Um Guia Eletrônico de Programação Baseado em Serviços Web e na Plataforma Android

Carlos Eduardo Ferreira Marins
Laboratory of Advanced Web Systems – UFMA
Av. dos Portugueses, Campus do Bacanga
São Luís/MA – 65085-580 - Brasil
edwardmarins@gmail.com

Mário Meireles Teixeira
Departamento de Informática – UFMA
Av. dos Portugueses, Campus do Bacanga
São Luís/MA – 65085-580 - Brasil
mario@deinf.ufma.br

ABSTRACT
The large volume of content provided by digital TV motivates the development of solutions that facilitate the choices of the viewers. This paper presents the architecture of an electronic programming guide based on web services using the REST architectural style. We also present the development of EPGdroid, an application on the Android platform, which consumes resources of the web service and provides a friendly interface to the end user.

Keywords
Web Services, SOA, REST, EPG, Android, Mobile Devices.

1. INTRODUÇÃO

Com a implantação da TV digital no Brasil tornaram-se muito bem-vindos esforços de pesquisa e produção de softwares que forneçam suporte ou ampliem as possibilidades dessa tecnologia.

Uma extensa variedade de programas e serviços vêm sendo disponibilizados por meio da TV Digital. A grande quantidade de conteúdos digitais disponíveis motiva a criação de aplicações que sejam capazes de coletar informações sobre os programas e serviços e apresentá-las de maneira organizada [5]. Nesse contexto, é essencial a existência de aplicações que apresentem para os usuários as diferentes opções de conteúdo oferecidas, de modo a auxiliar sua escolha. Essas aplicações são chamadas de Guias Eletrônicos de Programação [6].

Este artigo tem com objetivo o desenvolvimento de uma arquitetura interoperável, baseada em serviços web RESTful, que permita disponibilizar a programação atualizada das principais emissoras de TV, de modo que as mesmas possam ser facilmente consumidas por aplicações clientes desenvolvidas em diversas linguagens de programação e executando sobre plataformas heterogêneas.

2. SERVIÇOS WEB

Um Web Service, ou serviço web, é definido pelo W3C como sendo uma aplicação de software identificada por um URI, cujas interfaces e ligações são capazes de serem definidas, descritas, e descobertas como artefatos XML [8]. Um serviço web suporta interações diretas com outros agentes de software usando mensagens baseadas em XML trocadas via protocolos da Internet.

Dentre as principais abordagens para a utilização de serviços web destacam-se a Arquitetura Orientada a Serviços e o estilo arquitetural REST.

2.1 Arquitetura Orientada a Serviços

A Arquitetura Orientada a Serviços está baseada na relação entre um provedor, que oferece recursos e funcionalidades sob a forma de serviços web, e um consumidor de serviços, que se trata de um componente de software interessado em associar-se aos serviços. Com o propósito de facilitar essa associação, os serviços devem possuir uma especificação formal de sua interface e dos procedimentos necessários para que os consumidores de serviços possam se conectar a ele. Essa descrição é formalizada utilizando-se o padrão WSDL e o provedor de serviços pode, eventualmente, publicá-la em um registro de serviços, conforme ilustrado na Figura 1.

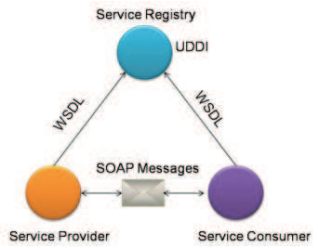

Figura 1: Estrutura da Arquitetura Orientada a Serviços

Um registro de serviços facilita a descoberta de serviços e a aquisição de informações de uso, provendo uma maneira uniforme de o consumidor descobrir quais serviços disponíveis estão aptos a satisfazer seus interesses. Este mecanismo uniforme é alcançado através do padrão UDDI.

Uma vez descoberto o serviço, o consumidor pode associar-se a ele para utilizar os recursos, dados ou funcionalidades fornecidas. Para isso, tanto provedor quanto consumidor de serviços trocam mensagens padronizadas através do protocolo SOAP.

A seqüência publicar/descobrir/associar e os elementos provedor, consumidor e registro de serviços constituem a essência da arquitetura orientada a serviços.

2.2 Estilo Arquitetural REST

Representational State Transfer (REST) é um estilo arquitetural híbrido para sistemas hipermídia distribuídos, derivado de vários

WebMedia '12, Oct 15–18, 2012, São Paulo, SP, Brazil.

estilos arquiteturais de software em rede [3], que define um conjunto de princípios que podem ser aplicados na construção de sistemas com uma Arquitetura Orientada a Recursos (ROA). Sistemas construídos segundo os princípios REST são chamados aplicativos *RESTful*.

REST está baseado no conceito de recursos que são identificados e disponibilizados através de URIs. Essas URIs são acessíveis a partir de links que retornam as representações dos recursos. Quando o cliente acessa uma dessas representações ele passa a ocupar um novo estado em relação à aplicação como um todo. Cada representação pode conter links para outros recursos, o que permite ao cliente navegar pela aplicação a partir da transição entre estados, daí o termo *Representational State Transfer*.

Também vale destacar a utilização de uma interface uniforme, ou seja, em REST o conjunto de métodos para cada elemento do sistema é conhecido e padronizado, sendo que a cada execução de um método, sua semântica é visível.

3. PLATAFORMA ANDROID

Com a popularização massiva dos celulares e dispositivos móveis com cada vez maior poder de processamento, a computação demonstra uma tendência natural para aplicações baseadas em plataformas para ambientes móveis. Uma das mais recentes e bem sucedidas é a plataforma Android.

A plataforma Android foi lançada em 2008 pela Google, por meio do consórcio *Open Handset Alliance* e, segundo [1], é uma pilha de software para dispositivos móveis que inclui um sistema operacional, middleware e aplicações denominadas críticas. O Android SDK (*Software Development Kit*) fornece as ferramentas e APIs (*Application Programming Interfaces*) necessárias para começar a desenvolver aplicações na plataforma Android usando a linguagem de programação Java.

A plataforma Android é um ambiente para dispositivos móveis que inclui um sistema operacional baseado no kernel 2.6 do Linux, um ambiente rico para desenvolvedores, além de suporte às tecnologias presentes na maioria dos dispositivos móveis e funcionalidades convencionais de telefones celulares [7].

3.1 Arquitetura Android

A arquitetura da plataforma Android, representada na Figura 2, é organizada em camadas que possuem atribuições específicas e gerenciam seus próprios processos.

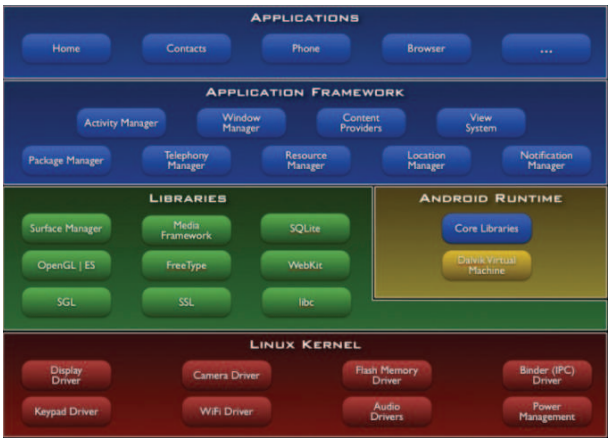

Figura 2: Arquitetura da plataforma Android.

A camada superior oferece um conjunto de aplicações de alto nível que são acessadas diretamente pelo usuário final, dentre as quais se destacam clientes de email, calendário, mapas, navegador, jogos e aplicações de terceiros. Logo abaixo da camada de aplicação, estão localizados os frameworks de aplicação, que incluem os programas que gerenciam as funções básicas do dispositivo como alocação de recursos, gerenciamento do telefone, oferecem informações sobre localização, notificações, alarmes etc.

Um pouco mais abaixo na pilha, Android inclui bibliotecas escritas em C/C++, bibliotecas de multimídia, visualização de camadas 2D e 3D, funções para navegadores web, funções de aceleradores de hardware, renderização 3D, funções para gráficos, fontes bitmap e vetorizadas e funções de acesso a banco de dados SQLite, dentre outras funcionalidades expostas aos desenvolvedores através dos frameworks de aplicação. No mesmo nível das bibliotecas está o ambiente de execução do Android (Android Runtime) que possui um conjunto de bibliotecas do núcleo Java e instâncias da máquina virtual Dalvik, uma máquina virtual própria, otimizada para dispositivos com poucos recursos computacionais e projetada especificamente para uso em ambientes embarcados.

Na base da pilha localiza-se o kernel do Linux, responsável pelo gerenciamento de memória, de processos e de E/S, pilha de rede, drivers de dispositivos, entre outros. É nesta camada que ocorre a abstração entre o hardware e o restante da pilha de software.

3.2 Principais Componentes

Existem quatro tipos principais de componentes Android: *activities*, *services*, *broadcast receivers* e *content providers*.

Atividades representam interfaces visuais da aplicação através das quais ocorre toda a interação com o usuário. Em geral, uma aplicação Android é composta por várias atividades, sendo que uma delas é carregada no inicio da aplicação, a qual é denominada atividade principal. A partir da atividade principal, outras atividades podem ser acionadas, cada uma com funções específicas, colaborando para a dinâmica do sistema. Apenas uma atividade pode estar ativa por vez, por isso, cada vez que uma nova atividade começa, a atividade anterior é interrompida e colocada em uma pilha (*back stack*).

A classe *Services* é usada para executar um serviço em segundo plano, geralmente vinculado a algum processo que deve ser executado por tempo indeterminado e possui um alto consumo de recursos, memória e unidade central de processamento [2].

Broadcast Receivers constituem uma implementação Android para um mecanismo produtor/consumidor, ou mais precisamente, um padrão do tipo observador. Através desse mecanismo uma aplicação pode registrar um receptor associado a um evento, de modo que no momento em que o evento ocorre o método *onReceive*() é disparado.

Por padrão, o Android executa cada aplicativo em sua própria *sandbox* para que todos os dados que pertençam a um aplicativo sejam totalmente isolados de outros aplicativos no sistema [4]. Os *content providers* (provedores de conteúdo) provém um nível de abstração para que qualquer dado guardado no dispositivo seja acessível por mais de uma aplicação [2].

4. ARQUITETURA PROPOSTA

Este trabalho descreve o desenvolvimento e a arquitetura de um serviço de guia eletrônico de programação, que adota uma abordagem de sistemas distribuídos baseada em serviços web que atendem às especificações do estilo arquitetural REST, e um cliente Android, responsável por fornecer uma interface amigável ao usuário final.

A partir da utilização do aplicativo os usuários são capazes de escolher um canal e visualizar sua programação diária, navegar na programação transmitida no momento corrente, podendo escolher um programa qualquer para visualizar seus detalhes e realizar uma pesquisa por programas de acordo com palavras-chave.

A arquitetura desenvolvida baseia-se nos componentes mostrados na Figura 3, os quais são detalhados a seguir.

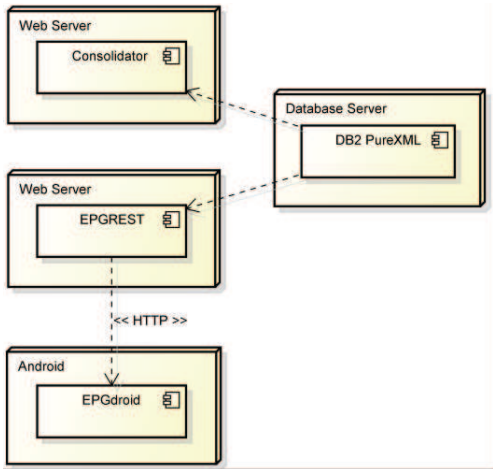

Figura 3: Arquitetura Proposta.

4.1 Componente Consolidator

Normalmente, as informações sobre programas e serviços são enviadas, por radiodifusão ou sob demanda, multiplexadas com outros tipos de dados como, por exemplo, áudio e vídeo [5]. No entanto, quando se deseja obter essas informações fora do contexto das televisões e decodificadores é necessária a utilização de fontes de dados alternativas, geralmente disponibilizadas na Web. Para este fim, em português, destacam-se como provedores de informações de programação de TV a tv.sapo.pt [1] e a Revista Eletrônica[2].

A Revista Eletrônica disponibiliza essas informações através do download de arquivos no formato XMLTV, que é um formato padronizado baseado em XML usado para obtenção, manipulação e compartilhamento de informações de programação de TV de vários canais.

O componente *Consolidator* tem por objetivo transformar estes arquivos em informação utilizável pelo serviço *RESTful*.

Para consolidar as informações e permitir o acesso às mesmas por outros componentes, o *Consolidator* efetua o download dos arquivos XMLTV, processando-os para os moldes específicos da aplicação e armazena estes arquivos em uma base de dados XML gerenciada pelo *SGBD DB2 PureXML*.

[1] http://tv.sapo.pt

[2] http://www.revistaeletronica.com.br

O *PureXML* é a nova tecnologia inserida no DB2 versão 9 permitindo que o banco de dados armazene documentos XML bem formados em colunas do tipo de dados XML, mantendo sua forma hierárquica. Deste modo não é mais necessário mapear documentos XML para o banco de dados ou armazená-los em objetos grandes.

4.2 Componente EPGREST

O componente EPGREST tem como função oferecer uma interface uniforme para acesso a informações de programação de TV, retornando seus resultados em um formato portátil, baseado em XML e facilmente interpretado por praticamente todas as linguagens de programação modernas.

EPGREST foi desenvolvido a partir dos princípios arquiteturais REST, utilizando serviços web para implementar funcionalidades bem definidas e independentes. Devido a estas características, os serviços podem ser consumidos por clientes em diferentes aplicações e processos de negócios, utilizando-se variadas plataformas e tecnologias.

Para a aplicação do guia eletrônico de programação foram definidos dois serviços básicos, *channel* e EPG, ambos possuindo apenas métodos GET.

O serviço *channel* possui somente um recurso, que disponibiliza uma lista com todos os canais que possuem suas programações registradas. O serviço *channel* é identificado pela URL */channel*.

O serviço EPG oferece um conjunto de subrecursos, representando documentos XMLTV, relacionados à programação de TV propriamente dita. O primeiro subrecurso recebe como parâmetro um código de canal e uma data, disponibilizando a programação inteira do canal na data especificada. O segundo subrecurso oferece um documento XMLTV contendo os programas, de todos os canais, que estão sendo transmitidos na data e hora atuais. Um terceiro subrecurso realiza uma busca na programação baseando-se em uma string passada como parâmetro. Outros subrecursos, embora não utilizados pelo cliente EPGdroid, estão disponíveis no serviço, entre os quais: a programação de um programa específico em um intervalo de datas, a programação de um dado canal em um intervalo de datas, todos os programas de uma determinada categoria em um intervalo de datas. As URLs para acesso aos recursos citados são, respectivamente:

- /EPG/{canal}/{data}
- /EPG/now
- /EPG/search/{palavra-chave}
- /EPG/program/{id_programa}/{data1}/{data2}
- /EPG/{canal}/{data1}/{data2}
- /EPG/category/{categoria}/{data1}/{data2}

4.3 Componente EPGdroid

Visto que o componente EPGREST foi desenvolvido como um serviço web *RESTful*, uma série de aplicações clientes, dos mais variados tipos, podem ser desenvolvidas a fim de consumir os recursos disponibilizados. Neste trabalho optou-se pela construção de um aplicativo sobre a plataforma Android, denominado EPGdroid.

O EPGdroid fornece um conjunto de interfaces para que o usuário possa ter acesso aos serviços do componente EPGREST. Estas interfaces são construídas usando os componentes básicos da plataforma Android descritos na Seção 3.2, entre outros, relacionados conforme diagrama da Figura 4.

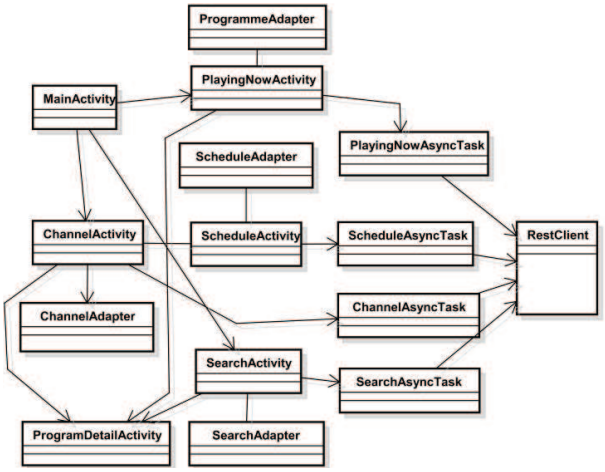

Figura 4: Diagrama de classes do EPGdroid.

A classe *MainActivity* é responsável pela navegação e acesso às funcionalidades providas pelo aplicativo. Ela utiliza o recurso *TabHost* da API Android para organizar o conteúdo da aplicação de uma forma intuitiva e prática, através do uso de abas. A partir de *MainActivity* três novas atividades podem ser acessadas: *ChannelActivity*, que exibe a lista dos canais disponíveis, PlayingNowActivity, que exibe a programação de cada canal no momento atual, e *SearchActivity*, que permite ao usuário efetuar uma busca na programação.

Outros elementos importantes do diagrama da Figura 4 são as classes que herdam de *AsyncTask*: *PlayingNowAsyncTask*, *ScheduleAsyncTask*, *ChannelAsyncTask* e *SearchAsyncTask*. Elas constituem um mecanismo apropriado para dar *feedback* ao usuário durante a execução de alguma operação que demande tempo.

Por fim, todo processamento relacionado ao serviço web foi encapsulado na classe *RestClient*. Esta classe é responsável por preparar e enviar as requisições HTTP e receber os resultados das solicitações.

Dentre as telas do EPGdroid, as que exibem os programas sendo transmitidos e a descrição de um programa são mostradas na Figura 5.

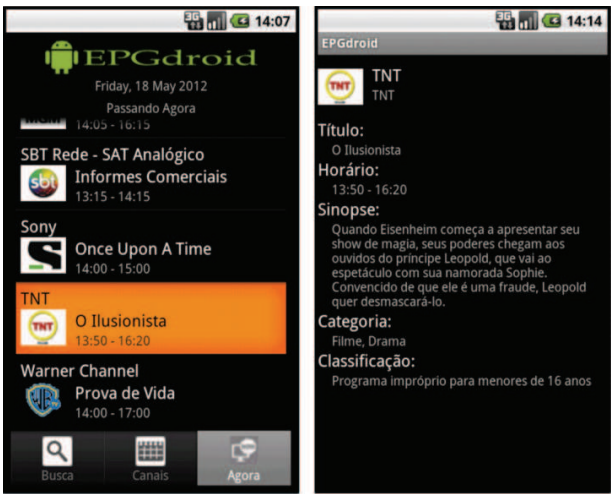

Figura 5: Telas do EPGdroid.

5. CONCLUSÃO

Este trabalho discutiu e implementou a arquitetura de um guia eletrônico de programação baseado em serviços web utilizando a abordagem REST, cuja interação com o usuário se dá por meio de um aplicativo desenvolvido sobre a plataforma Android.

Através deste trabalho pôde-se exemplificar como sistemas baseados em REST podem promover a dissociação entre as interfaces dos serviços e suas implementações, visto que os clientes precisam apenas descobrir essas interfaces e consumir os resultados disponibilizados sob o padrão XML, sem a necessidade de conhecer detalhes internos ao serviço.

Outro fator importante foi a utilização de um aplicativo Android interagindo com o serviço web *RESTful*, o que garante mobilidade e maior utilidade à aplicação, dada o aumento da preferência dos usuários por dispositivos móveis para a busca de informações relacionadas a consumo e entretenimento.

Como trabalhos futuros objetiva-se implementar um sistema de recomendação de programação baseado no perfil do usuário e realizar a integração com o IMDb (*Internet Movie Database*) de modo a aumentar o leque de informações sobre filmes e séries de TV, proporcionando maior interatividade ao usuário.

REFERÊNCIAS

[1] Android developers. User Interfaces. Disponível em: http://developer.android.com/guide/topics/ui/index.html. acessado em: 01 Maio 2012.

[2] Ferreira, G. D. Um middleware declarativo na plataforma AndroidTM para o Sistema Brasileiro de Televisão Digital (SBTVD). Dissertação, Universidade Federal do Espírito Santo, Vitória, ES, 2010.

[3] Fielding, R. T. Architectural Styles and The Design of Network-based Software Architectures. PhD thesis, University of California, Irvine, 2000.

[4] Gargenta, M. Learning Android. O'Reilly, Sebastopol, 2010.

[5] Maia, P. P. C., Leite, J., Batista, T. MyPersonal-EPG: Um EPG Personalizável e com Suporte à Recomendações. in WEBMEDIA (Belo Horizonte, 2010).

[6] Oliveira, F. N. B. de. Aplicação Adaptativa de Guia Eletrônico utilizando o Ginga-NCL. Dissertação, Pontifícia Universidade Católica Do Rio De Janeiro - Puc-Rio, Rio de Janeiro, RJ, 2010.

[7] Sousa, T. N. de. Desenvolvimento de uma Rede P2P para a Plataforma Android. Monografia, Universidade Federal do Maranhão, São Luis, Maranhão, 2011.

[8] W3C. Disponível em: http://www.W3C.org. Acessado em: 01 Maio 2012.