Serviço de Recomendação de Publicidade Personalizada (SRPP)
(Web-like Advertising for Digital TV)

Detalhamento de Projeto

Histórico de Alterações deste Documento		
Versão	**Autor(es)**	**Data**
0.01	Marco Cristo	08/jan/2010
0.02	Marco Cristo e Ângelo Bitar	09/jan/2010
0.03	Marco Cristo, Ângelo Bitar e Bruno Guimarães	12/jan/2010
0.10	Marco Cristo	19/jan/2010
0.11	Marco Cristo	20/jan/2010
0.12	Marco Cristo	26/jan/2010
0.13	Marco Cristo	28/jan/2010
0.14	Marco Cristo	02/fev/2010
0.15	Marco Cristo	15/fev/2010
0.16	Marco Cristo	16/fev/2010
0.17	Marco Cristo	19/mar/2010
0.18	Marco Cristo	25/mar/2010
0.19	Marco Cristo	03/abr/2010
0.20	Marco Cristo	17/abr/2010
0.21	Marco Cristo	27/abr/2010
0.22	Marco Cristo	29/abr/2010
0.23	Bruno Guimarães	30/abr/2010
0.24	Bruno Guimarães e Marco Cristo	3/mai/2010
0.25	Ângelo Bitar e Marco Cristo	6/mai/2010
0.26	Ângelo Bitar e Marco Cristo	14/mai/2010
0.27	Ângelo Bitar e Marco Cristo	23/jun/2010
0.28	Ângelo Bitar e Marco Cristo	31/jul/2010
0.29	Ângelo Bitar e Raiza Hanada	20/ago/2010
0.30	Ângelo Bitar e Marco Cristo	21/ago/2010

1. Introdução

Sistemas de publicidade operados em TV são caracterizados por três atores principais: (1) o telespectador da TV Digital, ou seja, seu usuário; (2) o provedor de serviço, ou seja, a entidade responsável pela distribuição da programação de TV e publicidade para os usuários; e (3) o anunciante responsável pelas propagandas a serem exibidas junto às mídias.

Tais sistemas são, em geral, caracterizados por (a) ambiente de entretenimento passivo de difícil mensuração de resultados; (b) veiculação de anúncios no intervalo da programação; (c) provedor de serviço tem papel preponderante na negociação de publicidade; (d) direcionamento de publicidade é contextual (obtido genericamente através de horários e conteúdo da programação), mas não personalizado; (e) negociação de publicidade é feita de forma contratual, não dinâmica, com garantias de exclusividade. Tais aspectos limitam direcionamento e personalização; (f) anunciante paga por audiência.

Ao contrário, em sistemas de publicidade na Web, se tira grande proveito de características como direcionamento, interatividade e ambiente colaborativo. De forma geral, tais sistemas são caracterizados por: (a) ambiente ativo de fácil mensuração de resultados; (b) veiculação de anúncios no conteúdo, de forma integrada; (c) papéis dos três atores são balanceados com o intuito de alcançar a satisfação de todos eles; (d) direcionamento de publicidade é calculado considerando interesses de curto e longo prazo do usuário (alto grau de personalização), bem como um rico contexto (características do conteúdo, localização etc); (e) múltiplos anunciantes

competem por espaço de publicidade dinamicamente, de forma não exclusiva; (f) anunciante paga por interação do usuário com publicidade.

Com a possibilidade de interação oferecida pelo ambiente de TV digital, é possível o emprego de sistemas de publicidade similares àqueles normalmente operados na Web, com algumas poucas adaptações.

Em outras palavras, ao contrário de sistemas tradicionais de publicidade em TV, o cálculo da propaganda mais apropriada para o telespectador pode ser feito com base na conjugação dos seus interesses, dos interesses do anunciante e do provedor de serviço, em tempo de exibição da programação. Informações sobre os usuários e propagandas podem ser transmitidas usando o canal de retorno. As propagandas podem ser sincronizadas com a programação no Set-top Box (STB) do usuário de forma a permitir máximo direcionamento e personalização. Como nos sistemas operados na Web, as propagandas podem ser obtidas por meio de um sistema de leilão.

Note que em tal sistema, o papel do provedor de serviço pode ser tanto o de uma empresa associada a um canal de TV (modelo mais tradicional, com grandes emissoras abertas como a Rede Globo), uma distribuidora de vários canais (modelo de TV fechada a cabo ou por satélite, com distribuidoras como a SKY) ou um mediador que funciona quase como uma agência de publicidade (modelo mais próximo ao da Web, com mediadores como o Google e Yahoo).

Se considerarmos a necessidade de minimizar custos de transmissão, sem deixar de tirar proveito de informação colaborativa de comunidades de anunciantes e telespectadores, bem como a possibilidade de fornecer um sistema de leilão de propagandas, alguns requisitos importantes de um sistema de publicidade, como o proposto, em um ambiente de TVD seriam:

1) O pré-processamento de informações e inserção da publicidade localmente *no STB*. Isto minimizaria custos de transmissão e evitaria excesso de processamento centralizado;
2) O cálculo centralizado de recomendação de publicidade. Isso possibilitaria o uso de uma estratégia de recomendação colaborativa;
3) A disponibilização de infra-estrutura externa à TV para cadastro de anunciantes e propagandas. Isso facilitaria a operação de um sistema de leilão;

Considerando tais requisitos, a arquitetura geral de um sistema como o descrito é dado na **Figura 1**.

Como observado na **Figura 1**, este sistema seria composto por três módulos, a saber, o Módulo do Anunciante, o Módulo do Provedor de Acesso e o Módulo do Usuário. O Módulo do Anunciante seria responsável pela coleta de seus interesses e propagandas. O Módulo do Usuário seria responsável pela identificação do usuário, recepção e exibição de propagandas. Finalmente, o Módulo do Provedor de Serviços seria responsável por agregar informações dos anunciantes, usuários e conteúdo para determinar as melhores propagandas. Destes módulos, o Módulo do Usuário iria residir na TV digital.

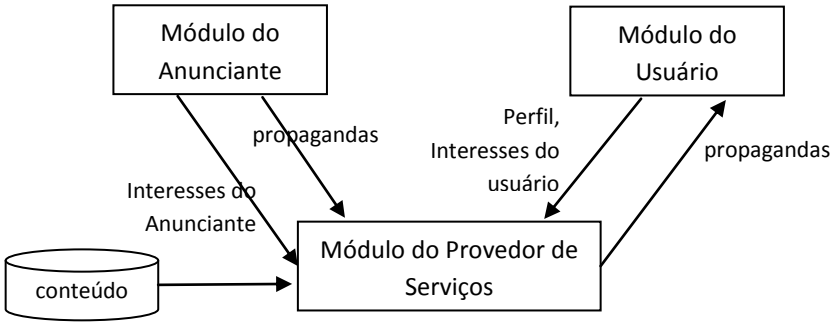


Figura 1: Arquitetura Geral de um Sistema de Publicidade Personalizada pra TV Digital, inspirado em sistemas similares em operação na Web

2. Objetivos do Projeto

Este projeto consiste no desenvolvimento de um Serviço de Recomendação de Publicidade Personalizada (SRPP) para o Sistema Brasileiro de TV digital (SBTVD), utilizando a plataforma padrão Ginga-NCL estendida com a inclusão de um módulo de mineração de dados (Módulo Recommender). *O SRPP corresponde ao Módulo Usuário ilustrado na* **Figura 1.**

Para o desenvolvimento do SRPP, é necessária a disponibilização de um serviço de identificação de características do usuário. Dada a natureza de um ambiente de TVD, entretanto, não se espera que usuários se identifiquem voluntariamente. Assim, o terminal de TV Digital deverá ser capaz de estimar qual o usuário atual e suas características. Como atualmente a plataforma Ginga-NCL *não* oferece um serviço de identificação implícita, um segundo objetivo deste projeto consiste no desenvolvimento deste serviço.

Mesmo que os usuários não costumem se identificar voluntariamente, se o fizerem o SRPP deve ser capaz de capturar e utilizar os dados informados. Assim, um terceiro objetivo deste projeto foi à criação de um serviço para identificar explicitamente o usuário, facilitando a coleta de dados demográficos como idade, sexo e nome.

Nas próximas seções, os módulos e componentes desenvolvidos são descritos em detalhes.

3. Serviço de Recomendação de Publicidade Personalizada

Nesta seção, descrevemos o Gerenciador de Contexto Implícito, o Módulo de Publicidade Local e o Módulo de Identificação Explicita do Usuário de TV Digital. O Gerenciador de Contexto Implícito é o componente que desenvolvemos no Ginga para a identificação implícita de usuários. O Módulo de Publicidade Local corresponde à implementação do Serviço de Recomendação de Publicidade Personalizada (SRPP), residente no terminal de acesso da TV digital. Finalmente, o Módulo de Identificação Explicita fornece ao usuário a facilidade de identificação voluntaria.

3.1. Gerenciador de Contexto Implícito

3.1.1. Introdução

O Ginga-NCL consiste de uma plataforma de software dividida em três camadas: (1) a camada de apresentação, responsável pela execução das aplicações interativas escritas usando as linguagens NCL e Lua; (2) o núcleo comum, responsável por prover recursos de troca de serviço, gerenciamento de perfil, recepção e processamento de vídeo, entre outras funções; e (3) a camada do sistema operacional que provê gerenciamento de memória e serviços de rede, entre outros.

Várias extensões têm sido propostas ao Ginga-NCL com o intuito de oferecer serviços adicionais. Por exemplo, a necessidade de desenvolvimento de sistemas sofisticados de análise de dados em TV, tais como sistemas de recomendação, estimularam o desenvolvimento de um módulo de mineração de dados. Este módulo é chamado de Módulo de Recomendação (MR). O MR oferece facilidades de escalonamento de processos, persistência de dados, acesso a informação contextual e algoritmos de mineração de dados.

Nesta seção, iremos descrever uma nova extensão ao Ginga-NCL: o provimento de um serviço para estimar características do usuário atual. Esta extensão é necessária porque o Ginga-NCL provê apenas suporte à identificação explícita de usuário. Isto se dá através do serviço de gerência de contextos do núcleo comum do Ginga-NCL.

O gerente de contextos mantém registro de contas e preferências dos usuários se estes se autenticam voluntariamente. Caso isto não ocorra, o Ginga considera que um usuário genérico anônimo (*default*) está autenticado e preferências são coletadas para este usuário. Como resultado, informações coletadas por aplicações na TV são mais sobre o padrão de uso da TV em si que sobre seus usuários em particular.

A disponibilização de um serviço de descoberta do usuário deve minimizar este problema, possibilitando maior nível de personalização de serviços.

Na próxima seção, descrevemos o gerenciador de contexto implícito, que estende o Módulo de Recomendação e, assim, dota o núcleo comum do Ginga-NCL da capacidade de determinar o usuário corrente.

3.1.2. Descrição

A **Figura 2** apresenta uma descrição geral da arquitetura do módulo de recomendação e suas interações com os componentes do núcleo comum do Ginga.

Como podemos ver na **Figura 2**, o módulo de recomendação é formado por seis componentes: Scheduler Agent (SA), Implicit Context Manager (ICM), Mining Agent (MA), Local Agent (LA), Filter Agent (FA) e Data Agent (DA).

O SA é responsável por disparar processos de mineração (MA) e identificação implícita de usuário (ICM). Ele se comunica com o Context Manager para determinar se há recursos disponíveis para a execução das escalas.

O MA encapsula os algoritmos de mineração. Informações relativas às interações do usuário com o STB e o tempo que ele permanece em determinado serviço são capturadas pelo LA. O FA atua sobre os resultados do MA, filtrando-os de acordo com a sua relevância. Ele ainda fornece interfaces de acesso ao histórico de interações do usuário e às informações das tabelas de meta-dados de serviços e programação (EIT e SDT). O DA gerencia a inserção e remoção de recomendações em uma base de dados SQL.

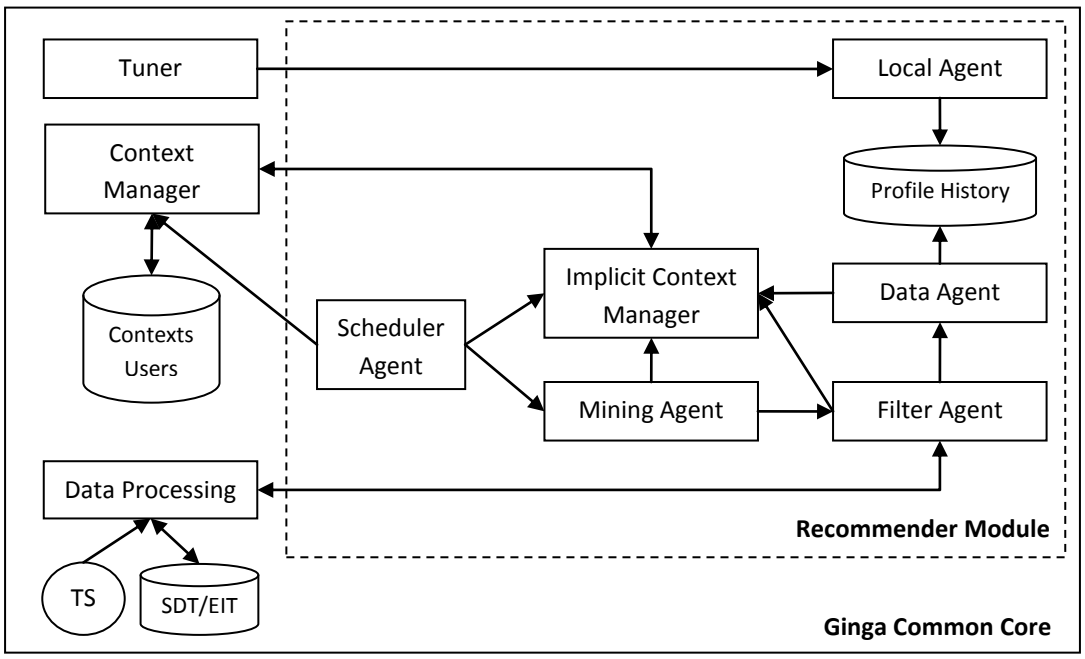


Figura 2: Arquitetura do Módulo de Recomendação e suas interações com os componentes do núcleo comum do Ginga

Finalmente, o último componente é o ICM. Este componente não pertencia originalmente ao módulo de recomendação e está sendo incorporado neste projeto[1].

O ICM é disparado periodicamente pelo SA. Como o término de um programa e o início de outro marca tipicamente um momento em que há mudança de usuário na TV[2], sempre que o ICM entra em execução, ele verifica se houve mudança de programa. Caso tenha havido, o ICM inicia o processo de identificação de usuário implícito. Para tanto, do MA, o ICM utiliza algoritmos de agrupamento[3]. Ele obtém informação sobre programação e programas vistos pelos usuários dos componentes FA e DA[4]. Finalmente, o ICM interage muito com o Context

[1] De fato, faz mais sentido pensar no ICM como um componente do núcleo comum que usa serviços do módulo de recomendação que como componente do módulo de recomendação.
[2] Esta informação foi obtida mediante análise de bases de dados, cedidas gentilmente pelo IBOPE à UFSCar, que registram o comportamento de usuários vendo TV.
[3] Esta funcionalidade foi incorporada ao MA como parte deste projeto.
[4] As classes que vão servir como interface para estas informações estão implementadas (TVData e DataInterface). Entretanto, os dados realmente usados no protótipo são obtidos de arquivos conveniente preparados para simular a sua transmissão ao longo do tempo, como seria observado em uma TV real que decodifica a informação de programação diretamente do TS.

Manager para obter informações sobre usuários autenticados bem como manter informações sobre usuários implícitos, incluindo os perfis que os caracterizam[5].

Note que ao interagir com o Context Manager, as informações mantidas pelo ICM são naturalmente disponibilizadas para o contexto de apresentação e, portanto, para aplicações NCL e Lua em execução[6]. Da mesma forma, o mecanismo de notificação do gerente de contexto pode ser usado para notificar mudanças de usuários implícitos.

O elemento essencial do ICM é o algoritmo usado para estimar o usuário implícito, descrito na **Figura 3**.

Sempre que chamado, independente de observar uma transição de programação, o ICM coleta estatísticas do conteúdo transmitido. Nesta implementação, em particular, cada usuário é representado pelo tempo que ele assiste cada um dos vários sub-gêneros de programação disponíveis (telejornal, novela, esportes, infantil, etc)[7]. Esta representação do usuário, por meio de tempos gastos em sub-gêneros, mantida internamente pelo ICM é chamada de perfil implícito.

A estatística coletada na última transição de programas é acumulada para cada um dos perfis implícitos candidatos mantidos pelo algoritmo. A cada transição de subgênero, um novo candidato é gerado[8]. Se um usuário se autentica explicitamente ou se o grau de certeza no perfil de um certo candidato é maior que um limiar, todos os candidatos gerados anteriormente são descartados[9], ao mesmo tempo que o perfil implícito é associado com uma conta de usuário. Note que cada usuário com uma conta (ou seja, todos os usuários exceto o anônimo) pode ter apenas um perfil implícito no sistema.

---

[5] Algumas modificações foram feitas nas classes PresentationContext e ContextManager para possibilitar o uso de seus serviços.

[6] As variáveis *default.curiprof* e *default.iprof*, acessíveis de qualquer aplicação NCLua, indicam o identificador de perfil implícito atual e o conteúdo do próprio perfil implícito, respectivamente. A variável *default.curiprof* tem a forma *userid*:*profileid*, onde *userid* corresponde ao identificador explícito do usuário atual e *profileid* corresponde ao identificador ativo do perfil implícito deste usuário. A variável *default.iprof*, por sua vez, corresponde a uma série de números separados por vírgulas. Os números indicam os valores dos vários atributos usados para representar o perfil.

[7] Observe que não usamos a estatística coletada pelo DA diretamente, pois ela não é acumulada em função dos vários sub-gêneros. Além disso, o DA considera apenas interações do usuário, deixando de lado, por exemplo, transições de programas. Também note que o perfil poderia ser formado por qualquer conjunto de informações. Por exemplo, transições de canais. A mudança da informação constante no perfil não implica em mudança na estratégia implementada. De fato, originalmente, pretendíamos usar informação fornecida pelo usuário na construção do perfil (dados demográficos como idade e sexo, bem como preferências declaradas de canais e interesses) em conjunto com a informação implícita. No decorrer do projeto, decidimos por evitar esta informação. Apesar disso, grande parte do componente de coleta explícita foi desenvolvido em NCL (ver Seção 4). Como mencionado antes, a variável *default.iprof* armazena a lista de valores dos vários atributos usados para representar o perfil, separados por vírgulas.

[8] No futuro pretendemos controlar o número de candidatos por combinar aqueles muito similares, via técnicas de agrupamento.

[9] Um usuário implícito é considerado explícito se o grau de certeza do ICM na identificação do usuário é superior a um certo limiar (94%). Se isto ocorre, o sistema age como se o usuário houvesse se autenticado (de fato, o ICM autentica o usuário, internamente).

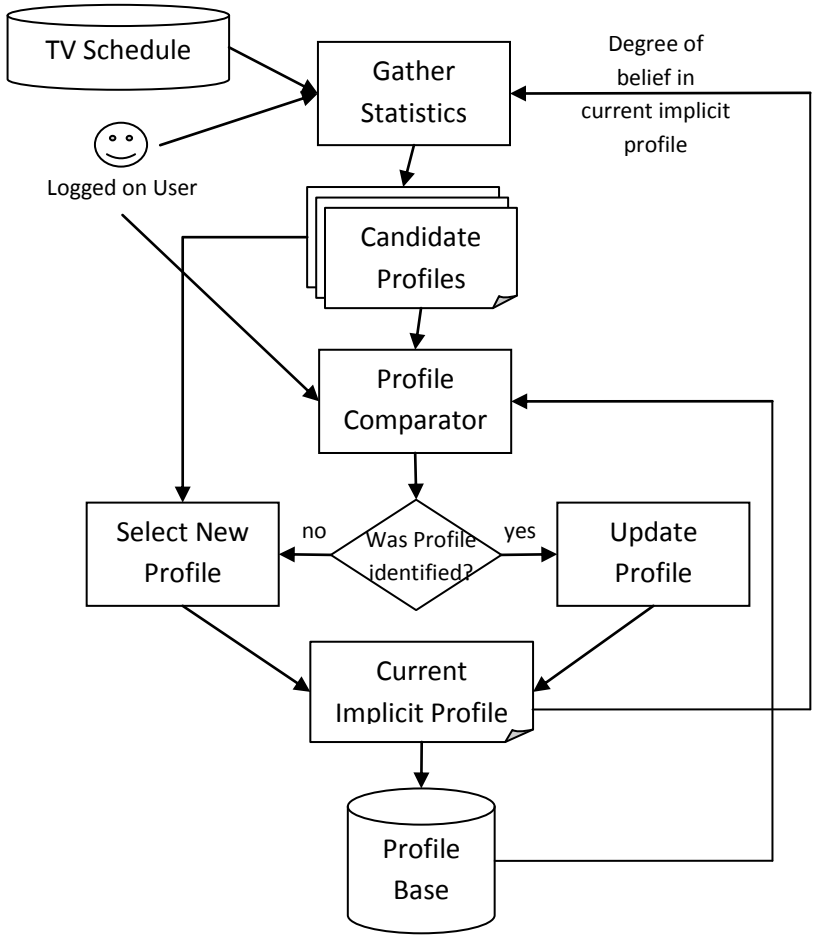


Figura 3: Determinação de usuário implícito

O ICM, então, compara cada um dos seus perfis candidatos com todos os perfis implícitos conhecidos e mantidos pelo Gerente de Contexto do Ginga (os perfis são mantidos como variáveis de contexto no STB)[10]. Caso nenhum dos perfis atuais seja identificado[11], um deles é selecionado como o perfil implícito corrente e inserido na base de perfis[12]. Caso um perfil implícito seja identificado como um daqueles na base, ele tem as suas estatísticas atualizadas. Se o perfil identificado difere do perfil implícito atual, o ICM torna este perfil o corrente. Para clientes NCLua notarem esta mudança, eles precisam monitorar a variável de contexto *default.curiprof*[13].

[10] Comparações são feitas usando um método vetorial simples baseado em cosseno. Seria interessante que, no futuro, fosse incorporado um método probabilístico mais sofisticado que leve em conta aspectos como popularidade de gêneros e assiduidade de usuários.

[11] Mais precisamente, se nenhum perfil candidato tem grau de similaridade com um perfil da base superior a um certo limiar, consideramos que o comportamento observado é novo.

[12] O critério de seleção usado no protótipo é a escolha do perfil mais antigo da sessão. Critérios alternativos seriam a escolha do perfil mais distinto ou de uma combinação dos candidatos.

[13] No protótipo, o acesso a esta variável (e também à variável *default.iprof*) é feito com auxílio do servidor de autenticação explícita (a ser descrito na Seção 4), uma vez que a versão usada do Ginga ainda não implementa o módulo *Settings*. Tal acesso é escondido, convenientemente, através da nossa

Os perfis são salvos como variáveis de contexto “default.iprof” pelo Context Manager. Um exemplo de um perfil salvo é dado a seguir:

```
:: = 0
 || 0
 default.focusBorderColor = blue
 default.iprof.0 = 1272392445,12,34,0,0,0,0,0,23,45,76,12,34,0
 ...
```

Neste exemplo, a quarta linha representa o perfil 0 do usuário 0, que é o usuário atual, de acordo com a convenção usada pelo Context Manager no arquivo de contextos. O primeiro valor depois do sinal de igual (1272392445) representa o momento em que o perfil foi criado enquanto os demais valores formam o vetor do perfil. Ou seja, o perfil 0 corresponde ao vetor <12, 34, 0, 0, 0, 0, 0, 23, 45, 76, 12, 34, 0>, onde cada inteiro corresponde ao tempo que usuário dedicou a um diferente gênero de programação. Note que, ao contrário dos demais usuários, o usuário anônimo pode ter múltiplos perfis.

À medida que o tempo passa, a base de perfis tende a manter perfis característicos dos usuários que usam aquela televisão. Note que certos perfis podem caracterizar grupos de usuários e não apenas usuários isolados.

A base de perfis implícitos é iniciada com perfis genéricos. Estes perfis correspondem a comportamentos típicos de diferentes classes de usuários (crianças, adolescentes, jovens, adultos e idosos, homens e mulheres, classes A, B e C).

Para evitar um crescimento muito grande desta base e, assim, mantê-la em um tamanho compatível com o de uma família típica, o ICM periodicamente identifica perfis muito similares, agrupando-os[14]. Além disso, perfis não utilizados por um período muito longo são eliminados da base.

3.2. Módulo de Publicidade Local

3.2.1. Introdução

Como visto na seção anterior, para determinar o contexto local de cada usuário, é necessário processar uma grande quantidade de informação relativa ao comportamento dos usuários e à programação. Em uma arquitetura de publicidade completamente centralizada, isto tanto

---

própria implementação do módulo *settings.lua*. Nesta implementação, todo o acesso a variáveis *default*, *user* e *system* é substituído por chamadas ao servidor de Autenticação.

[14] Esta funcionalidade foi incorporada ao MA. Em particular, foi implementado um algoritmo de agrupamento iterativo com previsão do número de grupos. O método de agrupamento é o *k-means*. A previsão do número de grupos é feita através da identificação do valor de *k* para o qual o coeficiente de inferência Bayesiana (BIC) deixa de apresentar aumentos significativos. Este valor de *k* é obtido por aplicação do método L. Para uma descrição do método L sugerimos o artigo “Determining the number of clusters/segments in Hierarchical Clustering” de Stan Salvador e Philip Chan. Para uma descrição do k-means iterativo, sugerimos a página de Marco Tuononen (http://cs.joensuu.fi/~mtuonon/). Finalmente, para uma descrição do coeficiente BIC sugerimos a página de Qinpei Zhao (http://cs.joensuu.fi/~zhao/).

implicaria em altos custos de transmissão quanto em excesso de processamento por parte do servidor de publicidade. Assim, nosso Serviço Recomendação de Publicidade Personalizada, é baseado em uma arquitetura cliente-servidor, onde o cliente reside na TVD, é capaz de receber e inserir publicidade no conteúdo sendo exibido, bem como enviar ao servidor a informação contextual necessária para que este possa selecionar as melhores propagandas. Este cliente, o Módulo de Publicidade Local, é descrito a seguir.

3.2.2. Descrição

O Módulo de Publicidade Local consiste de três componentes: o Ad Schedule Receiver (ADSR), o Ad Synchronizer (ADSY) e Ad Placer (ADPL). O ADSR é o componente responsável por receber escalas de publicidade e as publicidades, em si. O ADSY é responsável por sincronizar cada propaganda de acordo com a escala fornecida. Além disso, ele é responsável por monitorar mudanças de usuário e informá-las ao servidor de publicidade. Finalmente, o ADPL é responsável por apresentar a propaganda no conteúdo. Todos estes componentes estão implementados em Lua, o que possibilita ao servidor modificar a sua política local por re-enviar estes programas quando necessário.

Note que nas descrições a seguir, assumimos que:

1) O Módulo de Publicidade Local é reiniciado a cada mudança de canal[15]. Assim, nenhum dos componentes descritos precisa ser notificado sobre mudanças de canal. Isto não significa que escalas de propaganda recebidas sejam descartadas. Elas podem ser úteis caso o usuário volte a um canal para o qual ainda há uma escala válida;
2) A informação sobre a localização do servidor de publicidade é enviada junto com os componentes pelo provedor de serviço que, por sua vez, sabe o canal em que a TV está sintonizada. Neste sentido, o servidor de publicidade é dedicado ao canal. Note que isto não impede a formação de grandes comunidades de anunciantes/usuários e facilita processos de escalonamento de clientes. Também tem a vantagem de facilitar a manutenção de propagandas localmente para os canais mais assistidos pelos usuários;

A **Figura 4** apresenta uma descrição geral da arquitetura do módulo de publicidade local e suas interações com os componentes do Ginga e do servidor de publicidades.

Como podemos observar na **Figura 4**, o ADSR monitora o servidor de publicidade, em busca de novas escalas e propagandas. Sempre que estas são enviadas pelo servidor, o ADSR as armazena localmente e notifica ao ADSY sobre sua disponibilização. Note que o ADSR é responsável por implementar políticas de economia de espaço e uso da banda. Para tanto, o ADSR sempre procura manter um espaço máximo ocupado por propagandas, eliminando aquelas usadas menos freqüentemente. Ele apenas solicita ao servidor as propagandas que já não estão presentes localmente[16]. A informação constante na escala corresponde ao tempo de inserção da propaganda (data e hora), o tempo de exibição, as coordenadas do botão para

[15] Isto se deve em grande parte à forma como funciona o terminal de TVD implementado na Fucapi.
[16] Nenhuma política de espaço e uso de banda está implementada no protótipo.

fechar a propaganda e uma referência à propaganda em si[17]. As propagandas, por sua vez, são transmitidas como um agregado único comprimido, contendo um programa Lua e mídias correspondentes.

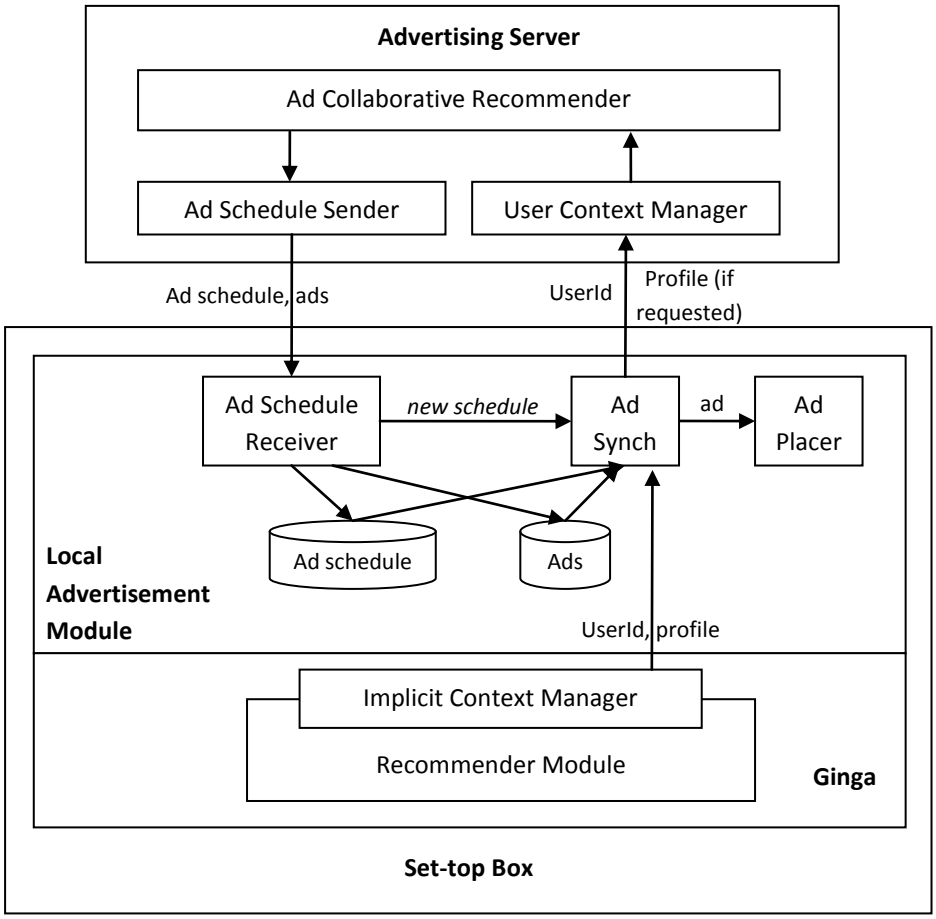


Figura 4: Módulo de Publicidade Local

O componente ADSY, por sua vez, acorda a cada meio segundo e verifica se há alguma propaganda a ser inserida no conteúdo, de acordo com a escala vigente. Caso haja, o ADSY elimina a entrada da escala, recupera a propaganda do banco de dados, a descompacta junto com as mídias correspondentes e notifica ao ADPL que a exiba. Ao notificar o ADPL, o ADSY informa as coordenadas do botão de controle.

O ADSY ainda monitora o ICM sobre uma mudança de usuário implícito[18]. Ao perceber uma mudança, o ADSY é capaz de saber quem é o novo usuário implícito ativo e seu perfil de

[17] Internamente, a representação ainda inclui informação sobre o serviço (canal) para qual a escala é válida. Isto é útil como estratégia de *cache*. O protótipo atual ainda não inclui esta informação.

[18] Para tanto, ele verifica se o conteúdo da variável de contexto *default.curiprof* foi modificada. Note que qualquer mudança explícita ou implica de usuário provoca uma mudança no valor desta variável, uma vez que ela é uma composição do identificador do usuário explícito e seu perfil implícito. Logo, o ADSY monitora qualquer mudança de usuário.

audiência por sub-genêro de programação[19]. O ADSY então pára a exibição de propagandas usando a escala atual. Ele também notifica ao servidor de publicidade, via canal de retorno, sobre a mudança ao mesmo tempo em que já envia ao servidor a identificação do usuário implícito corrente. Note que em resposta à notificação, o servidor pode solicitar o perfil do usuário. Se este for o caso, o ADSY envia a informação solicitada ao servidor[20]. Uma vez parado, o ADSY só retoma a exibição de propagandas após uma notificação do ADSR (de nova escala válida).

Finalmente, o ADPL é o componente que realmente insere a propaganda no conteúdo. Ele exibe o botão para fechar a propaganda na posição adequada e executa o programa Lua correspondente à propaganda. Caso o usuário solicite o fechamento da propaganda, ele termina a execução do programa correspondente.

Note que ao permitir a execução de qualquer tipo de programa Lua, o ADPL possibilita a exibição de diversos formatos de publicidade (incluindo interativas) em diferentes posições do vídeo. Como implementado atualmente, entretanto, o Módulo de Publicidade Local não é capaz de registrar o histórico de interação dos usuários com as propagandas, como na Web. Dada a importância desta informação para o servidor de publicidade, futuramente, um novo protocolo para as propagandas deve ser desenvolvido para permitir o registro desta informação e o seu envio ao servidor.

4. Módulo de Identificação Explícita do Usuário de TV Digital

4.1. Introdução

Um grande atrativo de tecnologias atuais é a possibilidade de personalizar aplicativos e interfaces com intuito de promover a facilidade de uso em um ambiente agradável. O GingaNCL oferece um conjunto de serviços para personalização e contextualização da TV. Contudo, deixa para o desenvolvedor dos terminais de acesso os detalhes relativos ao processo de autenticação. Neste tópico, descreveremos as funcionalidades do Módulo de Identificação Explícita do Usuário de TV Digital. Este módulo irá prover um serviço de identificação voluntária (explícita) do usuário.

4.2. Descrição

O Módulo de Identificação Explícita do Usuário de TV Digital consiste de dois componentes que facilitam o processo de autenticação do usuário da TV. O primeiro componente é o Authentication Server (AUTS) que oferece uma série de serviços para a identificação explícita

[19] Através da variável de contexto *default.curiprof*, ele obtem informação sobre usuário ativo explícito e seu perfil implícito atual. Através da variável de contexto *default. iprof*, ele obtem o perfil implícito do usuário.

[20] O envio do perfil não é feito compulsoriamente para dar ao servidor a possibilidade de implementar sua própria política de minimização de tráfego. O servidor pode, por exemplo, manter informações sobre perfis anteriores dos usuários e decidir que não vale a pena ainda atualizar estas informações. Note, entretanto, que atualmente o servidor não pode solicitar este perfil a não ser mediante uma notificação de mudança. Este serviço de solicitação de perfil, independente de uma notificação, pode ser implementado em uma versão futura.

de usuários e gerência de contas. O segundo componente é o Authentication Frontend (AUTF) que consiste de uma aplicação em NCLua que fornece uma interface amigável para os serviços fornecidos pelo AUTS. A **Figura 5** apresenta uma descrição geral da arquitetura do módulo de identificação explícita do usuário e suas interações com os componentes do Ginga.

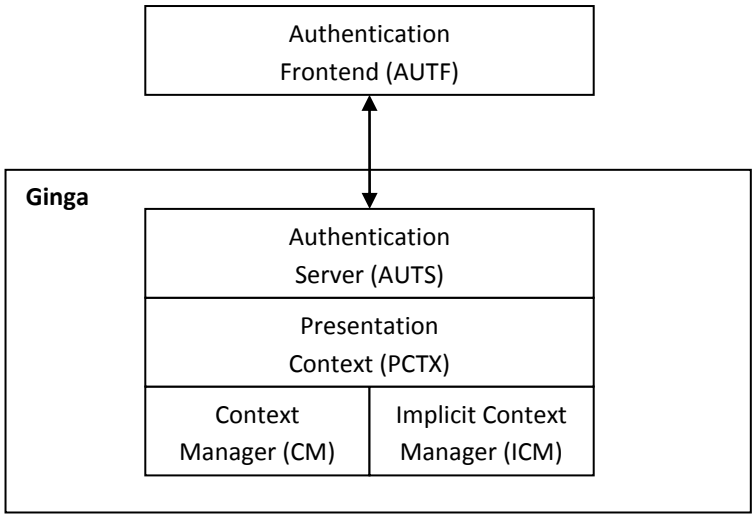


Figura 5: Módulo de Identificação Explícita do Usuário

O AUTS oferece serviços de autenticação no topo dos gerentes de contexto explícito (Context Manager - CM) e implícito (Implicit Context Manager - ICM). Ele acessa os gerentes de contexto por meio do gerente de contexto de apresentação (Presentation Context - PCTX). O acesso por meio do PCTX é necessário, uma vez que o PCTX oferece o serviço de acesso concorrente ao CM e ao ICM.

O AUTS é iniciado pelo PCTX e fica escutando requisições *socket* na porta 8183. O AUTS oferece os seguintes serviços[21]:

a) Inclusão de nova conta. Mediante uma senha de acesso, os seguintes dados devem ser informados: nome, senha, idade, código postal e sexo. Um identificador é criado para o usuário. Este se torna o usuário ativo (autenticado). Um perfil implícito também é automaticamente criado para o usuário;
b) Remoção de conta. Mediante a senha de acesso do usuário, sua conta pode ser removida. Todos os seus perfis explícitos são removidos. Após a remoção, o usuário ativo passa a ser o "default". Note que o usuário "default" não pode ser removido;
c) Atualização de conta. Mediante a senha de acesso do usuário, os seguintes campos podem ser modificados: nome, senha, idade, código postal e sexo;
d) Consulta a dados da conta. Mediante a senha de acesso do usuário, os seguintes

[21] Para fins de depuração e suporte temporário a serviços ainda não oferecidos pelo Ginga (como acesso a variáveis de contexto via módulo *Settings*), o AUTS oferece a possibilidade de consulta e atualização de valores de variáveis de contexto, mediante uma senha de acesso. Também para fins de depuração e monitoração, um conjunto de novas variáveis de contexto "debug.*" são mantidas pelos gerentes de contexto. Versões finais do AUTS não deverão oferecer este serviço, entretanto.

campos podem ser consultados: nome, idade, código postal e sexo;

e) Lista de usuários. Servidor devolve a lista de usuários explícitos conhecidos e seus identificadores. Não é necessário autenticação prévia;
f) Identificador do usuário atual. Servidor devolve o identificador do usuário autenticado no momento (usuário ativo). Não é necessário autenticação prévia;
g) Login. Mediante a senha de acesso do usuário, o servidor torna este o usuário atualmente autenticado (ativo);
h) Logout. Mediante a senha de acesso do usuário, o servidor torna ativo o usuário "default";

Note que todos os serviços acima são sincronizados com os gerentes de contexto, de forma que as informações das variáveis de contexto são sempre consistentes durante a execução do Ginga. A persistência dos dados fica a cargo do próprio Ginga. O controle de concorrência é feito pelo PCTX. Assim, não há conflito de acesso entre o servidor internamente e aplicações NCLua rodando no topo do Ginga.

O AUTF oferece ao usuário uma interface amigável ao AUTS, escrita em NCLua, como exemplificado na **Figura 6**. Nesta interface, o usuário tem acesso tanto ao vídeo que esta assistindo como a uma área de identificação. Como a gerência de contas deve exigir do usuário a entrada de muitos caracteres, um teclado virtual para uso com controle remoto é fornecido como uma ferramenta de conveniência. Note que quando o usuário ativa o serviço de autenticação, todos os demais serviços de publicidade são temporariamente desativados.

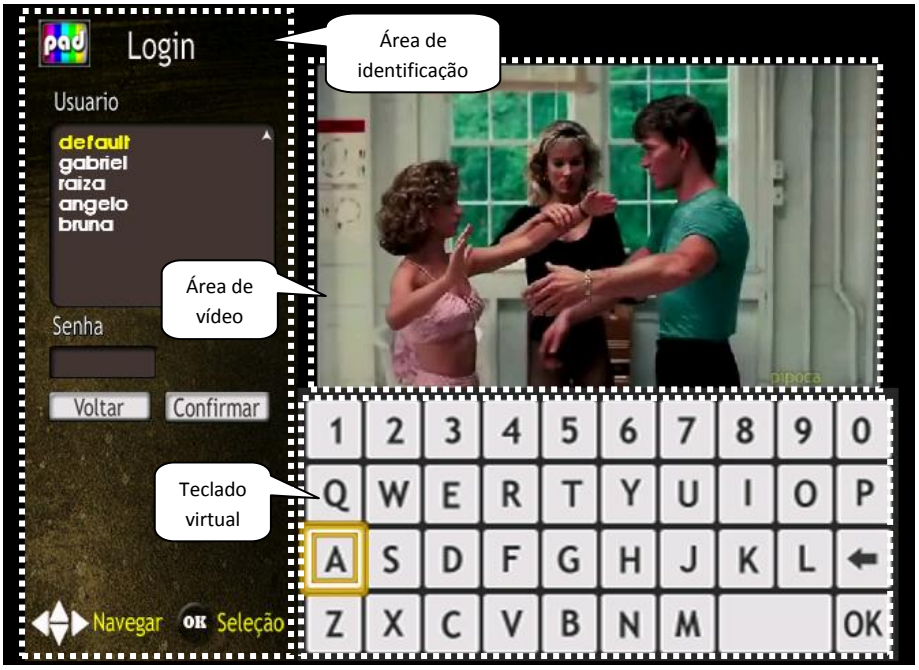


Figura 6: Interface de autenticação do usuário

O *menu* oferece as seguintes opções básicas: *login*, *mudar senha*, *novo usuário* e *remover usuário*. Através da funcionalidade *Login*, o usuário pode se identificar na TV. Através da opção *Mudar Senha*, ele pode modificar sua senha de acesso. A opção *Novo Usuário* permite a

criação de novos usuários. O usuário deve informar o nome, a idade, o sexo e a senha de acesso. O nome será o seu identificador de *login*. A opção *Remover Usuário* permite ao usuário excluir uma conta. Um pequeno manual da interface é fornecido em Apêndice anexo a este documento.

5. Protótipo

Nesta seção, descrevemos o protótipo usado para testar os componentes implementados. Em particular, descrevemos o cenário usado no protótipo, as instruções para execução do protótipo e uma execução do protótipo. O material necessário para executar o protótipo segue no DVD anexo a este documento.

5.1. Cenário

Neste protótipo, consideramos um cenário envolvendo um domicílio com uma família formada por quatro pessoas, uma televisão e dois canais.

Os membros da família são descritos na **Tabela 3**, bem como suas preferências em geral. Note que as duas mulheres (mãe e filha) têm comportamento geral bem próximo. Ambas se diferem dos homens (pai e filho). O menino apresenta um comportamento bem diferente dos demais.

Nome	Idade	Sexo	Perfil (preferências)
Gabriel	8	M	Desenhos animados (ação e comédia)
Raíza	17	F	Filmes (comédia e romance), Séries, Novela e Música
Ângelo	48	M	Filmes (ação), Esporte, Telejornal. Um pouco de Séries e outros Filmes
Bruna	44	F	Séries e Novela, um pouco de Filmes (comédia e romance)

Tabela 3: Família simulada em nosso protótipo

Para simplificar a simulação, os usuários são descritos por um perfil implícito pequeno, caracterizado pela audiência deles em relação aos seguintes sub-genêros de programação: outros, musicais, telejornais, romances, infantil, ação, novela, séries, esporte, comédias e documentários.

Nesta simulação, supomos que os usuários já utilizaram a TV por certo tempo, de forma que a TV acumulou certa quantidade de informação sobre os seus comportamentos. Para simplificar a visualização destes comportamentos, supomos que os perfis dos usuários estão identificados[22]. Para simular mais adequadamente um cenário de uso real, também adicionamos quatro padrões de comportamento espúrios (não identificados) que representam, por exemplo, padrões capturados pela TV quando membros da família assistem a programação juntos. Naturalmente, tais padrões não estão identificados e aparecem como

[22] É possível (e bastante provável) que em um ambiente real, nenhum dos usuários no cenário dado tivesse se identificado previamente na TV. Neste caso, todos eles apareceriam como perfis do usuário *default*.

perfis do usuário “default”.

Os canais disponíveis na simulação são dois, um especializado em filmes românticos e musicais, o “Cine Pipoca”, e outro em filmes de ação e comédia, o “Cine Ação”. A programação sintonizada na TV é descrita na **Tabela 4**. Nesta tabela, a coluna tempo indica o número de segundos, desde o início da programação, para o início da exibição de um serviço.

Tempo (seg)	Servidor	Serviço	Gênero	Sug-Gênero
0	Cine Pipoca	Dirty Dancing (1987)	Filme	Musical
60	Cine Pipoca	Pretty Woman (1990)	Filme	Romance
118	Cine Ação	Rocky (1976)	Filme	Ação
183	Cine Ação	Liar Liar (1997)	Filme	Comédia
241	Cine Pipoca	Greasy (1978)	Filme	Musical

Tabela 4: Programação usada em nosso protótipo (linhas em cinza correspondem ao canal Cine Ação). Note que foram realizadas trocas de canal aos 118 e 241 segundos.

Como observado na **Tabela 4**, a programação consiste de cinco fragmentos de filmes com cerca de um minuto cada, separados por intervalos de treze segundos.

O conteúdo das propagandas foi extraído de uma coleção mantida por um operador nacional de publicidade contextual para Web. Entretanto, elas receberam uma apresentação visual mais adequada ao ambiente de vídeo.

5.2. Instruções para iniciar a simulação

Copie o arquivo Ginga-SRPP.tar.gz do DVD para um diretório na maquina local. Depois, descompacte o arquivo copiado (supondo uma máquina Linux, use o comando tar xzvf Ginga-SRPP.tar.gz). Execute, então, o arquivo Ginga-SRPP/semre.vmx usando o VMWare Player. O VMWare Player deve abrir uma caixa de diálogo, perguntando se a máquina virtual foi movida ou copiada. Responda que foi copiada. A máquina virtual deve então entrar em execução. Assim, você terá a máquina virtual Ginga (GingaVM) rodando no computador hospedeiro (Host).

No Host, abra um terminal e faça um SSH para a GingaVM. Este terminal será usado para acompanhar a simulação. Por exemplo, supondo que o IP da GingaVM é 172.16.92.130, digite:

```
ssh -X root@172.16.92.130
```

Entre a senha “telemidia”. Em seguida, vá para o diretório “/usr/local/sbin” e execute a demonstração, como descrito abaixo:

```
cd /usr/local/sbin
bash demo.bash
```

O script demo.bash irá executar:

(1) o processo *adserver* (em background) que corresponde ao simulador do Servidor de Publicidade. Este servidor é responsável por enviar escalas de publicidade e publicidades para o cliente rodando no Ginga. Eventualmente ele será avisado pelo Ginga sobre trocas de usuários, o que o levará a recalcular as escalas e re-enviá-las.

(2) o processo *profileMonitor* (em *background*) que nos permite monitorar eventos que ocorrem no Ginga, relacionados aos usuários identificados, aos perfis reconhecidos implicitamente e à programação em exibição. Um pequeno manual do *profileMonitor* é fornecido em apêndice anexo a este documento.

(3) o Ginga-NCL junto com o script em Lua correspondente aos vários componentes do Módulo de Publicidade Local.

Quando a simulação terminar, derrube as aplicações rodando em background. Para isto, digite:

```
bash killdemo.bash
```

5.3. Descrição da simulação

Ao iniciar a simulação com o comando “bash demo.bash”, as janelas da programação de TV e do Profile Monitor (PM) serão exibidas. Para garantir o monitoramento automático dos estados internos do Ginga, selecione a caixa “Automatic update every 1 second”, conforme a **Figura 7**. Logo após selecionar a caixa de diálogo, uma série de informações é exibida no PM.

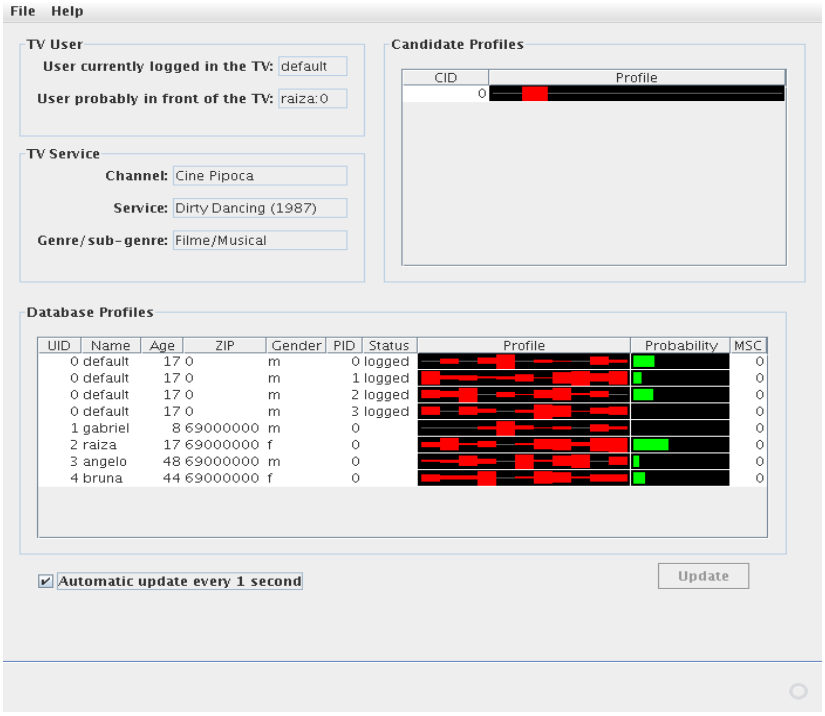


Figura 7: Janela de monitoração após primeira atualização.

Estas informações indicam que nenhum usuário está autenticado atualmente (neste caso, o Ginga assume como usuário autenticado o usuário anônimo “default”), que a TV foi ligada sintonizada no canal “Cine Pipoca” e que o programa atualmente em exibição é um filme do gênero musical chamado “Dirty Dancing”. No quadro “Databse Profiles”, podemos ver todos os perfis conhecidos pela TV. São quatro usuários que já se identificaram em sessões anteriores, além de quatro perfis anônimos capturados previamente pela TV.

No quadro “Candidate Profiles”, ainda podemos observar que, internamente, o Ginga construiu um único perfil candidato para comparar com os perfis previamente conhecidos. Como o programa em exibição na TV é um filme do gênero musical, o candidato construído foi reconhecido como alguém que gosta de musicais. Desde que um dos usuários da casa (Raíza, uma adolescente de 17 anos) gosta muito de musicais, o Ginga supõe que ela está em frente da TV, o que pode ser notado pela probabilidade do seu perfil 0. Note que, depois da Raíza, o perfil mais próximo do candidato atual é o perfil 0 do usuário anônimo.

Na seqüência da simulação, o MPL recebe, com base nas informações de conteúdo e usuário, duas propagandas e as exibe. Podemos ver as propagandas exibidas na **Figura 8**. As propagandas são exibidas em diferentes localizações na área de vídeo e são visualmente diferentes[23]. A primeira propaganda no filme *Dirty Dancing* é exibida na parte de baixo do vídeo como uma imagem transparente azul. A segunda é exibida no alto do vídeo com uma imagem de um casal dançando, à esquerda. Em ambas as propagandas, o botão “fechar” (uma seta para baixo, vermelha) é exibida e controlada pelo MPL.

Figura 8: Propagandas exibidas para o primeiro vídeo, *Dirty Dancing (1987)*.

Seguindo a simulação, o primeiro filme termina e tem início o segundo (*Pretty Woman*, um romance de 1990). A **Figura 9** corresponde ao estado interno do Ginga, após esta transição de programação. Com a mudança para a exibição de um filme romântico, outro candidato foi criado. A partir dos dois candidatos existentes, o Ginga estima que um novo usuário (Bruna, sexo feminino, 44 anos) passou a assistir TV. Podemos ver na coluna “MSC” (Most Similar

[23] A primeira propaganda é composta por um script Lua e uma imagem, enquanto a segunda é formada por um script Lua e duas imagens.

Candidate) da tabela “Database Profiles” os canditados mais parecidos com cada perfil. Note que a probabilidade da usuária Raíza continuar assistindo TV ainda é alta, como pode ser visto na coluna de probabilidades.

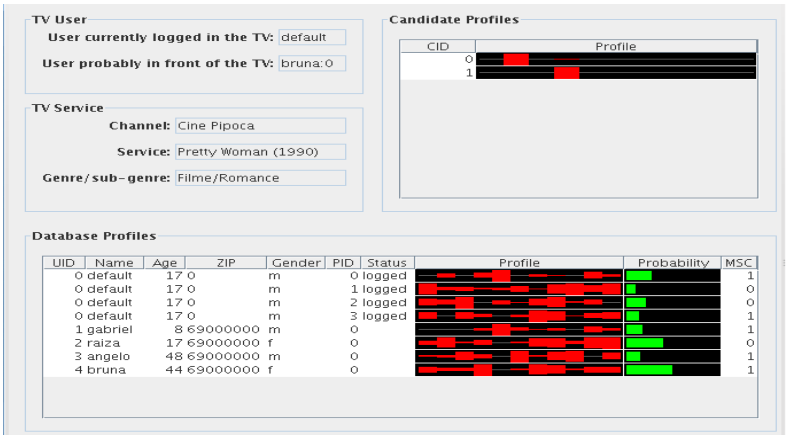
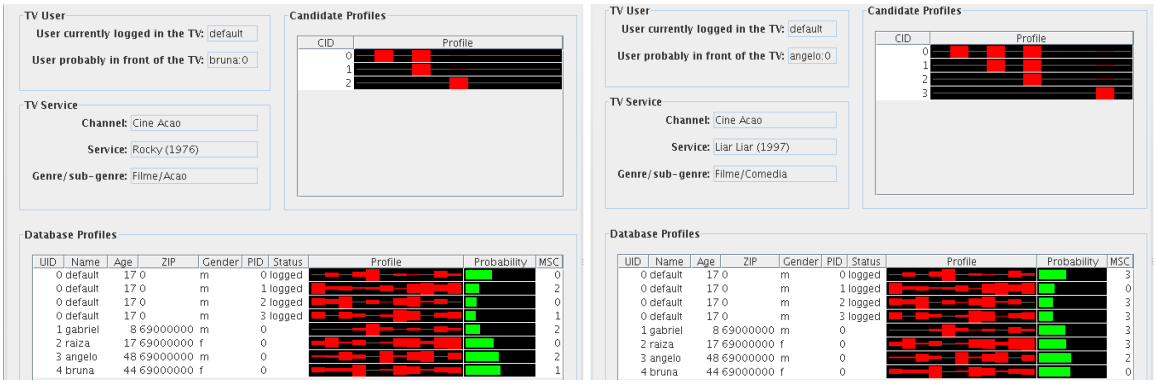


Figura 9: Janela de monitoração após primeira mudança de programação.

Após alguns segundos, é feita uma mudança de canal (Cine Pipoca para Cine Ação), que resulta na exibição de um filme de ação, *Rocky*. Apesar da probabilidade para o usuário adulto do sexo masculino (Ângelo) ser alta, a usuária atual, Bruna, também apresenta em seu perfil uma pequena preferência por filmes de ação. Como resultado, o Ginga assume que o usuário não mudou, como pode ser visto na **Figura 10a**. Logo em seguida, então, há uma nova mudança de programação e tem início uma comédia, *Liar Liar*. Desta vez, a melhor opção se torna o usuário Ângelo e o Ginga assume que ele se torna o usuário corrente (cf. **Figura 10b**).

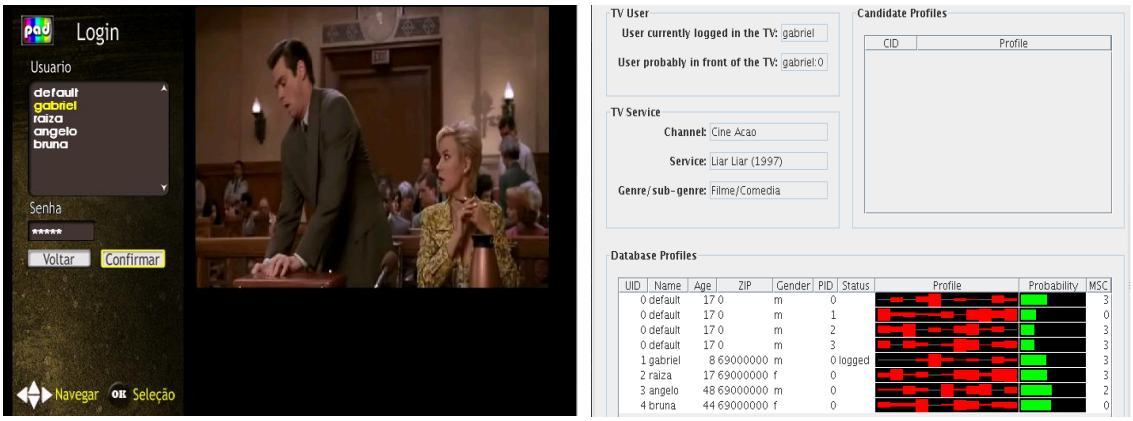


(a) (b)

Figura 10: Monitor após início dos filmes Rocky (a) e Liar Liar (b).

Note que até este momento, nenhum usuário se autenticou explicitamente. Imagine, no entanto, que durante a exibição do filme Liar Liar, o usuário Gabriel (um menino de oito anos) se autentica, conforme mostra a **Figura 11a**. Perceba que, enquanto o usuário se autentica, a exibição de propagandas é desabilitada. Em seguida, como pode ser visto na **Figura 11b**, ele se torna o usuário corrente, mesmo embora o Ginga ainda acredite que o comportamento até então fosse mais compatível com o do pai dele.

Internamente, o Ginga assume que ele deve ser o usuário em frente à TV. Assim, nenhum dos candidatos mantidos até então nesta seção devem corresponder ao comportamento do usuário Gabriel. Como resultado, os quatro candidatos são apagados.



(a) (b)

Figura 11: Autenticação do usuário Gabriel (a) e estado resultante (b).

Na próxima transição de programação, apenas um candidato será criado (**Figura 12**). Em outras palavras, o perfil do usuário Gabriel deve ser registrado a partir deste ponto. Está é, desde o início da simulação, a primeira vez que o usuário explicitamente autenticado corresponde ao usuário que o Ginga crê estar assistindo TV.

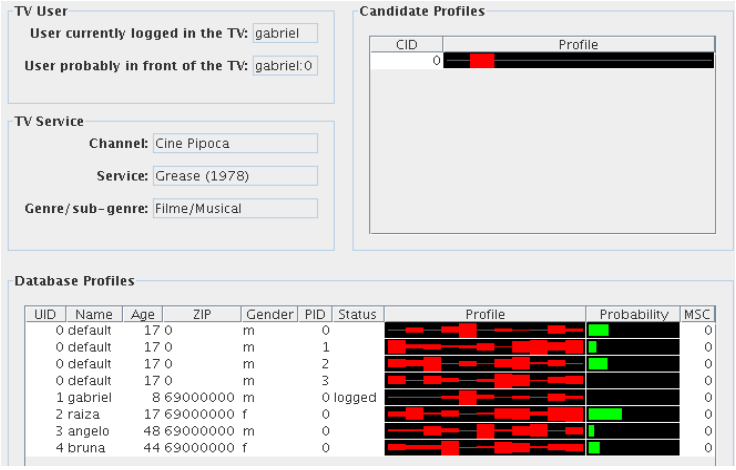


Figura 12: Estado do Ginga após transição de programação.

6. Detalhes de Implementação

Nesta seção, fornecemos alguns detalhes adicionais sobre a implementação do Gerenciador de Contexto Implícito, do Módulo de Publicidade Local e do Módulo de Identificação Explícita do

Usuário de TV Digital[24]. Note que não *são* descritas nesta seção modificações feitas no código fonte com o intuito de suportar o processo de monitoração da execução, já que tais funcionalidades não farão parte da versão final a ser entregue.

6.1. Gerenciador de Contexto Implícito

O gerenciador de contexto implícito é implementado através dos seguintes programas:

- `gingaNcl/src/lince/icm/ImplicitContextManager.cpp`: implementação da classe *ImplicitContextManager* que define o conjunto de serviços oferecidos pelo gerente de contexto implícito (ICM). O ICM, de fato, gerencia uma série de perfis de usuário que são usados para estimar quem é o usuário corrente. Estes perfis são chamados perfis implícitos. Note que o algoritmo descrito na **Figura 3** é implementado através do método *guessImplicitUser()*. Esta classe também inicia a classe *DataInterface*, responsável por simular a transmissão de meta-dados no TS, para fins de demonstração. A classe é definida no arquivo `gingaNcl/include/lince/icm/ImplicitContextManager.h`.

- `gingaNcl/src/lince/icm/ImplicitProfiles.cpp`: programa com a implementação da classe *ImplicitProfiles* que define e gerencia os perfis implícitos usados pelo ICM. Perfis implícitos são armazenados como variáveis de contexto. A classe é definida no arquivo `gingaNcl/include/lince/icm/ImplicitProfiles.h`. Note que esta classe se utiliza de serviços de agrupamentos, oferecidos pelo componente de mineração (*MiningAgent*) do módulo *Recommender*.

- `gingaNcl/src/lince/icm/DataInterface.cpp`: programa com a implementação da classe *DataInterface*, que simula a transmissão de meta-dados das Tabelas SI via TS, e da classe *TVData*, que representa os dados transmitidos pelo TS. A classe DataInterface simula a transmissão de dados de acordo com um arquivo texto (`/usr/local/sbin/sidata.csv`). Este arquivo apresenta, em cada linha, os seguintes campos, separados por vírgula: (a) tempo em segundos, a partir do início da simulação, para o início do programa; (b) nome do canal; (c) nome do serviço/programa; (d) gênero do serviço e (e) sub-gênero do serviço. A classe é definida no arquivo `gingaNcl/include/lince/icm/DataInterface.h`.

Note que para determinar implicitamente o usuário, é necessário verificar constantemente se o seu comportamento pode ser identificado como um dos comportamentos previamente observados ou, mesmo, se é um comportamento completamente novo. Para isso, o método *guessImplicitUser* da classe *ImplicitContextManager* deve ser invocado periodicamente. Para possibilitar esta execução periódica, nós modificamos o programa de escalonamento do módulo *recommender* (`gingaNcl/src/lince/scheduleragent/Scheduler.cpp` e `gingaNcl/include/lince/scheduleragent/Scheduler.h`). Este programa implementa a classe *Scheduler* que inicia o módulo de recomendação periodicamente (de acordo com o arquivo `/usr/local/sbin/recommender.ini`). Em nossa implementação, fizemos duas modificações neste programa:

[24] Embora esta seção foque em modificações realizadas em códigos-fonte, é importante ressaltar que outras modificações foram feitas em arquivos *makefile*. Em particular foram modificados os arquivos `gingaNcl/ginga-cpp/src/Makefile.am`, `gingaNcl/gingancl-cpp/src/adaptation/Makefile.am`, `gingaNcl/bin/Makefile.am` e `gingaNcl/gingancl-cpp/include/adaptation/Makefile.am`.

- A primeira consistiu em adicionar uma variável (*stopflag*) à classe *Scheduler* que possibilitasse a interrupção voluntária da execução dos algoritmos de mineração e detecção de usuário implícito. Esta interrupção é solicitada pelo módulo principal da implementação de referência do Ginga (`gingaNcl/ginga-cpp/src/main.cpp`), através da invocação do serviço *stop()* da classe *Scheduler* (que muda o valor de *stopflag*) seguido da chamada *pthread_join()* que aguarda o fim da execução do escalonador. Esta modificação foi necessária para evitar uma interrupção brusca do escalonador, garantindo uma finalização correta do sistema.

- A segunda modificação consistiu na invocação do método *guessImplicitUser()* da classe *ImplicitContextManager*. Note que o método só é invocado depois que o gerente de contexto foi iniciado. Para determinar se ele já foi inicializado, o método *instantiated*() foi adicionado à classe *PresentationContext*.

Finalmente, algumas mudanças foram realizadas nos arquivos de cabeçalho `gingaNcl/include/lince/filteragent/LoadCacheDatabaseInformation.h` e `gingaNcl/include/lince/localagent/User.h` para garantir a correta compilação dos programas.

Para realizar seus serviços, a classe *ImplicitProfiles* se utiliza de serviços de agrupamento oferecidos pelo componente de mineração do módulo de recomendação. Note que estes serviços foram adicionados ao módulo de recomendação, através dos programas:

```
gingaNcl/src/lince/miningagent/ikmeans/ikmeans.cpp
gingaNcl/src/lince/miningagent/ikmeans/kmeans.cpp
gingaNcl/src/lince/miningagent/ikmeans/memctrl.cpp
gingaNcl/src/lince/miningagent/ikmeans/interfc.cpp
gingaNcl/src/lince/miningagent/ikmeans/file.cpp
gingaNcl/src/lince/miningagent/ikmeans/cb.cpp
gingaNcl/src/lince/miningagent/ikmeans/reporting.cpp
gingaNcl/src/lince/miningagent/ikmeans/random.cpp
gingaNcl/include/lince/miningagent/ikmeans/ikmeans.h
gingaNcl/include/lince/miningagent/ikmeans/kmeans.h
gingaNcl/include/lince/miningagent/ikmeans/memctrl.h
gingaNcl/include/lince/miningagent/ikmeans/interfc.h
gingaNcl/include/lince/miningagent/ikmeans/file.h
gingaNcl/include/lince/miningagent/ikmeans/cb.h
gingaNcl/include/lince/miningagent/ikmeans/reporting.h
gingaNcl/include/lince/miningagent/ikmeans/random.h
gingaNcl/include/lince/miningagent/ikmeans/owntypes.h
```

No arquivo *ikmeans.cpp* acima, é implementada a classe *IterativeKMeans*. Esta classe fornece o serviço de agrupamento iterativo com previsão do número de grupos. A previsão do número de grupos é feita através da identificação do valor de *k* para o qual o coeficiente de inferência Bayesiana (BIC) deixa de apresentar aumentos significativos. Este valor de *k* é obtido por aplicação do método L, implementado através do método *LMethod*. O algoritmo de cluster utilizado é o clássico k-means implementado no arquivo *kmeans.cpp*. Neste mesmo arquivo, temos a implementação do coeficiente de inferência Bayesiana. Os demais programas fornecem vários serviços de apoio tais como gerência de memória, acesso a arquivos, algoritmos de grafos, geração de números aleatórios e exibição de informações.

6.2. Módulo de Publicidade Local

Todos os componentes do módulo de publicidade local são implementados como aplicações NCLua. Em particular, ele consiste dos seguintes programas:

- *AdSchedulerReceiver.lua*: Implementação do componente *AdSchedulerReceiver.* Recebe do Servidor uma notificação informando o envio das próximas propagandas e a atualização da escala (o arquivo *Escalas.txt*). Neste arquivo, são especificados o momento em que a propaganda deve ser apresentada, sua posição na tela e seu tempo de duração. O *AdSchedulerReceiver.lua* então analisa que propagandas ainda podem ser exibidas e as salva em *EscalaNova.txt*. Em seguida, notifica o *AdSynchronizer* sobre a nova escala e a possibilidade de exibição das propagandas.

- *AdSynchonizer.lua*: Juntamente com *UserInfo.lua*, compõe o *AdSynchronizer*. O principal objetivo é verificar no arquivo *EscalaNova.txt* em que momento a próxima propaganda deverá ser exibida, salvar as suas mídias na pasta */tmp* e notificar ao *AdPlacer.* Este procedimento só é válido se não houver ocorrido mudança de usuário. Em caso de mudança, o programa não faz nada até receber uma notificação do *AdSchedulerReceiver* de atualização das escalas.

- *UserInfo.lua*: Responsável por verificar quem é o usuário que está assistindo ao vídeo. Se houver mudança de usuário, envia um aviso ao servidor com o perfil do novo usuário.

- *propsocket.lua*: Compõe o *AdPlacer*. Tem por objetivo exibir a propaganda do diretório */tmp*. Escuta o *AdSynchonizer*, esperando por informações da próxima propaganda (posição e tempo de duração da propaganda, bem como posição do botão *Fechar*). De posse destas informações, ele informa a aplicação NCL que a propaganda deve ser exibida. No decorrer desse processo, a propaganda pode ser fechada manualmente pelo usuário através do botão fechar (tecla *CURSOR_DOWN*). Caso não haja intervenção manual, a propaganda segue o curso natural e se fecha no decorrer do tempo especificado pelo *AdSynchonizer*.

- *luaprop.lua*: programa responsável pela exibição da propaganda. Carrega as imagens do diretório */tmp/propmedia* e desenha a propaganda.

6.3. Módulo de Identificação Explícita do Usuário de TV Digital

O módulo de identificação explícita é formado por dois componentes. O primeiro é o servidor de autenticação que é executado no Ginga, a partir do PresentationContext. O servidor de autenticação é implementado no seguinte arquivo:

- `gingaNcl/gingancl-cpp/src/adaptation/context/AuthenticationServer.cpp`: neste arquivo, temos a implementação da classe *AuthenticationServer* que oferece uma série de serviços de manutenção de contas de usuários. As contas são armazenadas como variáveis de contexto, através de serviços fornecidos pelas classes *PresentationContext*, *ContextManager* e *ImplicitContextManager*. A classe é definida no arquivo `gingaNcl/ginganclcpp/include/adaptation/context/AuthenticationServer.h`.

Note que uma série de modificações foram efetuadas na classe *PresentationContext* para que ela pudesse invocar o servidor de autenticação. Em particular:

- Foi adicionada à classe o método *InitializeContext*() que invoca os métodos *initializeUserContext*() e *initializeSystemValuesI*(). Esta modificação foi necessária porque a implementação de referência do Ginga não suporta a mudança de contexto durante uma seção de apresentação. Com a mudança, contextos de usuário e sistema podem ser reiniciados quando é solicitada ao servidor de autenticação a autenticação de um novo usuário.
- O construtor da classe foi modificado para iniciar a *thread* responsável por invocar o servidor de autenticação. Da mesma forma, o método destruidor foi criado para solicitar e aguardar a finalização da *thread*.
- Foi adicionada à classe a variável *stop* para possibilitar a interrupção voluntária do servidor de autenticação.
- O serviço *saveUserAccounts*() é invocado no método *save*() da classe, de forma a garantir que os dados das contas dos usuários serão salvos ao fim da seção. De fato, apenas tiramos o comentário que havia na linha correspondente à invocação deste serviço.
- Foi adicionada à classe o serviço *getContextManager*() que retorna o ponteiro para o gerente de contexto que está sendo utilizado pela classe *PresentationContext*.
- Foi adicionada à classe os serviços *lock*() e *unlock*(). A classe *PresentationContext* é responsável pelo controle de acesso concorrente às variáveis de contexto. A adição destes serviços permite que outras classes que usem as mesmas variáveis de contexto, as utilizem por meio do mesmo controle de acesso concorrente, garantindo a integridade dos dados em tempo de execução. Este serviço é útil tanto para o servidor de autenticação quanto para o gerente de contexto implícito.
- Vários arquivos de definição foram adicionados ao arquivo de cabeçalho da classe (`gingaNcl/gingancl-cpp/include/adaptation/context/PresentationContext.h`). Estes arquivos de definição são necessários para o uso de sockets.

Além disso, as seguintes modificações foram efetuadas na classe *ContextManager* para que ela suportasse os serviços do servidor de autenticação:

- Foi adicionada à classe o serviço *addUser*() que permite a criação de um novo usuário dadas suas informações cadastrais. O serviço *addUser*() originalmente suportado pela classe permanece inalterado e é utilizado pelo novo serviço adicionado. O novo serviço gera identificadores automaticamente e impede a inserção de um usuário com identificador 0 (reservado para o usuário “default”).
- Foi adicionada à classe o serviço *removeUser*() que permite a remoção de um usuário que não o usuário “default”.
- Foi adicionada à classe o serviço *getUserIds*() que retorna uma lista dos identificadores dos usuários mapeados para as suas informações na área de contexto. Este serviço é particularmente útil para o gerente de contexto implícito.

Note que as mudanças na classe *ContextManager* implicaram igualmente em mudanças no arquivo de cabeçalho `gingaNcl/gingacc-cpp/gingacc-contextmanager/include/ContextManager.h` e no arquivo de interfaces `gingaNcl/gingacc-cpp/gingacc-contextmanager/include/IContextManager.h`.

O segundo componente do módulo de identificação explícita, a interface gráfica de acesso, é implementado através das seguintes aplicações NCLua:

- *indice.lua*: módulo responsável pelo menu de autenticação do usuário. Utiliza as bibliotecas *libFuncoes.lua*, *libTecladoAlfa.lua* e *libTecladoSettings.lua* para efetuar ações do menu (cadastros, remoções de usuário, login e mudança de senha). Ao ser invocado por uma página NCL, através do pressionamento da tecla "*M"*, exibe o menu e pára a exibição das propagandas. Quando um "M" é pressionado novamente, o menu é apagado e as propagandas são novamente habilitadas.

- *libFuncoes.lua*: biblioteca de funções do menu requisitadas pelo script *indice.lua*

- *libTecladoAlfa.lua*: biblioteca responsável pela interface do menu e teclado virtual.

- *libTecladoSettings.lua*: biblioteca responsável por fazer a comunicação com o Ginga, referente a operações de cadastro e verificação de usuários existentes.

Apêndice I: Interface de Acesso ao Autenticador

Durante a exibição da programação, a qualquer momento, o usuário pode invocar o serviço de autenticação para se identificar junto à TV. Para tanto, ele deve pressionar a tecla "M" (de menu)[25]. Ao invocar o serviço, um menu como o exibido na **Figura 13**, é exibido.

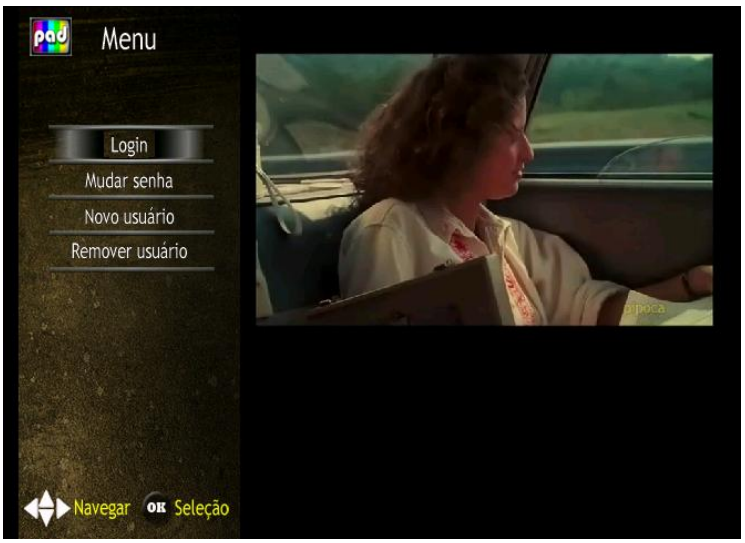


Figura 13: Menu inicial

Através deste menu, o usuário pode solicitar os serviços de autenticação na TV (opção login), alteração de senha (opção Mudar Senha), inclusão de novo usuário (opção Novo Usuário) e remoção de usuário (opção Remover Usuário). Para selecionar qualquer das opções, o usuário deve pressionar ENTER. Por exemplo, para ter acesso à opção de autenticação, o usuário deve pressionar ENTER sobre *Login*. Ao fazer, isto ele terá acesso a uma caixa de diálogo solicitando seu Nome e Senha, conforme **Figura 14**.

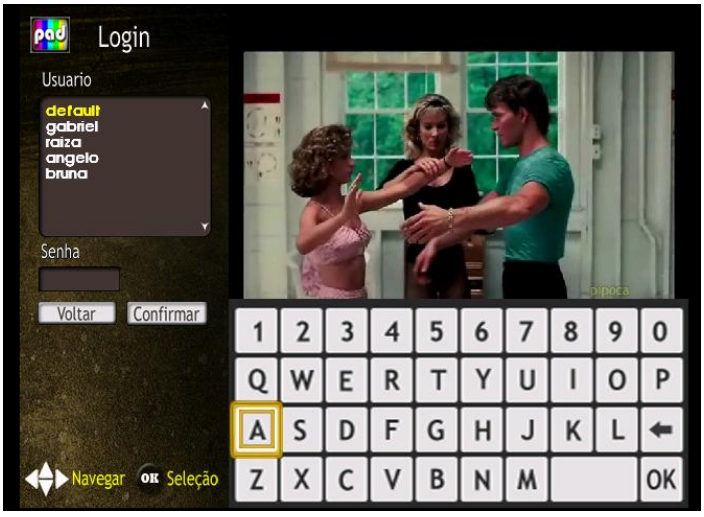


Figura 14: Autenticação do usuário

[25] Como este texto se refere a um protótipo rodando em um computador e não em uma TV, todas as interações do usuário com o Ginga são realizadas por meio de um teclado.

Para digitar o valor de um campo o usuário deve pressionar ENTER quando o campo desejado estiver com o foco (o que é indicado por um retângulo amarelo). Para trocar o foco, o usuário pode usar as setas direcionais. Campos podem permitir a edição direta de texto (*caixa de texto*) ou a seleção de uma entre uma série de opções (*menu*). Ao pressionar ENTER em um campo de menu, tal como o nome do usuário, este deve usar as setas direcionais para navegar pelas opções disponíveis e ENTER para escolher uma delas.

Ao pressionar ENTER em uma caixa de texto, o teclado virtual é invocado, conforme **Figura 15**. Para navegar pelo teclado, o usuário pode usar as setas direcionais e selecioná-las com ENTER. O teclado oferece teclas alfa-numéricas, uma tecla para apagar o último símbolo digitado e a tecla OK para concluir o uso do teclado. Ao concluir a digitação de dados, o usuário pode pressionar ENTER no botão "Confirmar" para enviar os dados para TV. Para cancelar a operação, o usuário deve pressionar o botão "Voltar".

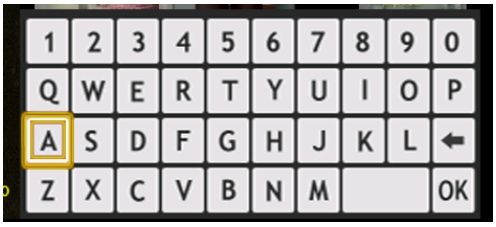


Figura 15: Teclado virtual

Selecionando a opção Mudar Senha do menu inicial, o usuário tem acesso a uma caixa de diálogo solicitando seu nome, senha antiga e senha nova, conforme **Figura 16**.

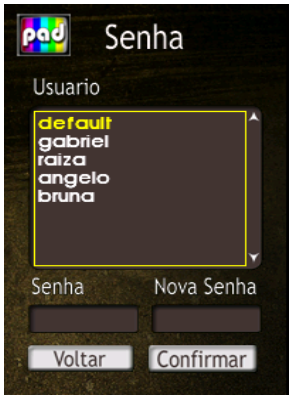


Figura 16: Alteração de senha

Selecionado a opção Novo Usuario, ele terá acesso a uma caixa de diálogo solicitando seu nome, senha, idade, sexo e código postal, conforme **Figura 17**. Note que para escolher um sexo, o usuário deve pressionar ENTER sobre a opção de sexo masculino (M) ou feminino (F).

Figura 17: Inclusão de usuário

Selecionado a opção Remover Usuario, o usuário terá acesso a uma caixa de diálogo solicitando seu nome e senha, conforme **Figura 18**. Ao pressionar o botão "Remover", o usuário será removido da lista de usuários da TV.

Figura 18: Remoção de usuário

Ao concluir uma operação, uma mensagem deve ser exibida indicando sucesso ou não, conforme exibido na **Figura 19**. Para voltar ao menu principal, o usuário deve pressionar ENTER. Então será exibido o menu, conforme **Figura 13**.

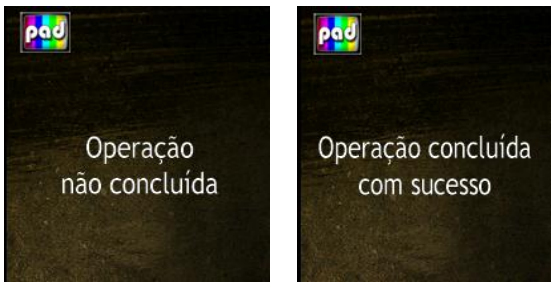


Figura 19: Mensagens de aviso

Apêndice II: profileMonitor

Ao executar o programa profileMonitor, é exibida a janela de monitoração apresentada na **Figura 20**.

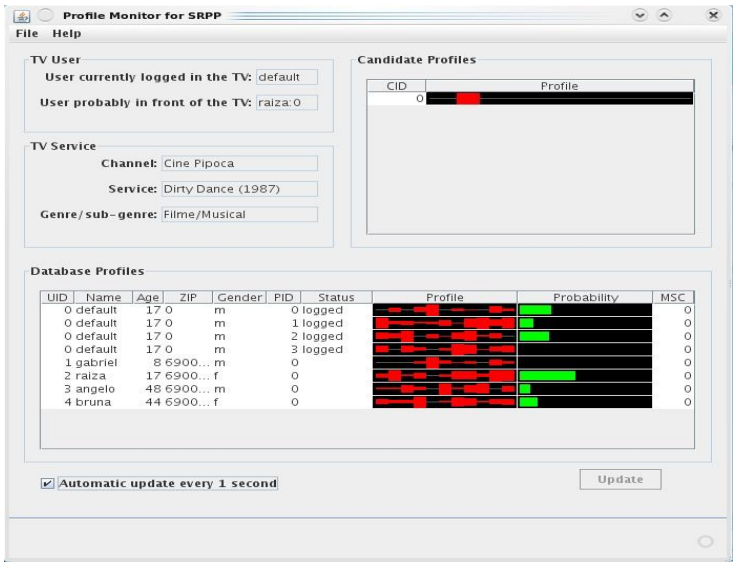


Figura 20: Janela de monitoração do SRPP

Através desta janela, é possível monitorar o funcionamento do Ginga enquanto o sistema de publicidade personalizada é executado. Em particular, são monitorados:

- Informações sobre o usuário atualmente autenticado (quadro *TV User*);
- Informações sobre o usuário (e seu perfil) que o Ginga acredita estar assistindo a TV (quadro *TV User*);
- O serviço em execução no Ginga (quadro *TV Service*, com informações sobre canal, serviço, gênero e sub-gênero de programação);
- Os perfis de usuários (candidatos) criados pelo Ginga, que poderiam estar assistindo a TV (quadro Candidate Profiles);
- Os perfis dos usuários (quadro *TV Profiles*). As informações dos perfis são identificador de usuário, nome, idade, CEP e gênero, bem como o identificador do perfil, seu estado atual (logado ou não), o valor dos atributos no perfil, a probabilidade de que este perfil seja o perfil do usuário atualmente assistindo a TV e o candidato que mais se assemelhou a este perfil em particular (MSC - *most similar candidate*).

Os perfis dos usuários são apresentados através de uma barra colorida, segmentada em tantos quantos forem os subgêneros de programação. Cada segmento na barra indica a importância de um diferente subgênero de programação para o usuário, de tal forma que quanto maior for a área colorida naquele segmento, maior é o tempo que o usuário passou assistindo a um conteúdo daquele sub-gênero. Por exemplo, a **Figura 21** exibe o perfil 0 (PID 0) do usuário 0 (UID 0). Note que este usuário tem maior interesse por programação infantil. Seus maiores interesses, em seguida, são música, romance e comédia. Ele tem ainda interesses menores por novela, filmes e séries.

As probabilidades dos perfis também são representadas por barras coloridas. Quanto maior a barra, maior a probabilidade correspondente. Para o mesmo usuário da **Figura 21**, notamos que a sua probabilidade de ser o usuário atual, conforme o Ginga, é de cerca de 0,2. Este usuário é mais similar ao candidato interno 0.

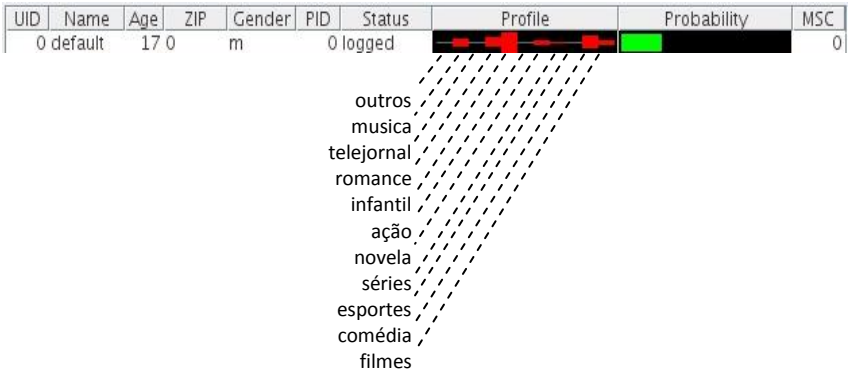


Figura 21: Perfil de um usuário

Para observar o estado atual da máquina Ginga (atualização isolada), deve ser pressionado o botão Update. Se a caixa de checagem na parte inferior esquerda da janela for selecionada, atualizações serão realizadas automaticamente a cada um segundo.

Para selecionar um diferente IP para o servidor de autenticação (a máquina onde o Ginga está em execução), deve-se usar a caixa de diálogo *Settings...*, acessada através da opção *File/Setings...* no menu principal.