HHS Sticker: Aplicação Médica em Java para TV Digital

Claudia O. Neves[1], Thales V. G. da Silva[1], Roberto C. R. Machado[2], Alexandre Sztajnberg[1, 2, 3]

[1]Bacharelado em Ciência da Computação, DICC/UERJ
[2]Mestrado em Engenharia Eletrônica - PEL/FEN
[3]Mestrado em Ciências Computacionais - CComp/IME
Universidade do Estado do Rio de Janeiro - UERJ
CEP - 20550-900 – Maracanã, Rio de Janeiro – RJ – Brasil

claudiadasneves@gmail.com, thvela@gmail.com,
roberto.rodrigues@uerj.br, alexszt@ime.uerj.br

Abstract. *This paper presents an interactive DiTV application to support the remote monitoring of elderly patients in their homes The application is able to receive relevant information such as the Care Plan directed by medical personnel and real-time alerts to notify the patient of any routine to be executed. The application also allows sending physiological measurements performed by the patient or caregiver to a Monitoring Central. This requires the installation of a digital TV running the Ginga middleware, and Internet access. The application, developed in Java facilitates the patient interacting with the application with no need to learn a new technology, via the TV remote. This application is integrated with a broader patient monitoring system, called HHS (Home Health System).*

Resumo. *O presente artigo apresenta uma aplicação interativa para TVDi para o acompanhamento remoto de paciente idosos em suas residências A aplicação é capaz de receber dados relevantes, tais como o Plano de Cuidados indicado pela equipe médica e alertas em tempo real para notificar o paciente de alguma rotina que deve ser executada. A aplicação também permite o envio de medidas fisiológicas realizadas pelo paciente ou cuidador para uma Central de Monitoramento. Para isso é necessária a instalação de uma TV digital com o middleware Ginga, e acesso à Internet. A aplicação, desenvolvida em Java, cria facilidades, via controle remoto da TV, para o paciente interagir com a aplicação, sem a necessidade de aprender uma nova tecnologia. Esta aplicação é integrada a um sistema de monitoramento de pacientes, chamado de HHS (Home Health System).*

1. Introdução

A televisão, presente em quase todas as residências segundo IBGE [IBGE, 2011], é uma mídia de fácil acesso e simples manuseio. Com o encerramento das transmissões analógicas previsto para 2016 [Enc, 2012], televisores analógicos serão substituídos pelos digitais, que além de reproduzirem imagens com maior qualidade, possibilitam o desenvolvimento de aplicativos interativos.

Na arquitetura proposta para a TV Digital (TVDi) os canais de descida (*emissora-usuários*) podem transportar conteúdo de áudio e vídeo, e também dados e aplicações, que devem aderir ao padrão Ginga. Esses conteúdos devem ser produzidos, estruturados e transmitidos pela emissora, e serão recebidos, processados, exibidos e executados no receptor digital ou *set-top box* do usuário. Neste caso, a possibilidade de interatividade se dá através dos dados recebidos junto com o áudio e vídeo e também através dos programas, recebidos da emissora, que podem ser executados localmente no equipamento do usuário. Entretanto, não é possível, com estes mecanismos continuar a interação, pois seria necessário uma forma de retorno, um mecanismo de comunicação para troca de mensagens. Alguns *set-top boxes* disponíveis oferecem esta possibilidade, chamada de canal de retorno, provendo em seu sistema mecanismos de conectividade, através de interfaces de rede. Neste caso, se o usuário possui conexão à Internet, é possível configurar o *set-top box* como um ponto da rede. Com isso, abre-se a possibilidade para a execução de programas na televisão, que (i) não dependam de uma emissora para serem recebidos, isto é, podem ser carregados (download) de um repositório da Internet e (ii) que necessitem de troca de mensagens, com um banco de dados ou com um servidor para receber ou enviar dados, requisições e respostas.

A interatividade possibilitada pelo sistema de TVDi com canal de retorno, tem o potencial para levar aplicações sérias a uma parte da população que não tem intimidade com tecnologia ou acesso à Internet com facilidade. Entre estas aplicações podemos citar o acesso a instituições bancárias, aplicações de Governo Digital (e-Gov) ou TeleSaúde (e-Health). Por outro lado, os lares brasileiros com acesso à Internet ainda são em número reduzido, se comparado aos que possuem aparelhos de TV. Por isso, são necessários investimentos para popularizar aparelhos de TVDi e o desenvolvimento do suporte ao canal de retorno usando tecnologias mais acessíveis ou subsidiadas como WiMAX, 3G, UHF, etc; visto que as tecnologias atuais, como: ADSL ou *cable modems*, ainda têm custo restritivo.

Na reportagem veiculada pelo Banco Mundial sobre TV Digital Brasileira [The World Bank Group, 2013] são apresentados os serviços de e-GOV providos pela EBC, Empresa Brasileira de Comunicação, grupo de mídia estatal, que oferece acesso interativo a vários serviços para o cidadão, incluindo serviços de saúde. O relatório *Brasil 4D: Estudo de impacto socioeconômico sobre a TV digital pública interativa* [World Bank, 2013] oferece mais informações sobre as primeiras avaliações do SBTD (Sistema Brasileiro de TV Digital). Um dos pontos de avaliação são as aplicações na área de saúde que executam sobre o Ginga, o sistema de *middleware* proposto para o sistema de TV Digital do Brasil. Os benefícios para a população de baixa renda são indiscutíveis. Mas a maioria das aplicações avaliadas é limitada à oferta de informação direcionada por área. Não foram avaliadas nesse relatório aplicações estritamente interativas.

O presente trabalho apresenta uma aplicação para TVDi para o acompanhamento remoto de pacientes idosos em suas residências A aplicação é capaz de receber dados relevantes, tais como o Plano de Cuidados indicado pela equipe médica, e alertas em tempo real para notificar o paciente de alguma rotina que deve ser executada. A aplicação também permite o envio de medidas fisiológicas realizadas pelo paciente ou cuidador para uma Central de Monitoramento. Para isso é necessária a instalação de uma TV digital com o *middleware* Ginga, e acesso à Internet. A aplicação desenvolvida

cria facilidades, via controle remoto da TV, para o paciente interagir com a aplicação, sem a necessidade de aprender uma nova tecnologia. Esta aplicação é integrada a um sistema de monitoramento de pacientes, chamado de HHS (*Home Health System*) [Sztajnberg, 2009].

A arquitetura da aplicação é apresentada, junto com um protótipo desenvolvido em Java, utilizando a plataforma Sticker Center [StickerCenter, 2013], da TQTVD [TQTVD, 2012] aderente ao padrão Ginga. Por esta razão a aplicação é chamada *HHS Sticker*. Elementos que permitem sua integração ao HHS também foram desenvolvidos, incluindo o acesso ao banco de dados e o mecanismo para enviar notificações de alerta ao paciente, que podem aparecer no *HHS Sticker* durante sua novela favorita, para lembrá-lo de uma medicação, por exemplo.

Este artigo está estruturado da seguinte forma. Na Seção 2 apresentam-se alguns conceitos do padrão brasileiro de TV Digital e seus componentes. Na sequência, Seção 3, é apresentado o sistema de monitoramento remoto de pacientes e descrita a arquitetura do HHS *Sticker*. Na Seção 4 é apresentada a arquitetura do sistema. A Seção 5 apresenta o protótipo implementado e uma discussão sobre detalhes da comunicação entre as partes, bem como, uma descrição de sua interface. Na Seção 6 é feito um rápido comparativo com outros trabalhos relacionados e na Seção 7, são apresentadas as considerações finais e algumas sugestões de trabalhos futuros.

2. Padrão Brasileiro de TV Digital

Um sistema de TV Digital consiste na transmissão de fluxos de bits contendo áudio, vídeo e dados de uma emissora para os aparelhos de TV. Para exibir o conteúdo, os aparelhos de TV precisam estar preparados para receber o sinal digital das emissoras, seja usando um conversor digital integrado ou um conversor externo, chamado de *set-top box (STB)*.

No Brasil, conversores seguem o padrão Brasileiro – SBTVD [HIST 2013] de TV Digital chamado comercialmente de ISDB-TB, que se baseia no padrão japonês, acrescido de algumas novas tecnologias e melhorias, como maior capacidade por canal usando compressão de vídeo MPEG-4 AVC (H.264) e maior robustez através do *middleware* Nipo-Brasileiro de especificação aberta "Ginga". Esse, desenvolvido pelos laboratórios Telemídia (da PUC-Rio) e LAViD (da UFPB) seguindo recomendação ITU-T [ITU, 2013] para serviços IPTV [IPTV, 2013] e permitindo aplicações interativas complexas em um ambiente também procedural [Grupos de Trabalho na SBTVD, 2011].

Em sua especificação 1.0, o Ginga apresenta o ambiente Ginga-NCL como subsistema lógico obrigatório, responsável pela execução de aplicações na linguagem declarativa LUA/NCL (Nested Context Language), desenvolvida pela PUC-Rio (Figura 1). Sua arquitetura permite a adição de extensões opcionais, entre elas o subsistema e ambiente Ginga-J [Ginga, 2013], desenvolvido pela UFPB (Ginga 2.0) que é responsável pela execução de aplicações na linguagem procedural Java. Essas aplicações são entregues aos dois ambientes pelo terceiro subsistema Ginga-CC (Ginga Common-Core), que cuida da exibição de vários formatos de mídia, do controle do plano gráfico para a especificação do ISDB-TB e também do acesso ao Canal de

Interatividade ou Retorno (responsável por controlar o acesso à camada de rede), previsto no padrão.

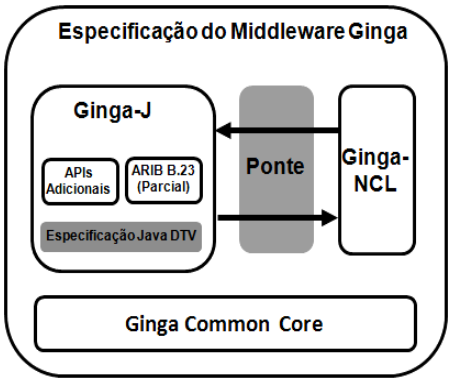


Figura 1. Especificação Ginga [Esp, 2009]

3. Ambiente de Desenvolvimento

Para o desenvolvimento da aplicação foi utilizada a plataforma de execução e desenvolvimento chamada AstroTV, aderente ao padrão Ginga e um *set-top box* com o AstroTV embarcado e com canal de retorno disponível.

3.1 AstroTV

O AstroTV [AstroTV, 2012] é uma implementação de *middleware* para TVDi no padrão Brasileiro, compatível com a especificação Ginga. O AstroTV é atualmente embarcado em *set-top boxes* e televisores em parceria com diversos fabricantes (Sony, LG, Panasonic, Philips, Toshiba, Sharp, D-Link) e possui recursos como: interatividade, portabilidade (interatividade em dispositivos móveis) e programação estendida em entretenimento e informação (acréscimo de conteúdo pela emissora).

A TQTVD também disponibiliza o AstroBox [AstroBox, 2011], uma ambiente virtualizado para simulação do ambiente AstroTV, distribuído com Linux Ubuntu 10.04 [Ubuntu, 2010] e executado através de uma máquina virtual, Virtual Box [V-Box, 2013]. Com o AstroTV e o AstroBox é possível executar *Stickers* (detalhes na Seção 3.3), que também são aplicações Ginga, e podem ser incorporados em qualquer receptor que possua o AstroTV e que dê suporte à solução de *Broadband* e *Broadcast* da TQTVD (o *StickerCenter*) [StickerCenter, 2013].

3.2 Java DTV

O Ginga-J, suporte Java ao Ginga, baseia-se nos pacotes do padrão JavaDTV [Kulesza, 2009]. Este padrão, por sua vez, usa como base a API LWUIT (*Lightweight User Interface Toolkit*), desenvolvida pela Sun. Diversos componentes da API JavaDTV foram utilizadas na aplicação desenvolvida, entre eles os componentes gráficos (*buttons*, *checkboxes*, *dropdown*, etc.), gerenciadores de *layout* (*menus*, *tabs,* etc.), estilos, temas, transição de tela animada e um modelo de tratamento de eventos bastante simples, similar a um *Applet*.

3.3 Stickers e a Plataforma StickerCenter

Para proporcionar um ambiente de programação produtivo, com suporte às aplicações Ginga Lua/NCL e Ginga-J, a TQTVD criou o conceito de *Sticker* [Ubuntu, 2010], com o suporte de um ambiente de execução e um repositório para download na web, chamado *StickerCenter* [StickerCenter, 2012].

Um Sticker (Figura 2) é uma aplicação interativa Ginga (Lua ou Java) similar em conceito a uma aplicação para celulares e *tablets* ou a um *Widget* - pequeno aplicativo com funcionalidade limitada, instalado e executado dentro de uma página web por um usuário final. Através dele é possível: acessar conteúdos complementares à programação da TV, participar de votações, acessar Internet, etc. *Stickers* podem ser adquiridos ou atualizados via *StickerShop*, baixados para o receptor via canal de interatividade (via Internet) ou através do canal de TV digital terrestre transmitido por emissoras que suportem a plataforma *StickerCenter*.

Figura 2. Sticker e suas áreas identificadas [UBU 2010]

Os Stickers seguem padrões de consistência visual, funcional e de navegação, com funções comuns para facilitar o aprendizado e a experiência de seu uso aos usuários. Elementos que constituem esse padrão são fixos e não podem ser customizados. Por exemplo, fontes, como ilustradas na Figura 2, não possuem variações de negrito e itálico e de tamanho, para proporcionar uma leitura confortável na TV. Abas são numeradas para serem acessadas diretamente pelo controle remoto.

Podemos interpretar o *Sticker* como uma padronização ou wrapping de aplicações TVDi. Para implementar a aplicação, propriamente dita, em Java, a classe principal deve implementar a interface *Xlet* [Xlet, JSR 217]. Para uma aplicação de

TVDi se caracterizar como um Sticker é necessário utilizar o pacote *com.tqtvd.stickercenter*.

Para que um Sticker esteja disponível para download e instalação em set-top boxes é necessário que o desenvolvedor faça *upload* para o site do *StickerCenter* de um arquivo compactado contendo os arquivos: imagens "ico.png" (96 x 84 pixels), "exibition.png" (66 x 55 pixel), "screenshot.png" (114 x 269 pixels) e uma pasta *source* contendo os códigos e imagens utilizadas na aplicação.

3.4 A Interface Xlet

A interface *Xlet,* para programação de aplicações de TVDi em Java, tem a estrutura similar à da classe Applet. Cada *Xlet* possui uma instância da classe *javax.microedition.xlet.XletContext*, que é um contexto associado, usado para prover um modo para a aplicação obter informações sobre o seu ambiente, além de comunicar mudanças de estados (Figura 3) em seu ciclo de vida.

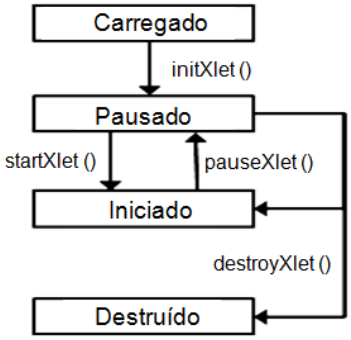


Figura 3. Ciclo de vida dos Xlets [Morris, 2012]

O *middleware* instancia a classe *Main* e identifica nela, o ponto de entrada da aplicação (classe *javax.microedition.xlet.Xlet*) e executa a aplicação utilizando a máquina virtual Java. A classe *Main* invoca os métodos responsáveis pela mudança de estados [Batista, 2007] e é responsável por gerenciar o ciclo de vida das *Xlets* conforme figura 4*;* controle este, semelhante ao dos *MIDlets* – aplicativos Java para dispositivos móveis – [Midlet, 2013]. Ambos oferecem a funcionalidade de pausar as aplicações em inatividade (não visíveis) com objetivo de liberar recursos em ambientes onde haja restrições de hardware [Kulesza, 2009].

```
public class Main implements Xlet{
   public void startXlet() throws XletStateChangeException { }
   public void pauseXlet() {  }
   public void destroyXlet(boolean arg0)
            throws XletStateChangeException { }
(...)
```

Figura 4. Assinatura da classe *Main* gerenciadora do ciclo de vida das *Xlets*

3.4 Set-top Box

O *set-top box* modelo DTB-331 [D-Link, 2013] foi utilizado para avaliar a aplicação. Assim como todo set-top box (STB) ele possibilita que televisões digitais ou analógicas que não possuam conversor digital integrado apresentem o conteúdo enviado pelas

emissoras via SBTVD. Ele possui o AstroTV embarcado, com suporte ao Ginga-J, e interfaces de rede Ethernet e WiFi. Com isso, foi possível contar com o canal de retorno ou de interatividade para retornar os dados selecionados ou incluídos pelo usuário ao banco de dados do HHS, hospedado em um servidor remoto. O usuário acessa e interage com o *HHS Sticker* através do controle remoto da TV.

4. HHS Sticker

O sistema *Home Health System* (HHS), no qual o HHS Sticker está inserido, tem o objetivo de monitorar, coletar, armazenar, avaliar e distribuir dados de contexto, compostos por informação de sensores (temperatura, luminosidade, umidade) dados de atividades do paciente, sua localização e medições fisiológicas (pressão arterial, ritmo cardíaco, etc.). Esses dados são enviados pela Internet para uma Central de Monitoramento, onde os mesmos são avaliados e persistidos em uma base de dados.

Os dados são disponibilizados para médicos e outros profissionais de saúde envolvidos no tratamento, que possuem históricos individualizados de cada paciente [Sztajnberg, 2009]. Os próprios pacientes, familiares e cuidadores também podem ter acesso a esses dados, porém de forma restrita, segundo orientação médica. O sistema pode enviar notificações aos pacientes de forma automática, segundo um Plano de Cuidados determinado pelo médico, ou manualmente (médicos e cuidadores).

No HHS é pressuposta uma infraestrutura instalada na residência do paciente (Figura 5) para coletar e monitorar os dados de contexto, composta por sensores de ambiente, pontos de acesso Wi-Fi para comunicação dos sensores e atuadores com um sistema de computação local [Macedo, 2011], que persiste as informações temporariamente e as transmite com segurança para a Central de Monitoramento.

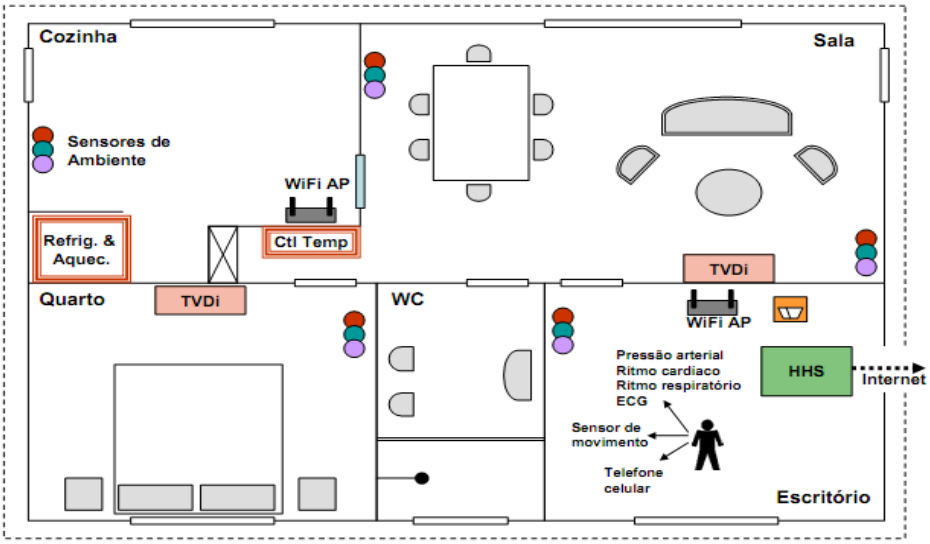


Figura 5. Sistema de Monitoramento Domiciliar da Saúde [Macedo, 2011]

A proposta do presente trabalho introduz o uso de TVs Digitais, capazes de receber e exibir mensagens, integrados à infraestrutura do HHS, permitindo a interação de pacientes, familiares e cuidadores com o sistema através de um equipamento que já

faz parte de seu cotidiano. Como preparação preliminar, cada *set-top box* da residência deve ser configurado para acessar o Sticker Center de onde o HHS Sticker será obtido.

4.1 Arquitetura

A arquitetura do sistema proposto neste trabalho é apresentada na Figura 6 e integra três elementos: (i) o cliente para apresentação de dados e interação paciente-médico, mais especificamente o próprio *HHS Sticker*, (ii) o servidor que recebe os dados coletados dos vários pacientes para persistência e realiza a comunicação com o *HHS Sticker* (uma extensão do sistema da Central de Monitoramento do HHS) e uma interface de suporte para os médicos e monitores enviarem mensagens e notificações ao HHS Sticker de forma simples. Essa função de envio de mensagens é integrada à interface de acesso do HHS.

O *HHS Sticker* foi desenvolvido em Java versão 1.3, por questões de compatibilidade com a JVM embarcada no *set-top box*. Deve ser carregado (*download)* a partir do site do Sticker Center através de conexão à Internet, configurada no *set-top box*. Os outros módulos de usuário e suporte, disponíveis no HHS, executam em computadores e notebooks que possuem mais recursos se comparados com o *set-top box*, e também foram desenvolvidos em Java (versão 1.6).

O *HHS Sticker* e os demais módulos comunicam-se com o servidor na Central de Monitoramento, que é responsável pela autenticação do usuário, a gerência de dados e notificações e o acesso ao banco de dados. Toda comunicação é realizada através de socket TCP.

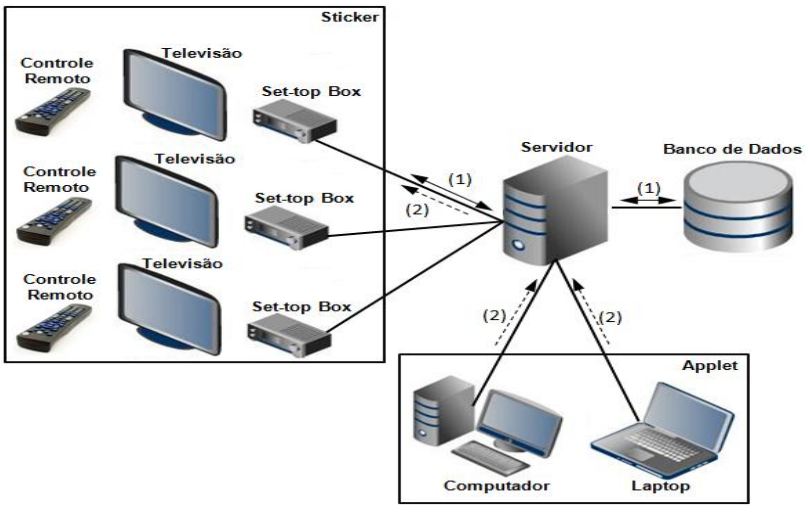


Figura 6. Arquitetura Geral da comunicação Cliente-Servidor

Após o *download* pelo canal de interação e a instalação no *set-top box*, o HHS *Sticker* pode ser acionado no aparelho pelo controle remoto. Através do endereço padrão da Central de Monitoramento embutido no *HHS Sticker*, ou introduzindo um endereço alternativo (diferentes provedores de cuidados médicos), qualquer usuário pode iniciar uma conexão com o servidor.

Estando o Sticker ativo e conectado, as interações de usuários com o sistema se dão conforme a Figura 6. O paciente pode navegar pelas abas da aplicação e decidir enviar medidas manuais e acessar as informações desejadas disponíveis no banco de

dados, como: plano de atividades, avisos e plano de cuidados (1). O médico, os monitores ou o próprio sistema podem enviar mensagens para o Sticker, notificando o paciente com alguma informação sobre o plano de cuidados (2).

4.2 Estrutura do Sistema

O *HHS Sticker*, módulo cliente do sistema, adere à estrutura básica de um *Xlet*, e faz uso de elementos adicionais para as funções de *Sticker* e para a comunicação com os demais módulos do sistema (Figura 7).

A classe *Main* é o *entryPoint*, que implementa a interface *Xlet*, instancia um objeto *Sticker*, realiza a configuração do mesmo seguindo seus padrões e caracterizações. A classe *RequestManager* é responsável por gerenciar requisições vindas do cliente, um objeto da classe *PageManager* (do pacote *StickerCenter*), que também implementa a interface *IResponseListener.* Ao receber uma requisição, salva referências do objeto requerente em uma lista de espera (método *addRequestReference* do *ResponseReceiver*) e em seguida chama o objeto *Connection* encarregado de estabelecer, manter ou retomar uma conexão com o Servidor, passando um objeto *Request* como parâmetro no método *sendRequest*. Essa referência do objeto, juntamente com seu código, também são salvos em uma lista pela classe *ResponseReceiver* que é acionada novamente assim que uma resposta for recebida, procurando o objeto requerente que espera pela resposta (um *IResponseListener*) e o notifica. Com esta abordagem, mesmo que o usuário troque a página ativa (*ActivePage*) a resposta é encaminhada ao elemento (*PageManager*) responsável pela requisição.

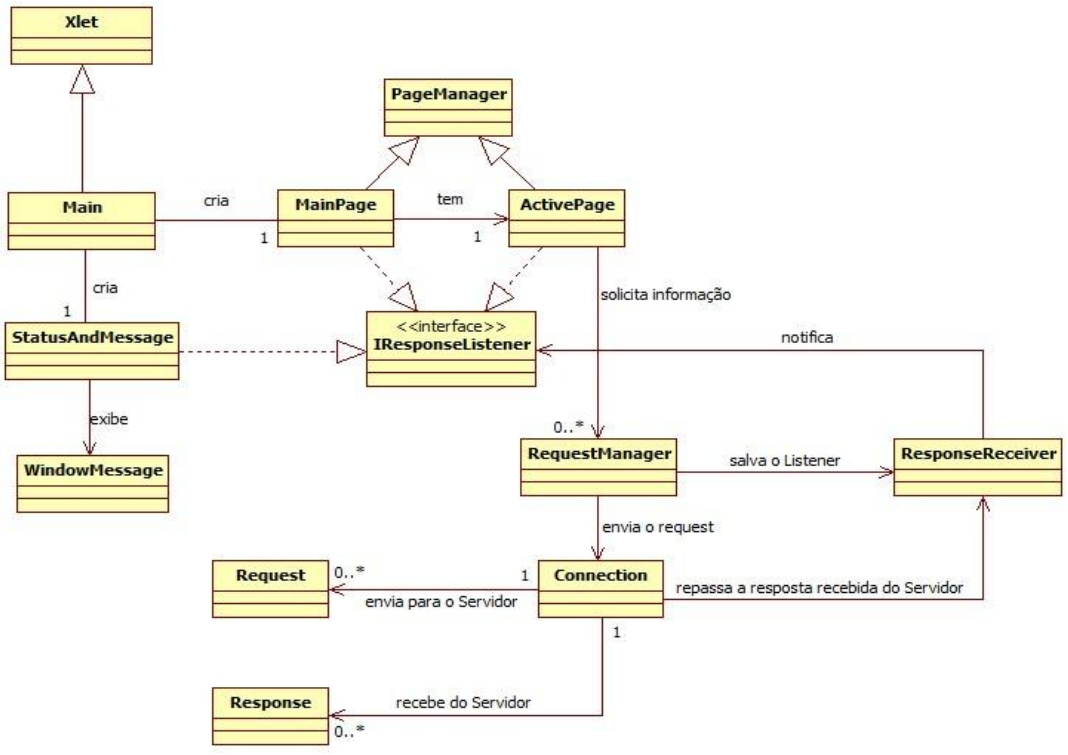


Figura 7. Estrutura do Sticker

Assim que *o HHS Sticker* inicia sua execução, uma conexão é estabelecida com o Servidor. As informações de endereço IP e número de porta para conexão com o Servidor são conhecidos e a classe *Connection* entra em execução para estabelecer a execução. Na primeira interação entre o *HHS Sticker* e o Servidor são informados os dados para *login* (usuário e senha cadastrados), obtidos na ação de enviar dados na *MainPage*. Uma vez estando o usuário conectado e autenticado, um canal persistente se forma e o *HHS Sticker* pode enviar requisições e dados, e receber informações e mensagens de notificação (Figura 8).

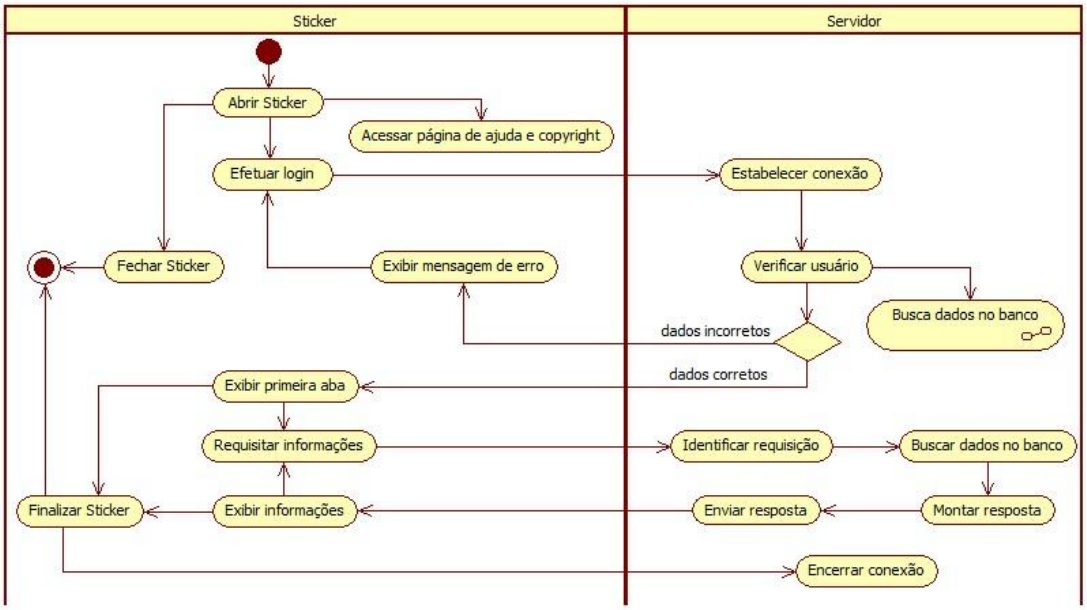


Figura 8. Fluxo de interação entre o HHS Sticker e o Servidor

Todas as mensagens recebidas pelo *HHS Sticker* chegam à classe *Connection*, que repassa as mesmas para a classe *ResponseReceiver*. Esta identifica qual objeto está esperando tal mensagem e repassa as informações para ele através de um evento de notificação. Cada mensagem tem um número de identificação que define se ela é uma resposta a algum pedido de um *PageManager* ou se é uma mensagem vinda direto do Servidor, como um pedido de Status ou de Aviso. Se a resposta for para um *ActivePage* (página *PageManager* que está ativa no momento) ela atualiza as informações na tela e se for uma mensagem ou verificação de status a classe *StatusAndMessage* será notificada e irá responder à verificação de status ou abrir uma janela na página ativa informando o conteúdo recebido.

A classe *Server* é a classe base do Servidor (Figura 9), responsável por aceitar conexões dos *Stickers*, e criar um *socket* TCP/IP ativo e passá-las adiante para um tratador *ServerConnection*, uma *thread* que por sua vez gerencia a troca de mensagens (requisições e respostas) desta conexão. Ao receber uma requisição o *ServerConnection* delega a execução da requisição ao objeto *ResponseManager*, chamando seu método *executeRequest* e aguarda a resposta para enviá-la ao cliente *Sticker*.

A classe *ResponseManager* valida o *token* de identificação do paciente e identifica o tipo de requisição recebida, passando-o para o método *ResponseMaker* que,

por sua vez, processa a requisição específica e devolve um *Response*. O *ResponseMaker* é um elemento importante na integração da aplicação HHS Sticker com o sistema HHS e o servidor de banco de dados. Para isso, ele utiliza a biblioteca DAO, comum a todos os módulos do sistema HHS. Em princípio, toda a funcionalidade do Servidor poderia estar embutida no sistema HHS. Entretanto, um servidor específico com finalidade de tratar os dados dos pacientes permitiu uma avaliação isolada do comportamento da aplicação sem dependência com o resto do sistema HHS que trata os dados dos sensores e, ainda assim, integrado ao mesmo.

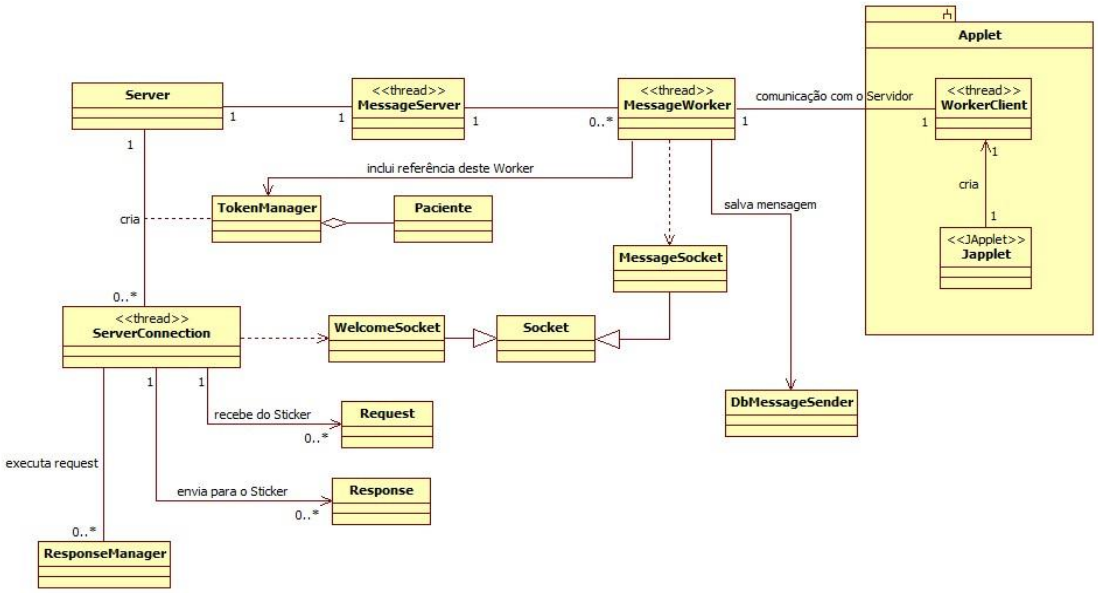


Figura 9. Estrutura do Servidor

A classe Server também é encarregada da criação do servidor de mensagens (uma *thread* especificamente utilizada para esta finalidade), aguardando o contato do Applet de Mensagens (também representado na Figura 9).

Um objeto da classe *TokenManager* é criado uma vez e armazena os dados de cada conexão com um cliente *Sticker*. Nele, ficam salvas as informações de identificação do paciente, a referência da conexão com o Sticker e informações da classe *MessageWorker* (encarregada de gerenciar a conexão com o Applet).

As classes *MessageWorker* e *MessageServer*, e as classes diretamente relacionadas a elas suportam a interação entre o Applet de Mensagens e o *HHS Sticker* (que, de certa forma passa a fazer papel de servidor). Quando o Applet de Mensagens estabelece uma conexão com a *MessageWorker* e o operador seleciona a opção de verificar o status do *Sticker*, esta classe primeiro verifica no *TokenManager* se existe algum registro de conexão na tabela referente ao paciente buscado. Se existir, o servidor irá enviar a mensagem para saber se o *Sticker* ainda está ativo ou não. Se o registro não existir na *TokenManager,* a mensagem de verificação de status não será enviada e uma exceção é lançada. Uma vez feita a verificação, o operador pode enviar uma mensagem de texto para o *HHS Sticker*.

Observa-se que esta parte do módulo Servidor também é importante na integração da aplicação com o sistema HHS. Primeiramente, apenas usuários cadastrados no banco de dados do sistema podem enviar mensagem para um *HHS Sticker* e, para isso, também precisam consultar a identificação do paciente no sistema. Além disso, as funções de agendamento e acionamento do mecanismo de envio de mensagens podem ser integradas ao sistema, sem a necessidade de um módulo separado para isso. A abordagem de se desenvolver um módulo dedicado a isso também se justifica pela independência na realização de testes de operação.

Comunicação entre os elementos

A Figura 10 detalha a comunicação entre o cliente e o servidor. O usuário (paciente) solicita uma informação através da interface do *Sticker* (1) e a página do *Sticker* que estiver ativa no momento realiza o pedido da informação desejada e se registra em uma lista de espera para aguardar a devida resposta. A informação, então, é encapsulada em um objeto *Request* e enviada ao servidor (2). Este recebe esta requisição, busca as informações no banco de dados existente (3), processa a resposta e a encapsula em um objeto *Response*, logo depois o envia ao *Sticker* (4), que sempre espera receber objetos deste tipo. Então, assim que a resposta chega até a entrada do *Sticker* (5), a página anteriormente registrada é notificada e mostra na tela o conteúdo solicitado (6).

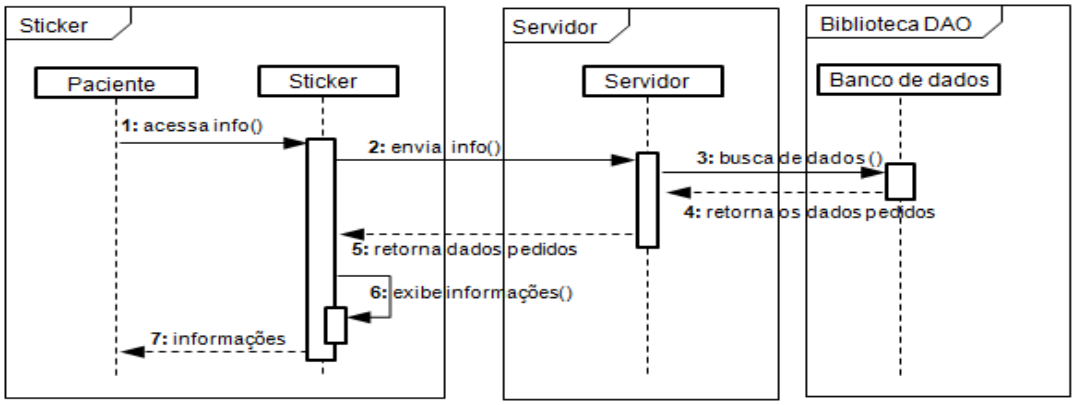


Figura 10. Comunicação Sticker-Servidor

No diagrama da Figura 11 é mostrada a comunicação da aplicação que permite o envio de mensagens e notificações para o *Sticker* (2), repassadas pelo servidor (3) invertendo os papéis cliente-servidor (4). Antes do envio da mensagem é possível verificar se o *Sticker* está em execução (8 a 12). Desta forma, é também esperado que o paciente receba a notificação e confirme que recebeu a mesma, indicando, por exemplo, que ele ou ela tem ciência de que deve realizar uma medida de pressão arterial ou ingerir um medicamento.

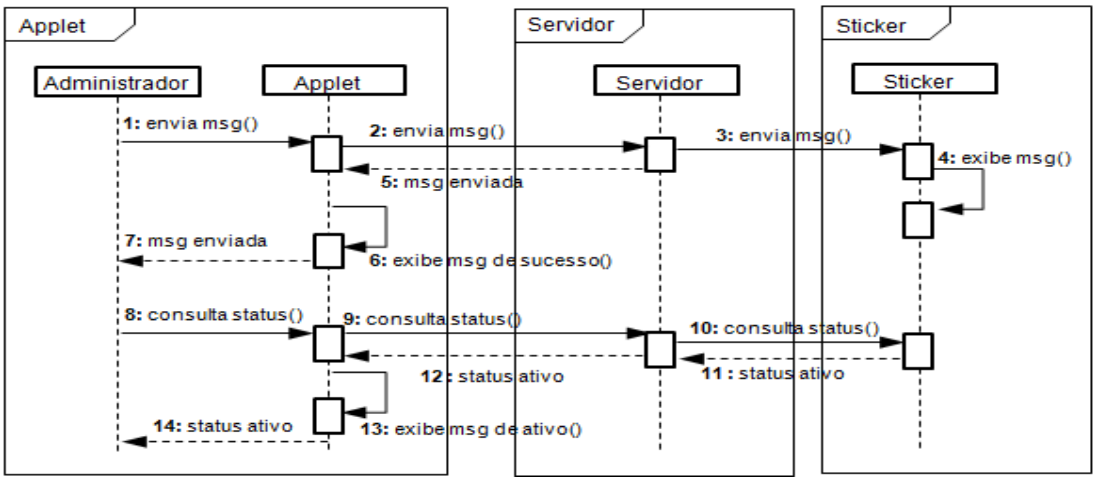


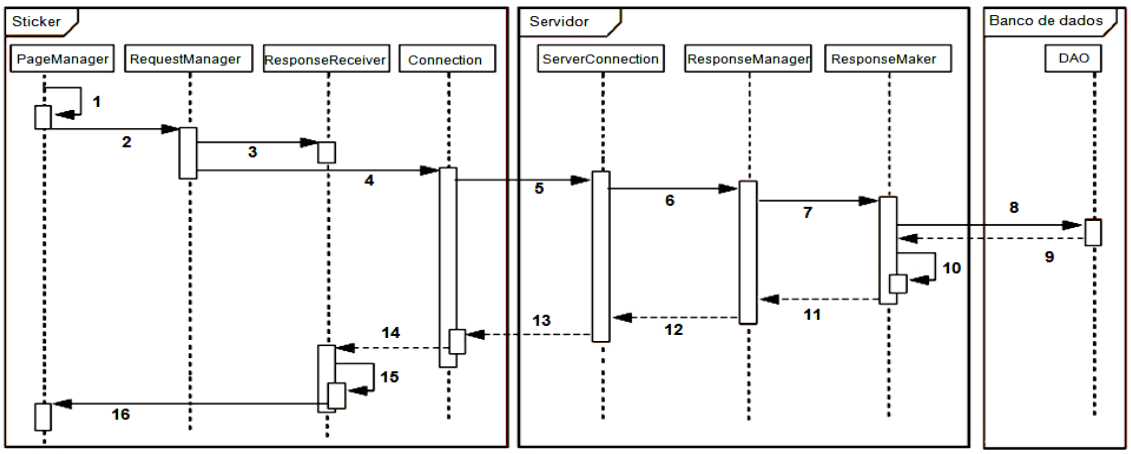
Figura 11. Comunicação Notificação e Status do Sticker

Etapas de requisição/resposta discutidas anteriormente são padronizadas, assim como, no diagrama da Figura 12, para requisição de *login*. Cada objeto que requisita uma informação ao Servidor é do tipo *PageManager* e deve implementar a interface *IResponseListener*, para que todos tenham o método *responseHasBeenReceived* (*Response response*). O Servidor está sempre à espera de um objeto do tipo *Request* e o *Sticker* está sempre à espera de um objeto do tipo *Response*. Os objetos dEsses tipos têm a indicação no próprio nome, por exemplo, um objeto de requisição de *login* se chama *LoginRequest*, assim como o objeto de resposta equivalente se chama *LoginResponse*.

Por exemplo, no primeiro contato é enviado ao Servidor uma requisição de *login* (contendo o nome de usuário e senha) e o cliente recebe como retorno um *LoginResponse*, contendo um *token* identificador daquele paciente e algumas informações sobre suas atividades, que vão preencher a primeira página do *Sticker*. Nos contatos seguintes o *token* é enviado para fazer a validação do paciente no sistema, sem a necessidade de reenvio de nome e senha, e a informação pedida só é respondida se a houver uma validação com sucesso. Seguindo a Figura 12, temos:

1. A classe *MainPage* solicita ao usuário seu nome e senha de acesso ao sistema e cria um objeto *LoginRequest* com as informações (1) e passa este para o *RequestManager* (2). Neste momento, é criado o socket de conexão com o Servidor.
2. O *RequestManager* chama o método *addRequestReference(int requestOp_code, IResponseListener listener)* do objeto *ResponseManager* para salvar a referência da classe que o chamou (3), no caso a *MainPage*, juntamente com um identificador do tipo de requisição. Depois ele chama o método *sendRequest(Request request)* da classe *Connection* para enviar a requisição ao servidor (4).
3. A classe *Connection* envia o objeto *request* pelo *socket* e aguarda a resposta (5).
4. No *Servidor*, a requisição é recebida e cria uma nova thread (*ServerConnection*) e passa o controle da conexão para a mesma, que assume o controle da comunicação com este *Sticker*.
5. Então a *ServerConnection* recebe a requisição, a transfere para o *ResponseManager* e aguarda a resposta (6).

6. *ResponseManager* identifica o tipo da requisição, delega a responsabilidade de processar a resposta à classe *ResponseMaker* e fica aguardando a resposta (7).
7. Como a requisição é de *login*, primeiramente o *ResponseMaker* usa a classe *TokenManager* para criar e armazenar um *token* de identificação do paciente, em seguida utiliza chamadas de métodos dos objetos do pacote *hhs.dao,* usado por todo o sistema HHS, para buscar informações no sistema existente (8) e com o retorno (9) ele cria um objeto de resposta, contendo o *token* (10), e retorna para quem o chamou, o *ResponseManager* (11). Quando não é uma requisição de *login*, a *TokenManager* é chamada (no *ResponseManager*) apenas para validar o usuário, buscando o objeto paciente através do *token* recebido, e na hora de retornar a resposta não é necessário enviar o *token* novamente, pois o cliente já sabe esta informação.
8. *ResponseManager* recebe o objeto de resposta e repassa para a classe *ServerConnection* (12), que envia direto para a classe Connection do *Sticker* que está esperando (13).
9. Ao receber a resposta, a Connection usa a *ResponseReceiver* (14) para notificar ao *listener* (no caso a *MainPage*) que a resposta foi recebida (15).
10. *MainPage* é notificada da resposta (16) e assim que isso acontece ela libera o acesso ao sistema para o paciente, que já pode ver as informações das suas atividades na primeira aba do *Sticker*.



1: cria um objeto de requisição( ); **9:** retorna dados;

2: faz requisição( ); **10:** cria um objeto de resposta();

3: salva referencia(); **11 e 12:** retorna resposta;

4: solicita envio da requisição(); **13:** envia resposta();

5 e 6: envia e transfere requisição(), respectivamente; **14:** repassa resposta();

7: [tokenValido=true]: solicita processamento da requisição(); **15:** busca a referência();

8: busca informações no sistema existente(); **16:** notifica e passa a resposta recebida();

Figura 12. Diagrama de interação request/response

5. Protótipo do *HHS Sticker*

A apresentação do *HHS Sticker* está dividida em três abas e uma janela (complemento das informações das abas), além das duas páginas de ajuda (auxílio das teclas do controle remoto) e copyright (autoria e versão do aplicativo). Esta apresentação segue o padrão visual recomendado para um *Sticker* (Seção 3.3). Cada aba possui conteúdo carregado dinamicamente com informações relativas ao paciente "logado".

Primeiramente, para ter acesso ao sistema, o paciente precisará fazer o *login* digitando seu nome de usuário e senha previamente cadastrados (Figura 13). Caso algum dado esteja errado, será exibida uma mensagem de erro pedindo para que ele tente novamente. Se os dados inseridos estiverem corretos, o *login* será feito com sucesso e a primeira aba aparecerá. Esta primeira aba contém as informações retiradas do *loginResponse* (a resposta de requisição de *login*), com as próximas atividades do paciente.

Figura 13. Realizando o *login*

Aba "Principal" (Figura 14). Contém informações retiradas do *loginResponse* (a resposta de requisição), aparece após a realização do *login* e contém uma caixa de notificação, além de uma lista em ordem cronológica das próximas atividades do paciente com seus respectivos detalhes: data, hora, duração e descrição.

Figura 14. Tela da Aba Principal

Aba **"Plano de Cuidados"** (Figura 15). Apresenta uma lista com todos os planos de cuidados, assim como seus respectivos detalhes: data, hora de início e término, descrição e atividades relativas.

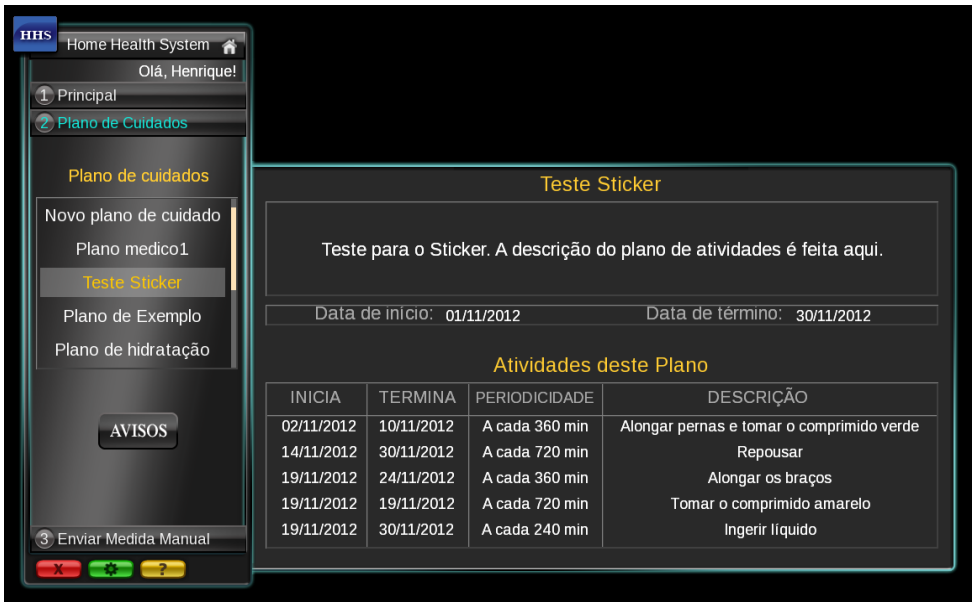


Figura 15. Tela da Aba Plano de Cuidados

Aba **"Enviar Medida Manual"**. Permite ao paciente enviar medidas aferidas manualmente. Existem duas opções de medida com janelas correspondentes conforme Figura 16: "Usar módulo de IA" – utilizada para envio de medida com análise do modo de inteligência artificial (apenas para pressão arterial) – e "Relacionar medida EVA" – utilizada para adicionar a escala de dor, juntamente com a medida. Ambas podem ser marcadas ao mesmo tempo, porém a janela ativa será equivalente a quando a opção "Usar módulo de IA" estiver marcada.

Figura 16. Opção "Usar módulo de IA" e "Relacionar medida EVA"

As "**Mensagens de Alerta"** (Figura 17) são enviadas por um administrador para informar o paciente sobre algum evento do plano de cuidados. Se o *Sticker* estiver aberto na hora do envio, a mensagem será mostrada na tela ativa, caso contrário, será armazenada com o *status*=0 e assim que o *Sticker* estiver ativo, serão apresentadas ao alterar automaticamente o *status* para 1.

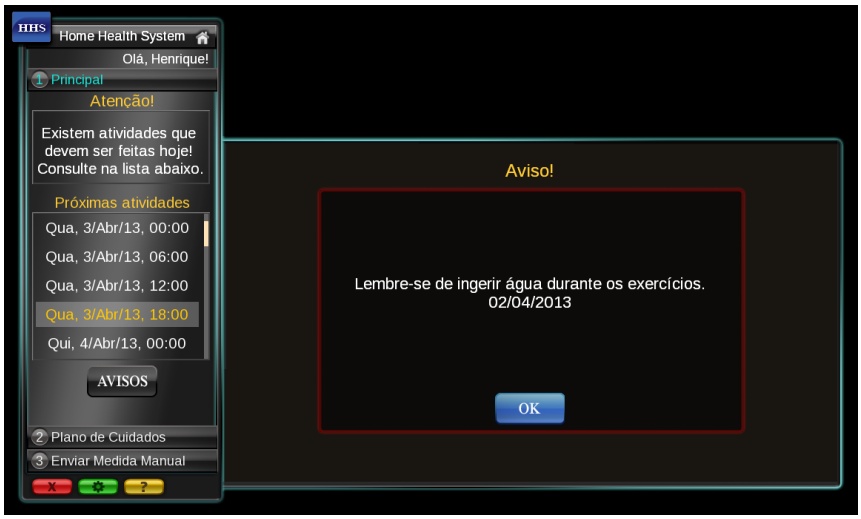


Figura 17. Sticker recebendo mensagem de alerta

5.2 O Applet de Envio de Mensagens

Para facilitar a rápida integração do sistema de envio de mensagens foi desenvolvido um módulo na forma de um Applet para que administradores, com conta criada no sistema HHS, pudessem enviar mensagens aos pacientes (Figura 18) para notificá-los de quaisquer informações inerentes ao tratamento.

A mesma interface também permite verificar se o Sticker está ativo (botão "STATUS DO STICKER") no momento. Este último serviço é disponibilizado pelo

HHS Sticker, para que a aplicação possa verificar se a TVDi está ligada, antes de enviar a notificação, ou buscar uma outra alternativa usando os serviços de contexto.

Para enviar a mensagem o administrador deverá localizar o nome do paciente, selecioná-lo em uma lista para o campo "Para", escrever a mensagem e clicar em "ENVIAR".

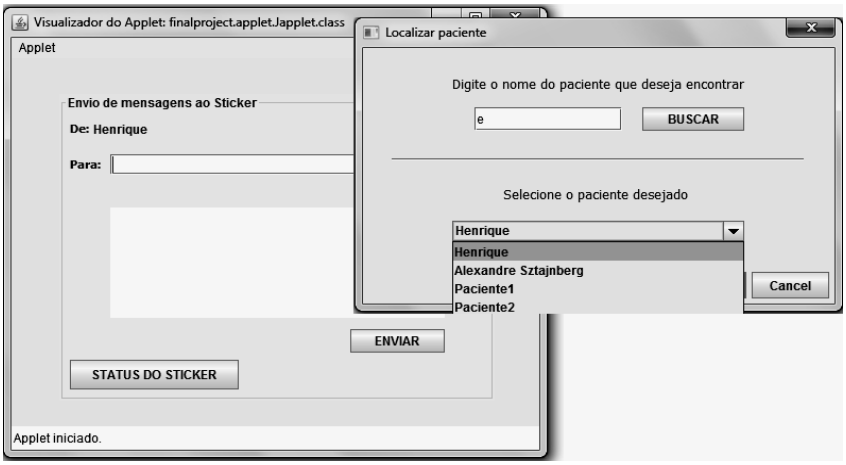


Figura 18. Tela de mensagens do applet

5.3 Detalhes de implementação

Todo desenvolvimento do *HHS Sticker* foi feito com a linguagem Java, segundo o padrão GingaJ. Conforme discutido na Seção 3.3 a classe *Main* implementa a interface *Xlet* e obtém e inicializa uma referência do objeto *Sticker,* no método *initSticker()*, utilizando o pacote *com.tqtvd.stickercenter* (Figura 19). Os métodos obrigatórios da interface *Xlet* são também implementados.

```
// Bibliotecas necessárias para o Xlet e para o Sticker
import javax.microedition.xlet.*;
import com.tqtvd.stickercenter.*;
[...]
public class Main implements Xlet{           // A implementação de Xlet
  public void initSticker(){                // A implementação do sticker
    stick = Sticker.getInstance();            // Instanciando o Sticker
    [...]
    stick.setWindowComponent(windowComponent);  //Definindo components

     mainPage = new MainPage();

     stick.setMainComponent(mainPage);

  }
  [...]
```

Figura 19. Trecho da classe *Main* (implementação do *Xlet)*

A estrutura do *Xlet*, seu ciclo de vida e a estrutura do *Sticker* obedecem a padrões de projeto estabelecidos. Entretanto, algumas soluções adotadas no código do *Sticker* e do Servidor merecem destaque, entre elas as estruturas de dados usadas para representar requisições de pacientes repassadas como padrão entre os objetos, bem como alguns aspectos de comunicação via *sockets* e seus empregos no *Xlet*.

Na Figura 20, destacam-se algumas partes da comunicação do *sticker* com o servidor, descrita na Seção 4.2, que exige a comunicação através de *sockets*. A iniciativa de requisição parte do objeto do tipo *PageManager*, ao criar um objeto de requisição (linha 1) e acionar o *RequestManager* enviando a requisição e a sua própria referência como parâmetros (linha 2). No método *requestReceiver* do *RequestManager* esta referência é registrada para encaminhar a futura resposta (linha 4) e a requisição é passada para a Connection, que a envia para o servidor (linha 5). Ao ser chamado, o método *addRequestReference* registra o código da requisição e a referência de quem espera a resposta (linha 8). O objeto da classe Connection, por sua vez recebe a requisição e a envia para o servidor através do *socket* (linha 11).

No lado do Servidor, a classe *ServerConnection* recebe a mensagem, também por um *socket*, já convertendo de Object para a classe *Request* esperado (linha 12), salva sua referência e em seguida encaminha a mesma para execução, chamando o método *executeRequest* (linha 14). No *ResponseManager*, o método *executeRequest* verifica a validade do token e o tipo da requisição (linhas 17 e 18), e aciona a rotina correspondente, que chama métodos do *ResponseMaker* para tratá-la e montar a resposta (linhas 19 a 21) A ação do *ResponseMaker* requer interação com o sistema HHS e será destacada adiante.

Uma vez obtida a resposta, o *ServerConnection* envia a resposta pelo socket e com isso, pelo lado do Servidor, a requisição está concluída (linha 24).

Novamente ao Sticker, *Connection* recebe a resposta pelo *socket* ativo (linha 25) e a passa para o *ResponseReceiver* (linha 26), que por sua vez notifica, chamando o método *notifyListener* (linha 27). No método *notifyListener* o *PageManager* que está aguardando a resposta é identificado (linha 29) e o mesmo é notificado chamando o método de *call-back listWillRecResponse.responseHasBeenReceived(rspns)* (linha 30), completando a interação.

```
// PageManager
1    Request myPlanRequest = new MyPlanRequest(3,CurrentLogin.getToken());
2    RequestManager.getReference().requestReceiver(myPlanRequest, reference);

// RequestManager
3   public void requestReceiver(Request request, IResponseListener listener){
4     ResponseReceiver.getReference().
                    addRequestReference(request.getOperationCode(),listener);
5     Connection.getReference().sendRequest(request);
6   }

// ResponseReceiver
7   public void addRequestReference(int reqOpCode, IResponseListener lstnr){
8     this.responseList.put(new Integer(reqOpCode), listener);
9   }

// Connection
10  public void sendRequest(Request request){
11    out.writeObject(request);

// ServerConnection
12   objectRequest = (Request) in.readObject();
13   objectResponse = ResponseManager.getReference().
14   executeRequest(objectRequest);
```

```
// ResponseManager
15  public Response executeRequest(Request request){
16    Response response = null;
17    if (TokenManager.getReference().checkUser(request.getToken()){
18       switch(request.getOperationCode()) {
      […]
19         case 3: {
20           response = ResponseMaker.getReference().doMyPlanSelect(
                    TokenManager.getReference().getUser(request.getToken()));
21           response.setOperationCode(request.getOperationCode());
22      }
        [...]
23    return response;}

// ServerConnection
24    if (objectResponse != null) out.writeObject(objectResponse);

// Connection
25    Response objectResponse = (Response) in.readObject();
26    ResponseReceiver.getReference().responseReceiver(objectResponse);

// ResponseReceiver
27  public void responseReceiver(Response rspns){ notifyListener(rspns);}

28  public void notifyListener(Response rspns){
29    IResponseListener listWillRecResponse = (IResponseListener)
                  responseList.get (new Integer(rspns.getOperationCode()));
30    listWillRecResponse.responseHasBeenReceived(rspns);
```

Figura 20. Trecho de código (cliente-servidor) do processo de requisição e resposta

Para discutir a interação do Servidor com o sistema HHS, no *ResponseMaker*, e seguindo o exemplo de requisição já utilizado, o acesso ao banco é requisitado para a obtenção do Plano de Cuidados (método *doMyPalnSelect*), conforme Figura 21. Como o paciente já está identificado no sistema, a integração de chamadas de métodos às bibliotecas DAO é direta. A referência ao paciente é passada ao método pela variável *currentPatient* (linha 1).

```
1   public MyPlanResponse doMyPlanSelect(Paciente currentPatient){

2     myTreatment = new TratamentoDao().
                                    pegarTratamentos(currentPatient);
3     myActivities = new AtividadeDao().
                       pegarAtividadesTratamento(myTreatment.get(j));
4     myActivities.set(i, TransformToActivity.makeTransformation(
                      myActivities.get(i)));
5     myPlanActivities.put(
                     new Integer(((Tratamento)myTreatment.get(j)).
                                    getIdTratamento()), myActivities);
6     myTreatment.set(i, TransformToTreatment.makeTransformation(
            (Tratamento)myTreatment.get(i)));
7     myPlanResponse = new MyPlanResponse
                                    (myTreatment, myPlanActivities);
8     return myPlanResponse;
    }
```

Figura 21. Trecho de código referente à execução da requisição do cliente

No caso da obtenção do Plano de Cuidados do HHS, contendo as rotinas prescritas pelo médico para o paciente, as estruturas de objetos são listas varridas iterativamente (os laços de iteração foram omitidos por simplicidade). Primeiro a referência ao tratamento é obtida (linha 2) e em seguida para cada tratamento as atividades são também obtidas (linha 3). Antes de retornar as informações, os objetos são "transformados" para simplificar os mesmos para atender a compatibilidade da versão 1.3 do Java executando no *set-top box* (linhas 4 e 6). Em seguida um resultado é criado com as listas de tratamento e atividades (linha 7), e finalmente este resultado é retornado ao chamador (linha 8).

6. Trabalhos relacionados

O livro *Digital Health Information for the Consumer: Evidence and Policy Implications* [Nicholas, 2007] avalia várias tecnologias para e-Health, dedicando o Capítulo 5 à TV Digital Interativa. Foram avaliados vários aspectos da tecnologia, incluindo o impacto para o paciente, para o médico e para os serviços de saúde já estabelecidos. Quatro consórcios apresentaram projetos piloto para esta avaliação: Communicopia, FlextechTelewest, Living Health e dktv (A Different Kind of Television). Basicamente o serviço oferecido nos quatro projetos era o acesso a informações específicas de áreas da saúde. A interatividade não foi usada como no *HHS Sticker*, onde existe interatividade do paciente com o médico e a equipe de monitoramento.

Uma proposta de uso da TV digital com interatividade para facilitar a comunicação paciente-médico é apresentada em [Niiranen, 2002], desenvolvido para o mercado finlandês. Nessa proposta, dados do paciente são salvos em arquivos XSL (*Extensible Stylesheet Language*) e apresentados na televisão em formato XML (*eXtensible Markup Language*). Adicionalmente existe um servidor de páginas WEB que recebe, via método POST, as medições realizadas e junto com outros dados do paciente através de uma comunicação cliente-servidor TCP/IP sobre PPP (Point-to-Point Protocol). A proposta ainda prevê a integração entre dispositivos de medição, usados nos tratamentos residenciais, e a TV digital, no envio dos dados. No *HHS Sticker*, a integração de dispositivos sem fio foi avaliada, mas dependeria de acionadores e código de terceiros que não estavam disponíveis.

O projeto PANACEIA-iTV [Prentza, 2005], na mesma linha adotada para o *HHS Sticker*, é um sistema que incentiva o paciente a monitorar sua saúde e acessar informações e suporte à saúde. O sistema foi testado com pacientes portadores de *Adult Congenital Heart Disease* (ACHD) grave no *Royal Brompton Hospital*, sendo os mesmos, monitorados via satélite, em suas residências, utilizando a tecnologia DVB-S. Um experimento com o sistema é relatado em [Karagiannis, 2006], envolvendo 9 pacientes hospitalizados e 12 pacientes não especializados. Após um rápido treinamento, os pacientes foram orientados a realizar medidas de pressão arterial, ECG, SpO2 e peso, e enviá-las através do sistema. A avaliação mostrou que o uso do sistema foi positivo para a maioria dos pacientes.

O projeto *MHPHomecare* [Angius, 2008] tem elementos comuns com o *HHS Sticker*. O sistema utiliza o padrão DVB-T (Digital Video Broadcasting – Terrestrial) para transporte de mídias no formato MPEG e executa uma aplicação Java, também estruturada a partir de um *Xlet* em um *set-top box* compatível. A aplicação desenvolvida

tem o objetivo de receber medidas fisiológicas através da interface de infravermelho do *set-top box*. Para isso foi desenvolvido um hardware externo microcontrolado que realiza a captura dos sinais biológicos de interesse e os transmite para o *set-top box* via interface de infravermelho. Em seguida o *Xlet* transmite os dados para um centro de cuidados remoto por uma conexão à Internet. O *Xlet* também provê acesso gráfico a outros elementos do sistema. Duas outras características devem ser mencionadas. O projeto utiliza um *smartcard* com informações personalizadas para o paciente, que facilita a configuração de parâmetros específicos, inserido na leitora do *set-top box*, lida pelo Xlet. Outra característica é o envio da aplicação para o *set-top box* pelo canal de difusão de *sofware* do padrão DVB-T. Com isso a atualização do software, quando necessária, é automática. Não há necessidade de download da aplicação de uma página web, como no *HHS Sticker*. Por outro lado, o envio de aplicações por um canal de difusão requer uma infraestrutura relativamente complexa e cara [RCA].

O projeto Med-Reminder [Stojmenova, 2013] utiliza a TVDigital para enviar notificações e lembretes. A preocupação com a interface gráfica, considerando que o público-alvo seria a população de 3ª idade, além da garantia de recebimento e leitura da notificação enviada pelo administrador (médico) é semelhante às funções do *Applet* de Mensagens do HHS *Sticker*. No entanto, o projeto [Stojmenova, 2013] apresenta problemas relativos à dificuldade em inserir lembretes usando o controle remoto, pela quantidade de informação necessária durante a criação. No *HHS Sticker* os lembretes são sempre enviados pelo médico, monitor ou pelo sistema de Plano de Cuidados e não pelo paciente. O paciente só precisa entrar com poucas informações, basicamente as medidas fisiológicas.

Em [Oliveira, 2009], os autores apresentam a proposta de uma arquitetura para monitoramento de pacientes utilizando o padrão Ginga, chamado DIGA Saúde TV, elencando características importantes para este tipo de sistema, similares ao HHS e ao *HHS Sticker*. Entretanto, não apresentam um protótipo consistente, nem realizam simulações com o sistema de comunicação ou de integração de elementos.

Entre as propostas relacionadas, Motiva da Philips é uma das mais abrangentes. Motiva é uma proposta da A Philips [Philips, 2013] para um sistema de telessaúde fortemente apoiado por TVDi. Entre os objetivos do Motiva estão um mecanismo de informação e lembretes baseados em regras, integrando suporte prático para o paciente aderir ao seu plano de cuidados. Através de aplicações para TV Digital e serviços de suporte, o sistema oferece informações educativas, sistema de mensagens personalizado, questionários de motivação e o envio de dados fisiológicos monitorados segundo recomendação médica. Os aspectos técnicos da implementação das aplicações de vídeo não são discutidos. Mas, diferentemente dos trabalhos com foco na implementação, como o *HHS Sticker*, a Philips oferece incentivo para a realização de testes clínicos. Destacam-se dois artigos do mesmo grupo, ambos [Domingo, 2013] e [Domigos, 2013], realizados como parte do CARME (*Catalan Remote Managenet Evaluation Study*), no Instituto Catalão de Saúde. Os dois testes realizados durante um ano dentro de padrões de rigor e ética, envolveram mais de 300 pacientes, com doença cardíaca. Os autores discutem todos os detalhes de seus resultados, como: a verificação da diminuição de reinternações, aspecto bastante positivo; a interação simples dos pacientes com o sistema; e até mesmo a diminuição da aderência às rotinas de envio de medidas fisiológicas através do sistema.

7. Conclusão

A arquitetura do *HHS Sticker* foi desenvolvida para ser integrada a um conjunto de módulos de clientes de contexto do sistema HHS (*Home Health Subsystem*). Além de permitir a exibição de conteúdo do acompanhamento médico dos pacientes e a transmissão de medidas fisiológicas, ele atua como sensor, fornecendo a informação de seu estado (ligado ou desligado), bem como qualquer outra informação que possa ser obtida dentro do ambiente GingaJ. Num cenário típico de uso do HHS é possível identificar se o paciente está próximo ou localizado no mesmo cômodo em que a TV se encontra e se a mesma está ligada (e o *HHS Sticker* devidamente conectado ao servidor). Assim, se o sistema ou o médico pretende enviar uma mensagem a este paciente neste instante, ele pode decidir fazer isso por meio da TV (através do Sticker), pois a probabilidade que ela seja vista é muito grande.

Um das características da aplicação é a simplicidade no manuseio e na integração com o *HHS Applet*, somados ao visual atrativo.

O fator custo também pode ser considerado. Segundo dados IBGE 2011, apenas 4,42% da população acima de 60 anos têm computador com qualquer tipo de acesso à Internet em sua residência. Por outro lado existem metas governamentais para desativação da televisão analógica, com planos de incentivo à aquisição de TV digital por parte da população. Ainda, entre as metas, a indústria está sendo incentivada a implantar canais de retorno e interagir através de infraestrutura que utilize o espectro de frequência VHF brasileiro (54 a 88Mhz e 174 a 216Mhz). O uso do canal de retorno via VHF tornará possível o uso de aplicações como o HHS Sticker até mesmo para áreas rurais, distantes e carentes da população.

O trabalho de integração *HHS Sticker* ao sistema HHS está em desenvolvimento. A interface para envio de notificações ao paciente, desenvolvida na forma do *Applet* de Mensagens será incorporada à interface com os médicos, bem como integrada ao sistema de gerenciamento do Plano de Cuidados. Outro ponto de aprimoramento é relacionado ao aspecto visual do HHS Sticker. Uma avaliação de usabilidade, com pacientes idosos, permitirá verificar que aspectos devam ser melhorados, desde a quantidade de informações apresentadas em cada aba, até o tamanho das letras exibidas.

Agradecimentos. Agradecemos à Faperj pelo apoio financeiro na aquisição dos aparelhos set-top box da TQTVD. Agradecemos à TQTVD pelo apoio técnico na elaboração deste projeto.

Referências

Angius G, Pani D, Raffo L, Randaccio P, Seruis S. "A tele-home care system exploiting the DVB-T technology and MHP", Methods Inf Med, Schattauer GmbH, Vol. 47, No. 3, pp. 223-228, 2008; doi:10.3414/ME9114

AstroBox. TOTVS, 2011. https://www.astrodevnet.com/AstroDevNet/ restrict/ astrobox_sobre.html, [Acesso em: Jan/2013].

AstroTV, TOTVS 2012. http://www.tqtvd.com/novo/br/astrotv-interna.html#tv1 [Acesso em: 07/2013]

Batista, Carlos Eduardo. “TV Digital - Java na sala de estar”. 2007. Revista Mundo Java, número 17, ano III – Editora Mundo.

Domingo M., Lupón J, González B, Crespo E, López R, Ramos A, et al. “Noninvasive remote telemonitoring for stable patients with heart failure: effect on number of hospitalizations, days in hospital, and quality of life”. CARME (CAtalan Remote Management Evaluation) Study. Rev Esp Cardiol, in press, doi:10.1016/j.recesp.2010.10.032.

Domingo M., Lupón J, González B, Crespo E, López R, Ramos A, et al, “Evaluation of a telemedicine system for heart failure patients: Feasibility, acceptance rate, satisfaction and changes in patient behavior”. Results from the CARME (CAtalan RemoteManagement Evaluation) study European Journal of Cardiovascular Nursing (2011), doi:10.1016/j.ejcnurse.2011.02.003

D-Link, RECEPTOR D-LINK (2011). http://www.dlink.com.br/produtos-detalhes/items/dtb-331.html [Acesso: 07/2013]

Enc, 2009. Encerramento TV Analógica. http://www.brasil.gov.br/sobre/ciencia-e-tecnologia/industria-eletronica-digital/tv-digital, [Acesso: Fev/2014]

Esp, 2014. Especificação Ginga. http://gingadf.com.br/blogGinga/ javaDtv Api/ javaDoc_V1.3, [Acesso: Fev/2014]

Ginga, 2013. Sobre o Ginga. http://www.ginga.org.br/pt-br [Acesso: 07/2013]

Grupos de Trabalho na SBTVD, 2011. http://sbtvdbcc.wordpress.com/2011/10/05/ grupos-de-trabalho-na-sbtvd/ [Acesso em: Abr/2013].

[HIST 2013]. ”História da TV Digital no Brasil”. http://www.dtv.org.br/informacoes-tecnicas/historia-da-tv-digital-no-brasil/ [Acesso: 07/2013]

IBGE, 2011. “Pesquisa Nacional por Amostra de Domicílios 21/09/12” http://www.ibge.gov.br/home/presidencia/noticias/imprensa/ppts/00000010135709212012572220530659.pdf [Acesso: 07/2013]

IPTV, 2013. http://www.ip.tv/iptv_site/ptb/html/client.html [Acesso: 07/2013]

ITU, 2013. ITU-T in brief. http://www.itu.int/en/ITU-T/about/Pages/default.aspx [Acesso: 07/2013]

Karagiannis, George E., Stamatopoulos, Vasileios G., George Roussos, Takis Kotis, Michael A Gatzoulis, "Health and lifestyle management via interactive TV in patients with severe chronic cardiovascular diseases" *J Telemed Telecare July 1, 2006 Vol 12*: pp. *17*-19,Telemed Telecare July 1, 2006 vol. 12 no. suppl 1 17-19 SAGE Journals, , UK. doi: 10.1258/135763306777978489J

Kulesza, R. “API JavaTV-Criando e Controlando Aplicações Java no Ginga”, 2009. http://www.tvdi.inf.br/site/artigos/Ginga-J/Desenvolvimento Ginga-J, JavaDTV, OpenGinga - Parte 2 – KULESZA, FERREIRA.pdf [Acesso: 07/2013]

Kulesza, R. e Ferreira, J. “Desenvolvimento Ginga-J – JavaDTV – OpenGinga”, 27/06/09. http://www.tvdi.inf.br/site/artigos/Ginga-J/Desenvolvimento Ginga-J, JavaDTV, OpenGinga - Parte 1 – KULESZA, FERREIRA.pdf [Acesso: 07/2013]

Macedo, E. L. C. ; Ferreira, D. B. ; Lemos, G. M. R. ; Sztajnberg, A. ; Loques, Orlando Gomes. “Suporte para Coleta e Persistência de Dados de Contexto em um Sistema de Monitoramento Domiciliar Remoto de Pacientes”. In: Computer on the Beach 2011, 2011, Florianópolis. Proceedings of the Computer on the Beach 2011, 2011.

Midlet Profile”. http://docs.oracle.com/javame/config/cldc/ref-impl/ midp2.0/jsr118/javax/microedition/midlet/MIDlet.html[Acesso: 07/2013]

Morris, S. “An Introduction To Xlets” [Figura: The state diagram for an Xlet] http://www.interactivetvweb.org/tutorials/javatv/xlet_intro [Acesso: 07/2013]

Nicholas, D., Huntington P., Jamali H., "Digital Health Information for the Consumer: Evidence and Policy Implications", Chapter 5 - Health Digital Interactive Television (DiTV), Ashgate Publishing, Ltd., England, 2007.

Niiranen, S., Lamminen, H., Mattila, H., Niemi, K., Kalli, S. Personal health care services through digital television – Computer Methods and Programs in Biomedicine 68 (2002) 249-259 - 25/01/2002

Oliveira, M.; Cunha, P.R.F.; da Silva Santos, M.E.; Bezerra, J.C.C. "Implementing home care application in Brazilian Digital TV", Global *Information Infrastructure Symposium, GIIS '09*, pp. 1 - 7, 2009. 10.1109/GIIS.2009.5307042

Philips HealthCare, "Philips Motiva". http://www.healthcare.philips.com/ br_pt/ products/ telehealth/ products/ motiva.wpd [Acesso: Fev/2014]

Prentza A., Stavroula Maglavera, George Stalidis, Eleni Sakka, Irini Lekka, Pantelis A. Angelidis, Lefteris Leondaridis, Nicos Maglaveras, and Dimitris Koutsouris, “Cost-effective health services for interactive lifestyle management: the PANACEIA-iTV and the e-Vital concepts” Journal of Telecommunications and Information Technology, April 2005: pp. 49-58. ISSN 1509-4553

RCA, "EITV Playout Professional", http://www.rcasoft.com.br/playout.php [Acesso em: Fev/2014]

StickerCenter da TOTVS. https://www.stickercenter.com.br/StickerWeb/pt_BR/ index.html [Acesso em: Abr/2013].

Stojmenova, E., Debevc, M., Zebec, L., Imperl, B., Cunha, P. Assisted living solutions for the elderly through interactive TV - Multimed Tools Appl (2013) 66:115–129 - http://scienceindex.com/

Sztajnberg, A., Rodrigues, A. L. B., Bezerra, L. N., Loques, O. G., Copetti, A., & Carvalho, S. (2009). “Applying context-aware techniques to design remote assisted living applications. International Journal of Functional Informatics and Personalized Medicine”, 2(4), 358-378.

The World Bank Group, "Brazil’s digital TV system delivers health, education and jobs at the touch of a remote control", FEATURE STORY, May 24, 2013. http://www.worldbank.org/en/news/feature/2013/05/22/Brazil-digital-TV-system-4D-benefits-poor, [Acesso: Jul/2013]

TQTVD. TQTVD (Sobre), 2012. http://www.tqtvd.com/novo/br/sobre.html [Acesso: 07/2013]

Ubuntu. “O que é o Ubuntu?”, 2010. http://www.ubuntu-br.org/ [Acesso: 07/2013]

V-Box, 2013. Virtual Box. https://www.virtualbox.org/ [Acesso: 07/2013]

World Bank. 2013. *Brasil 4D : Estudo de impacto socioeconômico sobre a TV digital pública interativa*. Washington DC; World Bank. http://documents. Worldbank .org/curated/en/2013/08/18203867/brazil-4d-study-socioeconomic-impact-digital-tv-interactive-public-brasil-4d-estudo-de-impacto-socioecon%C3%B4mico-sobre-tv-digital-p%C3%BAblica-interativa [Acesso: Out/2013]

Xlet, JSR 217 (Maintenance Release). http://docs.oracle.com/javame/config/cdc/ref-impl/pbp1.1.2/jsr217/javax/microedition/xlet/package-summary.html [Acesso em: Out/2012].