ações solicitadas ao servidor.

[1]Disponível em: http://fcom.dai.cefet-ma.br/

IV. Implementação do Framework de Comunicação FCOM

A implementação desses serviços está dividida entre as versões cliente e servidor do *framework* FCOM e baseia-se na utilização de *coroutines* (i.e. *threads* Lua). Um diagrama apresentando a arquitetura do FCOM é mostrado na Figura 2.

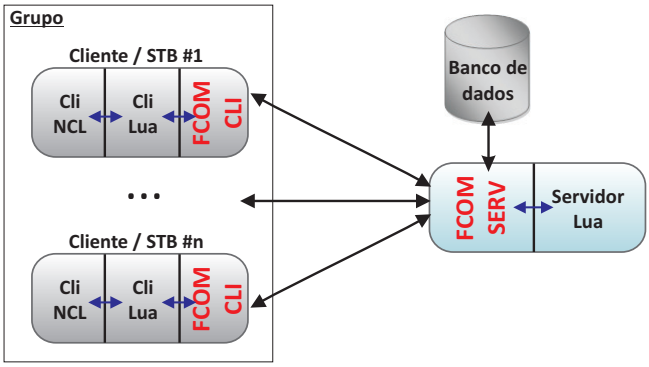

Fig. 2. Arquitetura do FCOM

As aplicações interativas desenvolvidas utilizando o FCOM devem ser escritas com a linguagem de programação Lua, tanto do lado cliente, quanto do lado servidor. Conforme pode ser visto na arquitetura do FCOM, é necessário que a aplicação interativa NCL (cliente) vincule um objeto de mídia do tipo *script* Lua e, a partir deste, passe a utilizar as funcionalidades da API do *framework*. Para fazer uso desta API, os *scripts* Lua da aplicação deverão utilizar o comando `require(script_FCOM)`[2].

As aplicações que executarão nos diversos *set-top boxes* (referenciados na Figura 2 como "STB") deverão ter três partes constituintes:

1) A aplicação declarativa NCL, responsável pelo gerenciamento da interface e da interação local com os usuários;
2) A aplicação Lua, que será invocada durante o ciclo de vida da aplicação como um objeto de mídia, responsável por implementar a camada de controle e interação entre a aplicação NCL e as funcionalidades do FCOM;

[2]O nome "`script_FCOM`" deve ser substituído pelo respectivo *script* do FCOM a ser utilizado

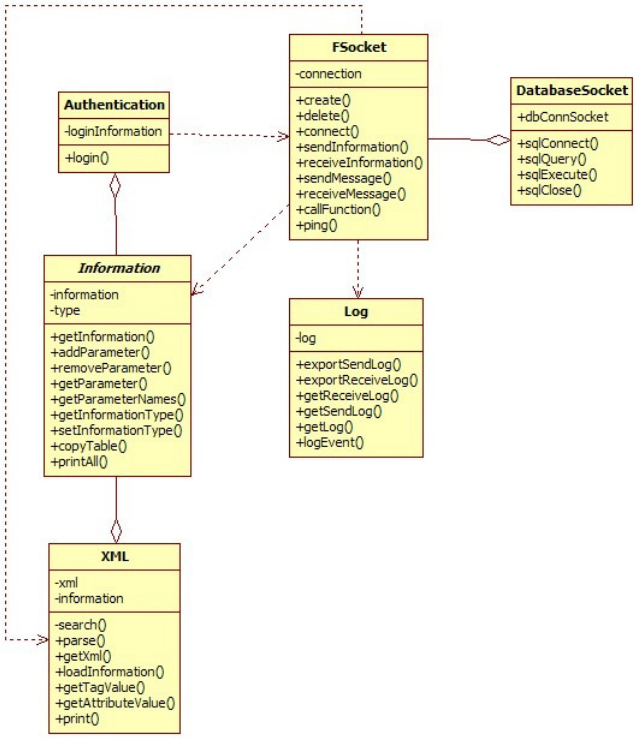

Fig. 3. Diagrama ilustrativo da relação entre os *scripts* Lua da porção cliente do FCOM

3) A API do FCOM, cuja implementação deverá ser incorporada à aplicação interativa.

A Figura 3 apresenta o diagrama que ilustra a relação entre os diversos *scripts* Lua que compõem a porção cliente do *framework* FCOM, bem como suas funções e tabelas[3] (apresentadas como atributos).

O principal *script* da porção cliente do *framework* é o `FSocket`. Ele fornece as principais funcionalidades do FCOM, principalmente no que concerne à troca de informações com o servidor remoto. Todas as operações realizadas por este geram *logs* através do *script* `Log`. Além disso, todas as solicitações realizadas à `FSocket` são transformadas em tabelas `Information` (informações), que contém pares *[variável, valor]* e um tipo associado, que define o tipo de informação que está sendo enviada (e.g. manipulação de banco de dados, chamada de função, parâmetros de autenticação, parâmetros da aplicação, etc).

Todas as informações enviadas são serializadas e transformadas para um formato XML através do *script* `XML`. Por fim, os *scripts* `Authentication` e `DatabaseSocket` fornecem interfaces que a aplicação poderá utilizar para realizar a autenticação e solicitações de acesso a um banco de dados remoto.

Já do lado do servidor deverão ser implementadas as diversas regras de negócio envolvidas na aplicação interativa. A implementação do servidor deverá fazer uso das funcionalidades disponibilizadas na porção servidor do *framework* para comunicar-se com as aplicações clientes. Dessa forma, o programador poderá se preocupar apenas com o desenvolvimento dos aspectos comportamentais do servidor (escritos em Lua), desobrigando-se a implementar os recursos de comunicação remota da aplicação. Um diagrama de classes que ilustra a relação entre os diversos *scripts* Lua que compõem a porção servidor do *framework* FCOM é apresentado na Figura 4.

A versão servidor do FCOM traz alguns *scripts* adicionais em relação à versão cliente. O *script* `GroupManager` fornece as funcionalidades de gerenciamento de grupos de clientes. As principais características de um grupo (contidas na tabela `groups`) são seu nome e uma lista de endereços IP representando cada cliente (ou usuário) do sistema. Já o *script* `IgnoredClient` permite o gerenciamento de informações das *black-lists* utilizadas pelo servidor, como o nome de usuário ou endereço IP, ou mesmo o tempo de permanência de um usuário na *black-list*.

No *script* `FunctionCall` é implementada a funcionalidade de chamada remota a procedimento. É este *script* que, efetivamente, realiza a execução de métodos desenvolvidos pelo programador da aplicação e retorna seus resultados. Por fim, o *script* `Database` permite ao servidor interagir com o banco de dados, mediante às solicitações dos clientes. A versão atual deste *script* permite o acesso nativo aos bancos de dados MySQL [18] e SQLite [19], além do acesso através de *drivers* ODBC (*Open Data Base Connectivity*).

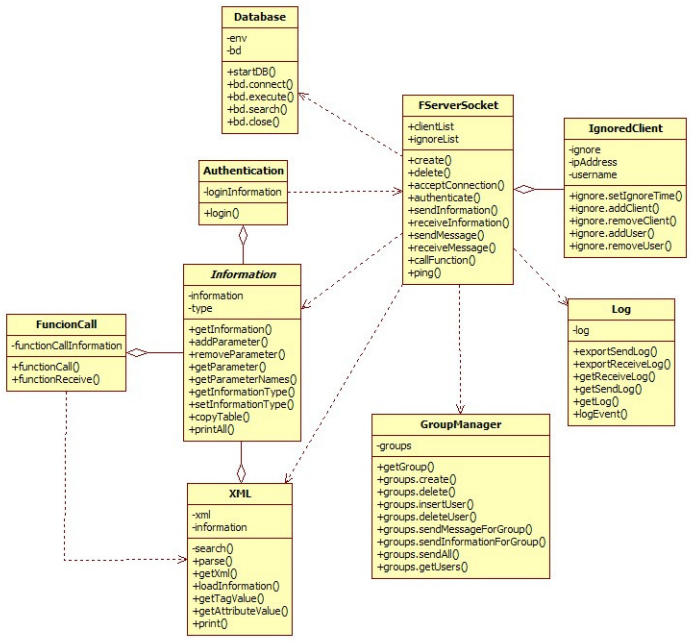

Fig. 4. Diagrama ilustrativo da relação entre os *scripts* Lua da porção servidor do FCOM

A. Aplicações desenvolvidas com o framework FCOM

Duas aplicações foram desenvolvidas com o intuito de validar a utilização do *framework* FCOM: (a) uma aplicação de chat e (b) uma aplicação de previsão do tempo. Ambas as aplicações foram executadas e testadas em uma máquina

[3]Tabelas em Lua representam o conceito de "objeto" da orientação a objetos [17]

virtual Ginga (*Ginga-NCL Virtual STB*), emulando o ambiente de um *set-top box* real.

Na aplicação de chat foi largamente utilizada a funcionalidade de gerenciamento de grupos de usuários. Após se autenticarem junto ao servidor, os clientes enviam uma informação contendo o nome do grupo ao qual fazem parte (e.g. `Information` [grupo, "SL"]). Estes usuários são então associados pelo servidor a cada um de seus respectivos grupos.

Ao enviar uma mensagem textual direcionada a um dado grupo, o servidor a retransmite a todos os usuários registrados neste grupo. Caso um usuário faça mal uso do chat é possível, ao administrador, enviar uma mensagem ao servidor solicitando a adição deste usuário à *black-list*. A Figura 5 apresenta uma tela da aplicação de chat desenvolvida com o uso do FCOM.

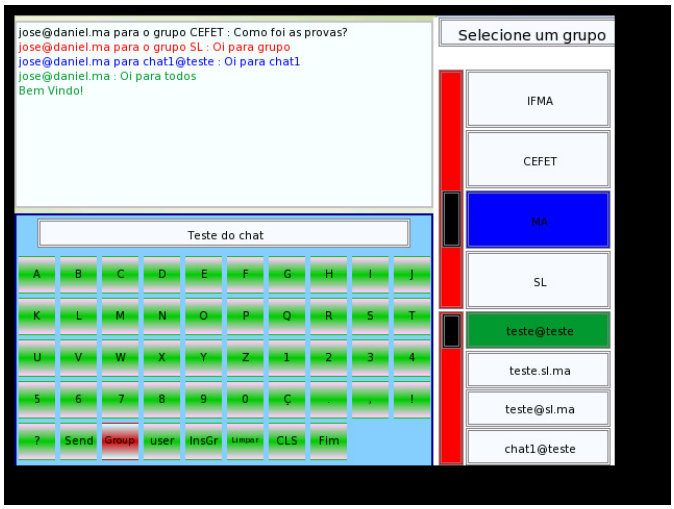

Fig. 5. Tela da aplicação de chat desenvolvida com o uso do FCOM

Já a aplicação de previsão do tempo tem seu funcionamento baseado na funcionalidade de chamada remota a procedimento. Ao acessar o sistema, é exibida a lista de cidades e o mapa do estado do Maranhão[4]. A partir desta tela o usuário pode selecionar alguma cidade na lista. Esta ação dispara uma chamada a uma função remota que retorna uma tabela contendo as informações climáticas da localidade selecionada. Essa funcionalidade pode ser observada na Figura 6, que apresenta uma tela da aplicação de previsão do tempo.

É possível observar que o leque de oportunidades provido pelo *framework* FCOM é bastante largo. Brevemente, pretende-se desenvolver uma aplicação de *t-commerce* utilizando os recursos de acesso a banco de dados.

V. Conclusões e Trabalhos Futuros

A implantação do sistema brasileiro de TV digital e de seu padrão ISDTV-T permitirão o desenvolvimento de um novo panorama para a TV nacional. Neste novo cenário, o telespectador tornar-se-á um sujeito ativo em sua relação com os programas televisivos. Isto será possível graças aos recursos

[4]Considerando que o objetivo da aplicação era apenas validar o *framework*, a aplicação foi desenvolvida com o escopo apenas para este estado

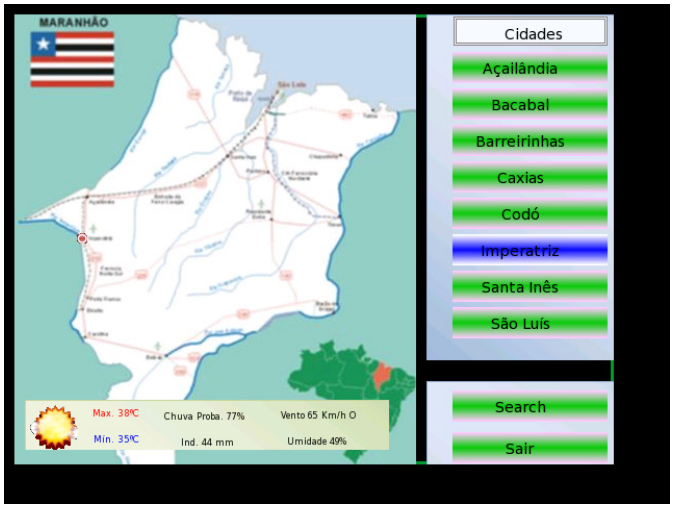

Fig. 6. Tela da aplicação de previsão do tempo desenvolvida com o uso do FCOM

de interatividade da plataforma, que cria oportunidades de negócios, aprendizagem, entretenimento e inclusão social.

Nesse contexto, o *framework* FCOM pretende contribuir com a comunidade de desenvolvedores de programas interativos para TV digital, provendo mecanismos simples e robustos de comunicação com servidores remotos para aplicações interativas. No intuito de auxiliar o desenvolvimento de aplicações televisivas que utilizem recursos de interatividade intermitente e permanente, este *framework* permite que aplicações possam fazer uso de todas as capacidades do canal de interatividade da TV digital para fornecer os mais diversos tipos de serviços.

Este artigo apresentou o FCOM, sua arquitetura e implementação. Além disso, foram apresentadas algumas aplicações de referência desenvolvidas com o auxílio do *framework* e alguns aspectos de implementação das mesmas. O FCOM fornece diversos recursos importantes no contexto das aplicações interativas para TV digital.

Além das funcionalidades implementadas no escopo deste trabalho, pretende-se futuramente investigar alguns aspectos, como: a incorporação de recursos de criptografia ao mecanismo de autenticação; a criação de diversos níveis hierárquicos de usuários; desenvolvimento de recursos automáticos para a inclusão de usuários na *black-list*, com base nas mensagens enviadas por este; desenvolvimento de uma versão Java da porção servidor do *framework*; a utilização do padrão JSON [16] em relação ao XML para transmissão de informações; o desenvolvimento de recursos de comunicação P2P (*peer-to-peer*) com vistas a criar uma nova classe de aplicações para a TV digital, envolvendo a comunicação não só entre clientes e um servidor, mas também entre os próprios clientes. Além disso, pretende-se avaliar o consumo de recursos de processamento e memória do *framework*.

Agradecimentos

Os autores agradecem pelo financiamento da bolsa de Iniciação Científica fornecida pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) ao aluno José Daniel. Adicionalmente, os autores agradecem ao Instituto

Federal do Maranhão (IFMA) pela infraestrutura de pesquisa disponibilizada ao aluno.

REFERÊNCIAS

[1] ALENCAR, M. S. *Televisão Digital*. Editora Érica, 2007.

[2] FERNANDES, J.; LEMOS, G.; SILVEIRA, G. *Introdução à televisão digital interativa: arquitetura, protocolos, padrões e práticas*. In: Anais do Congresso da Sociedade Brasileira de Computação. Salvador: [s.n.], 2004. p. 24. Disponível em: `http://www.cic.unb.br/~jhcf/MyBooks/itvdi`. Acesso em setembro de 2008.

[3] SOUZA, C. T; OLIVEIRA, C. T. *Especificação de Canal de Retorno em Aplicações para TV Digital Interativa*. Anais do XXII Simpósio Brasileiro de Telecomunicações - SBrT'05. Campinas, 2005.

[4] RESENDE, L. E. A. *Desenvolvimento de uma ferramenta de análise de desempenho para o padrão de TV Digital ISDB-T*. Dissertação de Mestrado, PUC-Rio. Maio, 2004.

[5] HASKELL, B. G.; PURI, A.; NETRAVALI, A. N. *Digital video: an introduction to MPEG-2*. 5th edition. Springer, 1997.

[6] ISO/IEC 13818-6:1998. *Information technology – Generic coding of moving pictures and associated audio information – Part 6: Extensions for DSM-CC (available in English only)*. ISO/IEC, 1998.

[7] BRASIL. *Decreto 5.820, de 29 de junho de 2006. Dispõe sobre a implantação do SBTVD e sobre a transição para este novo sistema*. Disponivel em: `http://www.planalto.gov.br/ccivil/_Ato2004-2006/2006/Decreto/D5820.htm`. Acesso em abril de 2008.

[8] SOARES, L. F. G; SOUZA, G. L. *Interactive Television in Brazil: System Software and the Digital Divide*. 5th European Conference on Interactive TV (EuroiTV), 2007.

[9] BARBOSA, S. D. J.; SOARES, L. F. G. *TV digital interativa no Brasil se faz com Ginga - Fundamentos, Padrões, Autoria Declarativa e Usabilidade*. In: Sociedade Brasileira de Computação. (Org.). Jornada de Atualização em Informática 2008.

[10] GINGA. *Comunidade Ginga do Portal do Software Público Brasileiro*. Disponível em: `http://www.softwarepublico.gov.br/dotlrn/clubs/ginga/one-community?page_num=0`. Acesso em maio de 2009.

[11] *ABNT NBR 15606-2:2007. Televisão digital terrestre - Codificação de dados e especificações de transmissão para radiodifusão digital - Parte 2: Ginga-NCL para receptores fixos e móveis - Linguagem de aplicação XML para codificação de aplicações*. ABNT, 2007.

[12] HUNTER, D.; RAFTER, J.; FAWCETT, J. *Beginning XML*, 4ed. Indianapolis: Wiley Publishing, 2007.

[13] NCL. *Portal da linguagem NCL (*Nested Context Language*)*. Disponível em: `http://www.ncl.org.br/`. Acesso em maio de 2009.

[14] LUA. *The Lua Programming Language web site*. Disponível em: `http://www.lua.org/`. Acesso em maio de 2009.

[15] LuaSocket. *LuaSocket: Network support for the Lua language*. Disponível em: `http://www.tecgraf.puc-rio.br/luasocket/`. Acesso em maio de 2009.

[16] JSON. *RFC 4627 – The application/json Media Type for JavaScript Object Notation (JSON)*. Network Working Group, Internet Society. July, 2006.

[17] IERUSALIMSCHY, R. *Programming in Lua – online book*. 1st edition. ISBN 85-903798-1-7. December, 2003. Disponível em: `http://www.lua.org/pil/index.html`. Acesso em maio de 2009.

[18] MySQL. *Portal do servidor de banco de dados MySQL*. Disponível em: `http://www.mysql.com/`. Acesso em maio de 2009.

[19] SQLite. *Portal do servidor de banco de dados SQLite*. Disponível em: `http://www.sqlite.org/`. Acesso em maio de 2009.