UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
ESPECIALIZAÇÃO EM TECNOLOGIA JAVA

NAJIB RHAFAEL ALVES EL ALAM

GESTÃO FINANCEIRA PESSOAL UTILIZANDO ANDROID

MONOGRAFIA DE ESPECIALIZAÇÃO

CURITIBA
2011

NAJIB RHAFAEL ALVES EL ALAM

GESTÃO FINANCEIRA PESSOAL UTILIZANDO ANDROID

Monografia apresentada ao Curso de Especialização em Tecnologia Java do Departamento Acadêmico de Informática – DAINF - da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Especialista.

Orientadores: Prof. Caio Nakashima e Prof. Paulo Bordin

CURITIBA

2011

RESUMO

EL ALAM, Najib. Gestão Financeira Pessoal Utilizando Android. 2011. 40 f. Monografia (Especialização em Tecnologia Java) – Programa de Pós-Graduação em Tecnologia, Universidade Tecnológica Federal do Paraná – UTFPR. Curitiba, 2011.

O objeto de estudo deste trabalho é o desenvolvimento de uma aplicação de Gestão Financeira Pessoal para dispositivos Android, tomando como base uma aplicação *web* já existente, comunicando-se com esta aplicação através de um *Web Service*, permitindo que o usuário possa gerenciar suas finanças pessoais a partir de dispositivos Android.

Palavras-chave: Tecnologia Java, Android, Web Services, Finanças Pessoais.

ABSTRACT

EL ALAM, Najib. Personal Finance Management Using Android. 2011. 40 f. Monografia (Especialização em Tecnologia Java) – Programa de Pós-Graduação em Tecnologia, Universidade Tecnológica Federal do Paraná – UTFPR. Curitiba, 2011.

The object of this work is the development of a Personal Financial Management application for Android devices, based on an existing web application, communicating with this application via a Web Service, allowing the user to manage their personal finances from Android devices.

Keywords: Java Technology, Android, Web Services, Personal Finances.

LISTA DE FIGURAS

LISTA DE QUADROS

SUMÁRIO

1 INTRODUÇÃO

Há vários meios possíveis de se obter acesso a informações geradas pelos diversos sistemas computacionais existentes. Na atualidade esses sistemas de informações geram centenas de terabytes de dados porém uma pequena fração desses dados viram informação concreta e realmente utilizada para fins de consulta ou tomada de decisão. Uma das justificativas para esses fatos são custos elevados para extrair os dados, transformá-los em informação útil e torná-los visíveis nas diversas mídias hoje disponíveis, tal como Internet, celulares, tablets e outros dispositivos móveis.

Em conjunto com esses diversos meios de acesso a informações, estão as tecnologias utilizadas para gerar e manter esses dados, dentre elas podem ser destacadas Java, PHP e ASP.NET. Tanta tecnologia envolvida gera também de forma proporcional implementações e soluções distintas para acesso aos dados.

A partir de um caso real de informações disponíveis em um sistema de gerenciamento de finanças pessoais, verificou-se que é mais cômodo para o usuário ter disponível em seu dispositivo móvel a informação da posição atualizada das suas finanças, podendo auxiliá-lo na escolha do melhor momento para realizar uma compra, planejar uma viagem ou até mesmo aproveitar uma promoção a cerca das oportunidades imediatas, hoje em dia geradas pelos diversos mecanismos de marketing. Atualmente, a única forma de acesso disponível para visualização ou entrada de dados desse sistema de finanças é pelo *website* na Internet. Essa forma de acesso atende as necessidades básicas de seus usuários porém, o fato do usuário realizar uma compra e somente depois, quando estiver em casa ou no trabalho, poder realizar a entrada dessa informação no sistema, acaba por acarretar a ausência desse registro por esquecimento ou até mesmo falta de disciplina no controle das finanças pessoais, gerando descontrole e falta de estímulo ao hábito de manter em dia as informações financeiras.

Portanto, o objetivo geral desde trabalho é demonstrar como é possível construir uma ferramenta de gestão de finanças pessoais utilizando tecnologia Android, acessando um *Web Service*, possibilitando dessa forma que haja uma interação mais dinâmica do usuário com as suas informações financeiras, agregando maior controle, comodidade, precisão e agilidade na disposição dessas

informações.

Como objetivo específico, o desenvolvimento da aplicação irá auxiliar os usuários na gestão das finanças pessoais, através do uso das funcionalidades implementadas, bem como técnicas e métodos de comunicação entre um dispositivo Android e o Sistema de Controle de Finanças Pessoais, utilizando *Web Service* como meio de acesso, trazendo como resultado a visualização instantânea dessas informações.

Para tanto, no segundo capítulo serão evidenciados os detalhes do tema proposto, conceitos de *Web Services* e Dispositivos Android. O terceiro capítulo será dedicado às especificações, à metodologia e às técnicas utilizadas para construir o sistema proposto e a camada de comunicação com o *Web Service*. No quarto capítulo será demonstrada a validação do sistema proposto, os detalhes do ambiente de teste e os resultados obtidos.

O presente trabalho se justifica pela demonstração de como uma aplicação desenvolvida para dispositivos Android, pode melhorar a experiência do usuário na gestão das finanças pessoais, facilitando o uso do Sistema de Controle de Finanças Pessoais, consolidando também técnicas utilizadas para utilização de uma camada de aplicação voltada ao acesso a *Web Services* em dispositivos Android.

2 TEMA PROPOSTO: CONCEITOS

Este capítulo será dedicado a embasar o tema proposto, abrangendo a descrição do ambiente de uso do Sistema de Controle de Finanças Pessoais, denominado No Meu Bolso e demonstrar o problema a ser resolvido. Também serão mostrados os conteúdos referentes às áreas envolvidas nesse trabalho, abrangendo o referencial teórico de *Web Services,* Dispositivos Android e a biblioteca KSOAP2.

2.1 TEMA PROPOSTO

O tema proposto foi idealizado após a necessidade latente de utilização de uma forma dinâmica e instantânea que possibilitasse ao usuário ter acesso a informações gerenciadas pelo Controle de Finanças Pessoais denominado No Meu Bolso, armazenadas em um banco de dados que está situado em um servidor na Internet.

A estrutura atual de utilização do Sistema de Controle de Finanças Pessoais No Meu Bolso, permite que um usuário se cadastre pelo *website* e a partir de então possa realizar a gestão dos seus recursos financeiros utilizando as diversas funcionalidades lá disponibilizadas.

Esse sistema foi desenvolvido em PHP 5.2.9, utilizando banco de dados Mysql 5, biblioteca *Javascript Jquery* para tratar as requisições AJAX. Utiliza ainda camada *Web Service* denominada NuSoap 1.123 que implementa os padrões SOAP e WSDL, que serão detalhados mais à frente.

Utilizando esse sistema *on-line* o usuário dispõe de funcionalidades que permitem inclusão, alteração ou remoção de um lançamento, bem como cadastro de contas para poder facilitar a classificação dos recursos financeiros. Também faz parte das funcionalidades a importação de arquivos no formato .ofx (*Open Financial Exchange*) que é um padrão largamente utilizados por praticamente todas as instituições financeiras do mundo. O usuário conta ainda com gráficos que demonstram de forma imediata a distribuição percentual dos valores recebidos e gastos, tornando transparente e instantâneo o entendimento da situação em que ele se encontra.

Apesar dessas funcionalidades disponíveis no *website*, ter os principais recursos em um dispositivo móvel traria maior comodidade e rapidez na gestão dessas informações. Dessa forma, foi idealizada a integração desses dois mundos,

web e móvel utilizando-se de transações disponibilizadas por um *Web Service* já construído e ativo.

Conforme afirmam especialistas do mercado da Internet, está próximo o fim dos *desktops*, dando lugar a *smartphones* e outros dispositivos móveis.

> Também o Google considera que o fim dos desktops pode estar mais próximo do que se imagina. No ano passado, durante a conferência Digital Landscapes, realizada em Dublin (Irlanda), o diretor de operações da gigante das buscas na Europa, John Herlihy, disse que os desktops deveriam se tornar irrelevantes em três anos. Para ele, o clássico conjunto formado por grandes gabinetes de metal, monitor, teclado, mouse e fios espalhados, seriam substituídos num futuro bem próximo por smartphones e outros dispositivos móveis, como o principal meio de diversão e entretenimento. Para ele, uma peça-chave para que essa mudança de comportamento se consolidasse definitivamente seria o amadurecimento de ferramentas de computação em nuvem, que deixaria a tarefa de processar uma série de atividades parrudas com servidores dedicados. “Dispositivos móveis vão permitir que o conhecimento seja acessível a todos", ressaltou na ocasião. (ESTADO DE MINAS, 2011).

Sendo assim, o sistema de Gestão Financeira Pessoal para dispositivos Android está alinhado à tendência mundial de mobilidade e irá demonstrar como essa troca de informações pode ocorrer de forma ágil, transparente e independente de tecnologia, explorando recursos presentes nas tecnologias empregadas em dispositivos Android.

2.2 WEB SERVICES

Web Services são componentes que permitem às aplicações enviar e receber dados em formato XML. Cada aplicação pode ter sua própria linguagem, que é traduzida para uma linguagem universal, o formato *eXtendable Markup Language*(XML) (SANTOS, 2007, p. 90).

Esta tecnologia possibilita a comunicação de dados com a utilização de arquivos XML. Segundo Reverbel (2006), XML é uma linguagem para representação de dados que é extensível e naturalmente independente de plataforma, além de ser amplamente utilizada pela indústria. Em *Web Services*, tanto a descrição de um serviço, quanto a comunicação entre serviços é feita usando XML.

É importante salientar que um dos motivos para tornar o uso de *Web Services* tão atrativo é o fato desse modelo ter como base tecnologias que são padrões na área de tecnologia da informação, em particular XML e HTTP.

O funcionamento de um *Web Service* depende de diversas tecnologias que juntas o possibilita disponibilizar um serviço requerido pelo usuário com rapidez e segurança. Nesta seção são exibidos os componentes de um *Web Service* bem como o papel de cada um na composição de um serviço.

2.2.1 O protocolo SOAP

O protocolo SOAP – *Simple Object Access Protocol* é um padrão de comunicação utilizado em *Web Services*. Esse protocolo visa garantir padronização na forma de troca das mensagens e é baseado em XML. A figura 1 mostra os elementos que compõem uma mensagem SOAP.

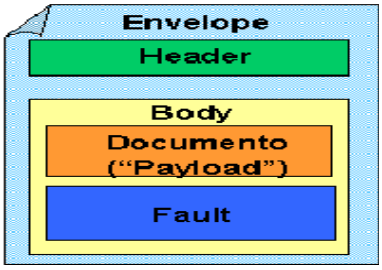

FIGURA 1 - ESTRUTURA DE UMA MENSAGEM SOAP

Toda mensagem SOAP deve conter o *Envelope*, que é o elemento raiz do documento XML. O envelope pode conter declaração de *namespaces* e também atributos adicionais como o que define o estilo de codificação, responsável por determinar como os dados são representados no XML.

O *Header* é um cabeçalho opcional que carrega informações adicionais, como por exemplo, se a mensagem deve ser processada por um determinado nó intermediário. É importante lembrar que, ao trafegar pela rede, a mensagem normalmente passa por diversos pontos intermediários, até alcançar o destino final. Quando utilizado, o *header* deve ser o primeiro item do envelope.

O *Body* é o corpo da mensagem. É um elemento obrigatório e contem o *payload* ou informação a ser transportada para o seu destino final. O elemento *body* pode conter um outro elemento opcional denominado *fault* ou falha, usado para carregar mensagens de status e erros retornados pelos "nós" ao processarem a mensagem. O quadro 1 representa o exemplo de uma chamada ao serviço No Meu Bolso e o envelope SOAP gerado.

```
<SOAP-ENV:Envelope

SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"

xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns9757="urn:server.getSaldoAnoMes">

<SOAP-ENV:Body>

   <ns9757:getSaldoAnoMes xmlns:ns9757="urn:server.getSaldoAnoMes">

       <idUsuario xsi:type="xsd:int">1</idUsuario>

      <anoMesReferencia xsi:type="xsd:string">2011/04</anoMesReferencia>

   </ns9757:getSaldoAnoMes>

</SOAP-ENV:Body>

</SOAP-ENV:Envelope>
```

QUADRO 1 - EXEMPLO ENVELOPE SOAP

2.2.2 WSDL

Web Service Description Language é uma linguagem baseada em XML utilizada para descrever um *Web Service*. Dessa forma, um cliente consegue saber qual o formato dos métodos a serem chamados, quais parâmetros são necessários. Nesse documento o *Web Service* define as suas interfaces, operações, esquemas de codificação entre outras informações.

Com base nessa descrição do serviço, a implementação do *Web Service* pode ser feita em qualquer linguagem de programação, mas normalmente são utilizadas linguagens que possuem integração com a Internet, como por exemplo Java Servlets, ASP ou PHP.

Quando o cliente deseja enviar uma mensagem para um *Web Service*, ele obtém a descrição do serviço, disponível no WSDL, em seguida constrói a mensagem, informando os tipos de dados, conforme consta na definição desse documento. Dessa forma a mensagem está pronta para ser enviada para o endereço onde o serviço está localizado, a fim de que possa ser processada. O *Web Service* quando recebe essa requisição, então valida-a conforme as informações contidas no documento WSDL. A partir de então, o serviço remoto sabe como tratar a mensagem, sabe como processá-la e como montar a resposta ao cliente.

A figura 2 fornece uma visão geral desse processo.

FIGURA 2 - APLICAÇÃO CLIENTE ACESSANDO SERVIÇO REMOTO

2.3 ANDROID

Android é uma plataforma de *software* de telefonia móvel, encabeçada pelo Google, e é a grande aposta desse mercado. Surgiu em novembro de 2007 através de um consórcio denominado *Open Handset Alliance* ou Aliança de Telefonia Móvel Aberta. Em pouco tempo vários aparelhos celulares já estavam utilizando esse sistema operacional que de certa forma tem como principal objetivo remover barreiras para o sucesso no desenvolvimento e venda de uma nova geração de softwares aplicativos de telefonia móvel (ROGERS, 2009, p. 2).

Basta olhar o cenário da concorrência entre os Celulares *SmartPhones* que antes estava sob completo domínio da Nokia, que apesar de ainda ser o maior fabricante de celulares, está perdendo terreno para dispositivos iPhone, BlackBerry e Android, provocando assim mudanças para uma perspectiva onde tudo aponta para o crescimento da utilização plataformas nas quais o grande diferencial passa a ser o leque de serviços e facilidades ofertadas.

Essa rapidez na disponibilização de conteúdo para esse mercado promissor, depende em muito da plataforma que está firmado um dispositivo. No caso dos dispositivos Android, por se tratar de uma plataforma em código aberto, a comunidade existente de desenvolvedores cria uma quantidade imensa de aplicações para uso coletivo, de forma gratuita ou paga, democratizando o acesso a

aplicações ou até mesmo fomentando o mercado de desenvolvimento de soluções baseadas em tal tecnologia (IDGNOW, 2010).

2.3.1 Diferencial

O Android é totalmente o oposto dos demais *smartphones* que utilizam software proprietários e relativamente fechados, como por exemplo Nokia série 60 com o sistema operacional Symbian ou o Windows Mobile da Microsoft. Nesses sistemas operacionais os programas não são código aberto, de modo que qualquer alteração se torna muito difícil, pois possuem detalhes como *frameworks* de criação de aplicações e multimídia, que são elementos proprietários. Em muito desses aparelhos o usuário não consegue sequer instalar uma nova aplicação, sob alegação do fabricante de que isso é necessário para preservar a integridade das redes, evitando a instalação e vírus ou programas de spam (ROGERS, 2009, p. 3).

2.3.2 Arquitetura

A arquitetura Android permite que aplicações sejam desenvolvidas dentro do conceito de reúso, possibilitando assim que uma aplicação seja incorporada a outra, como por exemplo o processo de envio de *e-mail*, no qual o usuário sinaliza que deseja enviar um *e-mail* e o próprio sistema operacional se encarrega de encontrar uma aplicação compatível com aquela tarefa, disponibilizando-a ao usuário. Isso é possível por meio de *Intents* ou intenções, que é a forma de comunicação entre processos adotada pelos dispositivos Android. É necessário que alguns conceitos da arquitetura sejam esclarecidos, antes de ser apresentada a solução para acesso a *Web Services*.

A figura 3 apresenta os principais elementos e camadas da arquitetura Android.

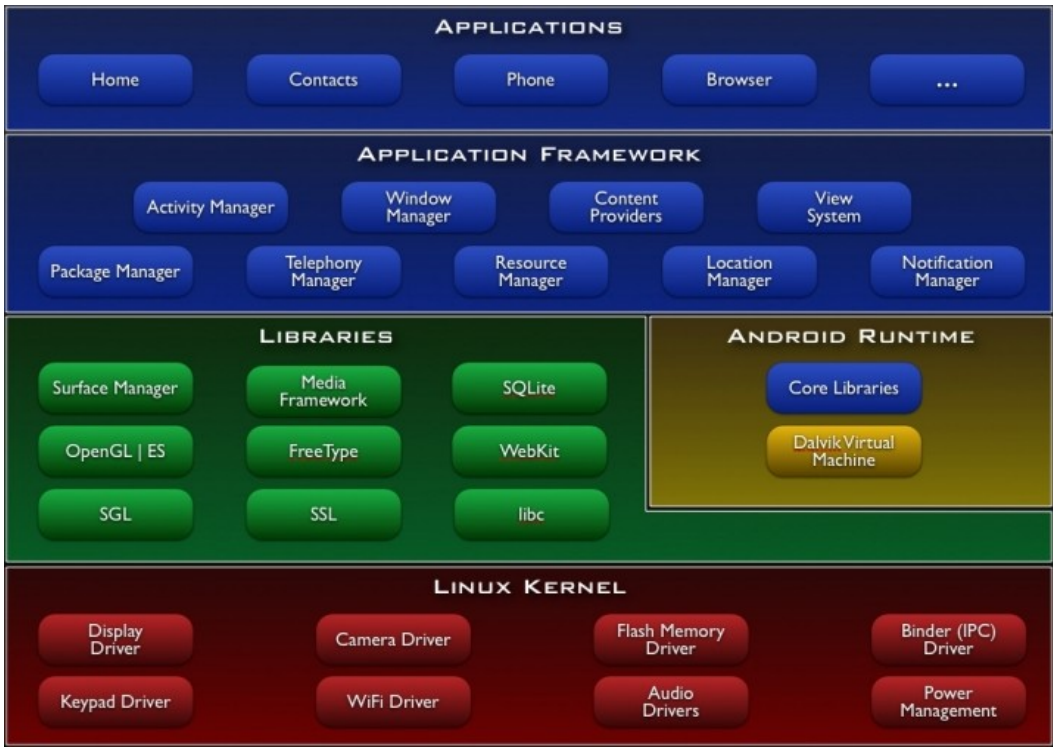

FIGURA 3 - ARQUITETURA ANDROID
FONTE: ANDROID DEVELOPERS

2.3.3 Entendendo cada camada

O Android é um conjunto de aplicativos para dispositivos móveis que inclui um sistema operacional, *middleware*, e aplicações chave, tal como gerenciador de contatos, navegadores, funções de telefonia, etc.

A seguir serão demonstradas as principais características de cada camada da arquitetura Android.

2.3.3.1 Aplicações

Android possui embarcado um conjunto de aplicações incluindo cliente de e-mail, programa para envio de mensagens SMS, calendário, mapas, navegadores, gerenciador de contatos, entre outras. Todas as aplicações são escritas usando a linguagem java de programação.

2.3.3.2 Framework de aplicação

A arquitetura das aplicações Android é voltada à simplicidade e reuso de componentes. Dessa forma, os desenvolvedores possuem acesso irrestrito às mesmas APIs usadas pelas aplicações centrais. As aplicações podem ser utilizadas

por outras aplicações, logicamente limitada as restrições de segurança. Esse mecanismo permite que componentes de aplicação possam ser substituídos pelo próprio usuário, como por exemplo o aplicativo de envio de mensagens SMS, que pode ser substituído por qualquer outro aplicativo que tenha a mesma finalidade, ficando a critério do usuário a escolha. Focando especificamente no reuso de componentes, podem-se destacar os dois componentes abaixo.

- Atividades (*Activity*) são trechos de código independentes, executados de acordo com a necessidade da aplicação, geralmente estão associadas a uma tela de entrada ou listagem de informações. A maior parte do código executável criado para o Android será executado no contexto de uma Atividade.
- Intenções (*Intents*) permitem que uma aplicação selecione uma Atividade tomando como base a ação desejada pelo usuário e também em dados que operam. Esse recurso possibilita por exemplo que ocorra a troca de dados entre atividades diferentes, sendo uma forma de comunicação entre processos.

2.3.3.3 Bibliotecas

Android conta com um conjunto de bibliotecas escritas em C/C++ que são utilizadas em vários componentes do sistema. Essa potencialidade está disponibilizada ao desenvolvedor através do *framework* de aplicação. Dentre as diversas bibliotecas disponíveis, destacam-se:

- Biblioteca de Mídia, que é responsável pela reprodução e gravação de áudio e vídeo nos formatos populares como AMR, ACC, MP3, MPEG4 e H.264, bem como imagens estáticas (fotos), incluindo JPG e PNG;
- SQLITE, um poderoso e leve banco de dados relacional que está disponível para todas as aplicações.

2.4 KSOAP2

Apesar das várias versões do SDK – *Software Development Toolkit* ou Kit de Desenvolvimento de Programas do Android conterem bibliotecas que disponibilizam vastos recursos para o programador, não há suporte algum à tecnologia de *Web Services*. Dessa forma quem necessitava consumir um serviço *web* poderia fazê-lo,

porém de forma muito trabalhosa utilizando conexão HTTP usando java.net ou bibliotecas org.apache.http, informando os dados de forma aberta e explícita na URL (de *Uniform Resource Locator*), em português Localizador-Padrão de Recursos e fazendo o *parser* do XML de forma manual. Uma solução possível seria aproveitar uma biblioteca existente, concebida inicialmente para Java ME, neste caso kXML-RPC e KSOAP.

kXML-RPC é uma implementação do XML-RPC para JME, protocolo esse que permite chamada de procedimentos remotos usando HTTP como transporte e XML como envelope. XML-RPC foi desenhado para ser o mais simples possível, enquanto possibilita a transmissão, processamento e retorno de estruturas complexas de dados.

KSOAP é um projeto de código aberto de uma biblioteca que proporciona a uma aplicação cliente acessar um *Web Service* SOAP em ambientes específicos de aplicações Java escritas em JME ou Applets. O projeto KSOAP2 é originado do KSOAP que é uma biblioteca utilizada para acesso a *Web Service* por aplicações Java ME desde 2001, entretanto esse projeto foi abandonado por volta de 2006. Com o lançamento da plataforma Android em 2007, verificou-se a necessidade de comunicação entre esses dispositivos e os serviços *web* legados (KSOAP PROJECT, 2011).

Originalmente a implementação do KSOAP2 para Android surgiu no grupo de desenvolvedores do Google, por Jorge Jimenez. Após várias versões o projeto teve sua manutenção encerrada quando por por volta de 2009/2010 Manfred Moser assumiu o projeto e lançou novas versões corretivas da biblioteca que atualmente está na versão 2.5.1 (KSOAP2-ANDROID, 2011).

3 METODOLOGIA DA CONSTRUÇÃO

Este capítulo tem como objetivo apresentar alguns detalhes referentes à construção do sistema proposto, destacando os caso de uso, a comunicação utilizando *Web Services* a partir de dispositivos Android bem como: linguagem de programação, plataforma de desenvolvimento, técnicas e metodologia.

3.1 TECNOLOGIA DE DESENVOLVIMENTO

O sistema foi desenvolvido utilizando as seguintes tecnologias, plataformas e componentes de hardware:

- Sistema Operacional Windows XP®;
- Ferramenta de desenvolvimento IDE Eclipse Gallileo® versão 3.5.0;
- Plugin ADT – *Android Development Toolkit* versão 8.0.1;
- Linguagem de programação, o Java versão 6 da Oracle®;
- Biblioteca ksoap2 android assembly versão 2.4;
- Smartphone LG GW620 com sistema operacional Android 2.2 Froyo.

3.2 ESPECIFICAÇÃO DO SISTEMA

O sistema proposto para prova de conceito implementa os seguintes casos de uso:

- **Realizar Login**: Trata da captura dos dados do usuário como e-mail e senha e realiza comunicação com o *Web Service* no sentido de validar os dados fornecidos, concedendo ou não o acesso ao sistema.
- **Listar Lançamentos**: Efetua a consulta dos lançamentos de entrada ou saída, tomando como base o período em mês e ano atuais.
- **Incluir lançamento**: Disponibiliza interface de tela para que o usuário possa realizar a inclusão de um novo lançamento de entrada ou saída.
- **Remover lançamento**: Efetua a remoção de um determinado lançamento selecionado pelo usuário na lista de entradas ou saídas.
- **Editar Lançamento**: Com base no lançamento selecionado, efetua a busca dos dados do lançamento pelo *Web Service* e disponibiliza essas informações para edição do usuário.
- **Mostrar Saldos**: Através do *Web Service*, efetua consulta dos saldos,

gerando demonstrativo com base no período em mês e ano selecionados.

- **Atualizar Preferências**: Captura a indicação do usuário para permanecer autenticado no sistema até a próxima troca de senha.
- **Realizar Operação Web Service**: Consiste em capturar os parâmetros informados pelos métodos que se utilizam desse serviço e realizar a invocação do método do *Web Service*, tratar esse retorno e então devolver o XML gerado como resultado. Este caso de uso é utilizado pelos demais casos de uso, exceto Atualizar Preferências.

Podemos verificar na Figura 4 o diagrama de caso de uso demonstrando a realização do login do usuário.

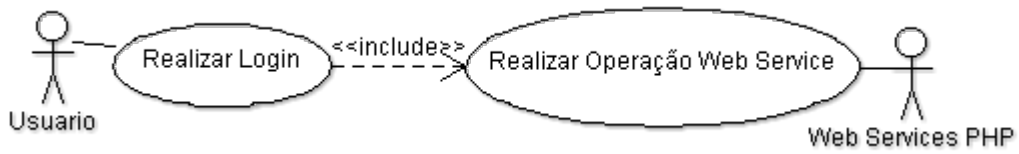

FIGURA 4 - DIAGRAMA DE CASO DE USO REALIZAR LOGIN

CONTRATO:

- Nome: realizar login.
- Ator Principal: Usuário.
- Ator de Suporte: *Web Service* PHP.
- Pré-condições: Que o usuário tenha realizado cadastro do e-mail, e senha pelo site, bem como tenha realizado a ativação da conta e esteja conectado à rede de dados via 3G ou Wi-Fi,
- Pós-condições: Login efetuado.

Podemos verificar na Figura 5 o diagrama de caso de uso demonstrando a execução da funcionalidade Listar Lançamentos.

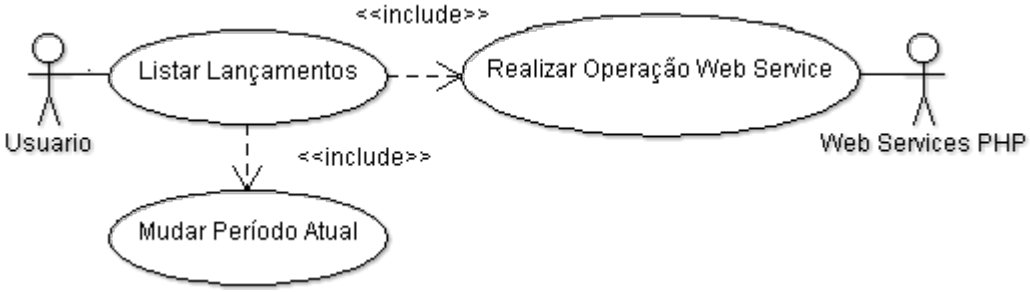

FIGURA 5 - DIAGRAMA DE CASO DE USO LISTAR LANÇAMENTOS

CONTRATO:

- Nome: Listar Lançamentos.
- Ator Principal: Usuário.
- Ator de Suporte: *Web Service* PHP.
- Pré-condições: Que esteja conectado à rede de dados via 3G ou Wi-Fi, que esteja autenticado e selecione uma das abas de listagem de entradas ou saídas.
- Pós-condições: Lista de entradas ou saídas disponibilizada.

Podemos verificar na Figura 6 o diagrama de caso de uso demonstrando a execução da funcionalidade Incluir Lançamento.

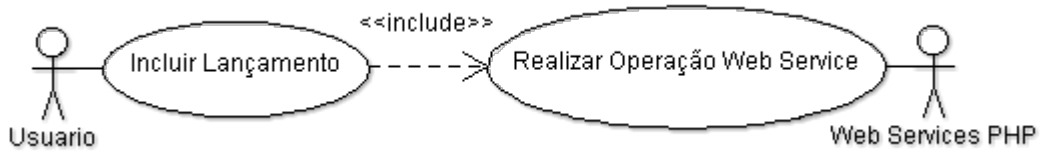

FIGURA 6 - DIAGRAMA DE CASO DE USO INCLUIR LANÇAMENTO

CONTRATO:

- Nome: Incluir Lançamento.
- Ator Principal: Usuário.
- Ator de Suporte: *Web Service* PHP.
- Pré-condições: Que esteja autenticado e conectado à rede de dados via 3G ou Wi-Fi.
- Pós-condições: Lançamento incluído.

Podemos verificar na Figura 7 o diagrama de caso de uso demonstrando a execução da funcionalidade Incluir Lançamento.

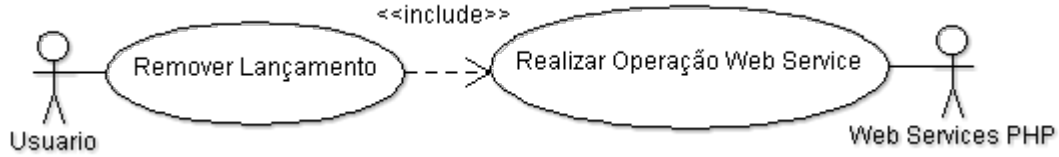

FIGURA 7 - DIAGRAMA DE CASO DE USO REMOVER LANÇAMENTO

CONTRATO:

- Nome: Remover Lançamento.
- Ator Principal: Usuário.

- Ator de Suporte: *Web Service* PHP.
- Pré-condições: Que esteja autenticado e conectado à rede de dados via 3G ou Wi-Fi e selecione um item da lista de entradas ou saídas.
- Pós-condições: Lançamento Removido.

Podemos verificar na Figura 8 o diagrama de caso de uso demonstrando a execução da funcionalidade Editar Lançamento.

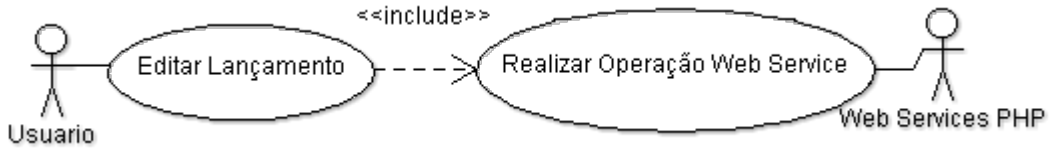

FIGURA 8 - DIAGRAMA DE CASO DE USO EDITAR LANÇAMENTO

CONTRATO:

- Nome: Editar Lançamento.
- Ator Principal: Usuário.
- Ator de Suporte: *Web Service* PHP.
- Pré-condições: Que esteja autenticado e conectado à rede de dados via 3G ou Wi-Fi e selecione um item da lista de entradas ou saídas.
- Pós-condições: Lançamento Alterado.

Podemos verificar na Figura 9 o diagrama de caso de uso demonstrando a execução da funcionalidade Mostrar Saldos.

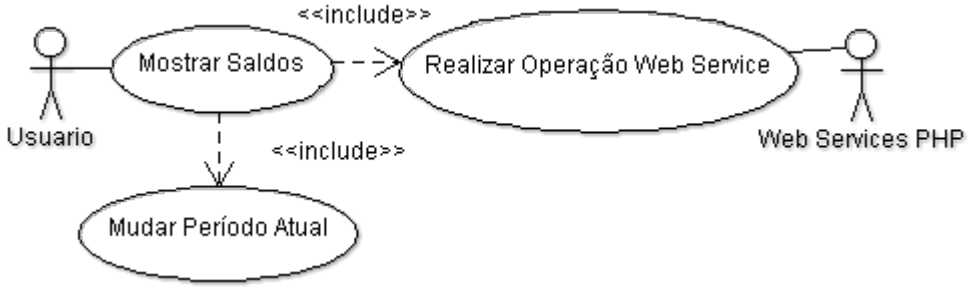

FIGURA 9 - DIAGRAMA DE CASO DE USO MOSTRAR SALDOS

CONTRATO:

- Nome: Mostrar Saldos.
- Ator Principal: Usuário.
- Ator de Suporte: *Web Service* PHP.
- Pré-condições: Que esteja autenticado e conectado à rede de dados via 3G ou Wi-Fi e selecione a opção saldos.

- Pós-condições: Mostrar posição atualizada dos saldos.

Podemos verificar na Figura 10 o diagrama de caso de uso demonstrando a execução da funcionalidade Atualizar Preferências.

FIGURA 10 - DIAGRAMA DE CASO DE USO ATUALIZAR PREFERÊNCIAS

CONTRATO:

- Nome: Atualizar Preferências.
- Ator Principal: Usuário.
- Ator de Suporte: Nenhum.
- Pré-condições: Que esteja autenticado e conectado à rede de dados via 3G ou Wi-Fi.
- Pós-condições: Preferências atualizadas.

3.3 CAMADA DE ACESSO A WEB SERVICES

O ponto chave do sistema proposto é a camada de acesso a *Web Services*, que possibilita troca de informações entre o dispositivo Android e o *Web Service*. Tal camada deve implementar forma de capturar o nome do serviço a ser acessado, bem como os parâmetros necessários, enviar a requisição ao *Web Service*, tratar esse retorno e devolver o XML obtido à função que originou a requisição.

Essa camada é composta por uma classe que proporciona a conexão com o serviço bem como o pós processamento do XML recebido.

3.3.1 Classe WebServiceLayer

Nesta seção será detalhada a classe WebServiceLayer que trata da recepção dos parâmetros e nome do serviço a ser executado, realiza a chamada, trata o retorno e devolve o XML ao método chamador.

3.3.1.1 Método: executarMetodo()

O método executarMetodo trata da recepção dos parâmetros informados pelos métodos chamadores. Métodos chamadores são métodos de outras classes

que se utilizam dessa funcionalidade para invocar um serviço, o qual está identificado pelos parâmetros juntamente com o seu nome. Esse método possui dois parâmetros de entrada e uma lista de retorno e sua chamada pode ser demonstrada conforme abaixo:

LinkedHashMap executarMetodo(**String nomeMetodo, LinkedHashMap parametros**)

onde:

<LinkedHashMap> Lista que recebe as informações de retorno do método;

<nomeMetodo> indica o nome do serviço a ser consumido do *Web Service*;

<parametros> indica uma lista de parâmetros requeridos pelo serviço a ser consumido. Exemplo:

```
wsRetorno = new LinkedHashMap();
parametros = new  LinkedHashMap();

parametros.put("usuario", idUsuario);
parametros.put("tipoLancamento", TIPO_SAIDAS);
parametros.put("anoMesReferencia",anoAtual+"/"+mesAtual);
WebServiceLayer ws = new WebServiceLayer();

wsRetorno =  ws.executarMetodo("getListaLancamentos",
parametros);
```

QUADRO 2 - EXEMPLO CHAMADA EXECUTARMETODO

A lista de retorno descrita no QUADRO 2 como **wsRetorno** possui os seguintes atributos:

<WS_ITEM_RETORNO_XML = xml>, contém o XML retornado pelo serviço.

<WS_ITEM_RETORNO_MENSAGEM = mensagem>, contém um texto livre para indicar o descritivo de sucesso ou falha na execução do serviço.

<WS_ITEM_RETORNO_STATUS = status>, indica se o serviço foi executado com sucesso **<WS_STATUS_OK = OK>** ou falha **<WS_STATUS_ERRO = ERRO>**.

3.3.1.2 Método: tratarRetornoWebService()

O método tratarRetornoWebService tem como objetivo remover caracteres indevidos que são incluídos no retorno pelo *Web Service* PHP. Sem esse tratamento, o XML obtido não poderia ser lido por um parser. Um parser é um componente de software, biblioteca ou programa padrão que trata especificamente o isolamento do desenvolvedor em relação às especificidades do XML e validações de sintaxe.

4 VALIDAÇÃO DO SISTEMA PROPOSTO

Este capítulo tem como objetivo demonstrar exemplos de utilização do sistema proposto, possibilitando que o usuário possa atuar na realização dos casos de uso elicitados, detalhando também como cada caso de uso implementado trata a comunicação entre o dispositivo Android e o *Web Service*.

4.1 REALIZAR LOGIN

Para ser possível realizar login, o usuário deve estar previamente cadastrado no site. A partir de então o usuário poderá informar e-mail e senha para ter acesso às demais funcionalidades disponibilizadas pelo sistema proposto. O usuário para realização da autenticação é um endereço de e-mail. A senha é composta por caracteres alfanuméricos, conforme demonstrado pela figura 11:

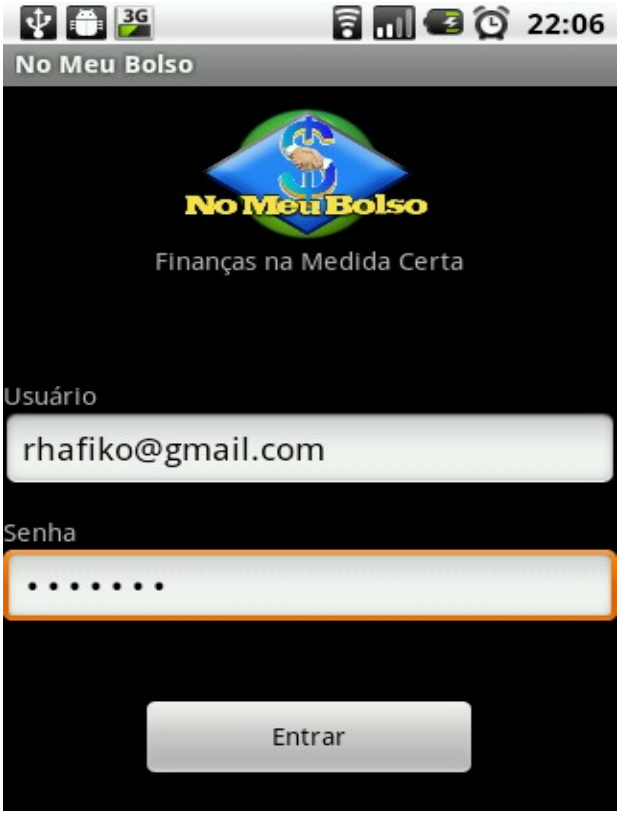

FIGURA 11 - TELA REALIZAR LOGIN

O processo de autenticação, bem como os demais casos de uso implementados, proporcionam validação dos dados informados pelo usuário, conforme pode ser visto na figura 12, a qual demonstra a situação em que são informados dados inválidos para realização da autenticação.

FIGURA 12 - TELA DE CONSISTÊNCIA DA AUTENTICAÇÃO

Todo esse processo de validação dos dados é realizada pelo *Web Service*, o qual implementa a regra de negócio em sua totalidade, devolvendo para a aplicação Android o resultado da execução, que por sua vez reconhece o retorno e mostra a mensagem conforme a situação identificada ou encaminha o fluxo de execução da aplicação para o próximo passo do caso de uso.

O processo de realização de login obedece as tarefas, conforme demonstrado no diagrama de estados abaixo:

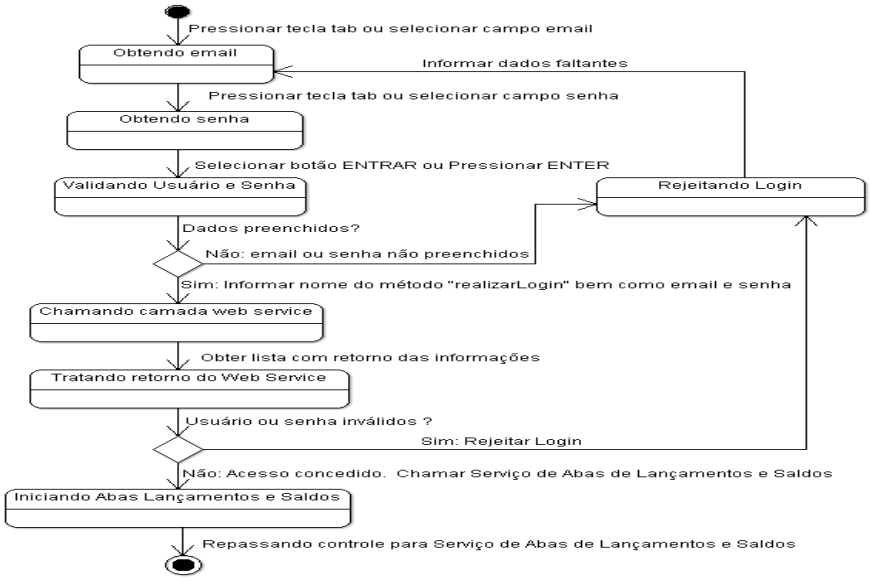

FIGURA 13 - DIAGRAMA DE ESTADOS REALIZAR LOGIN

4.2 LISTAR LANÇAMENTOS

Após ter sido realizado o login, o usuário estará autenticado no sistema, o que possibilitará ter acesso à listagem dos lançamentos de **Entradas** ou **Saídas**, também tratados pelos termos Receitas ou Despesas.

De forma automática, o sistema reconhece qual o mês e ano a serem utilizados como parâmetros de pesquisa das informações de lançamentos, tomando como base a data corrente e código do usuário obtido pelo processo de autenticação. Dessa forma, invoca o serviço responsável por pesquisar no *Web Service* os lançamentos de Entradas e à medida que o usuário aciona a aba correspondente às Saídas, também repete essa mesma consulta, conforme figura 14.

FIGURA 14 - ABAS COM OPÇÕES DE VISUALIZAÇÃO

Por padrão os lançamentos de entradas são visualizados após autenticação do usuário, conforme demonstra figura 15.

FIGURA 15 - LISTAGEM DAS ENTRADAS

O processo de que efetua a listagem dos lançamentos segue um conjunto de tarefas, conforme demonstrado no diagrama de estados abaixo:

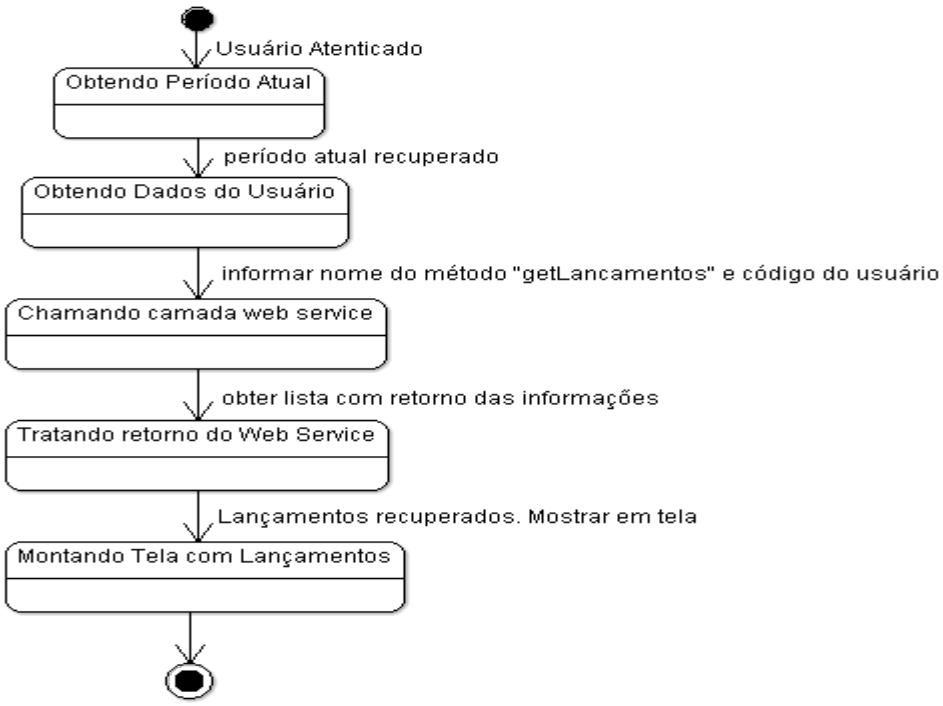

FIGURA 16 - DIAGRAMA DE ESTADOS LISTAR LANÇAMENTOS

4.3 INCLUIR LANÇAMENTO

Uma das principais funcionalidades do sistema consiste em permitir que o usuário possa realizar a inclusão de um lançamento de forma rápida e prática, atualizando de forma instantânea os registros no banco de dados do *site*. Para tal, assim como as demais funcionalidades que interagem com lançamentos e saldos, é necessário que o usuário esteja autenticado.

A figura 17 demonstra exemplo de menu de contexto utilizado para acessar a opção de **Novo** lançamento.

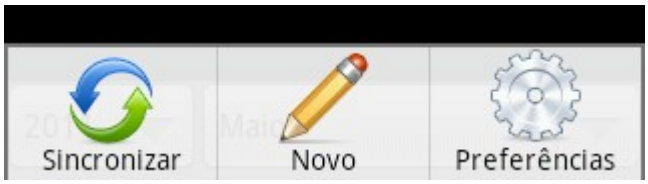

FIGURA 17 - MENU DE CONTEXTO

Acessando a opção **Novo** disponível no menu de contexto da aplicação, será realizada comunicação com o *Web Service* a fim de obter as contas cadastradas

pelo usuário, mostrando assim a interface que possibilita ao usuário informar os dados que irão compor o registro de lançamento, indicando:

- Forma de recebimento/pagamento;
- Classificação;
- Data de lançamento;
- Data de vencimento;
- Data de processamento;
- Valor do lançamento;
- Histórico;
- Observações;
- Número de documento.

A figura 18 demonstra exemplo de tela de entrada de dados para um novo lançamento de entrada.

FIGURA 18 - TELA NOVO LANÇAMENTO

O diagrama de estados a seguir, demonstra como ocorrem as interações entre a seleção da nova opção pelo usuário, inclusão dos dados no formulário, a confirmação da transação, comunicação com o *Web Service* realizando inclusão do lançamento, resultando na inclusão de um novo registro.

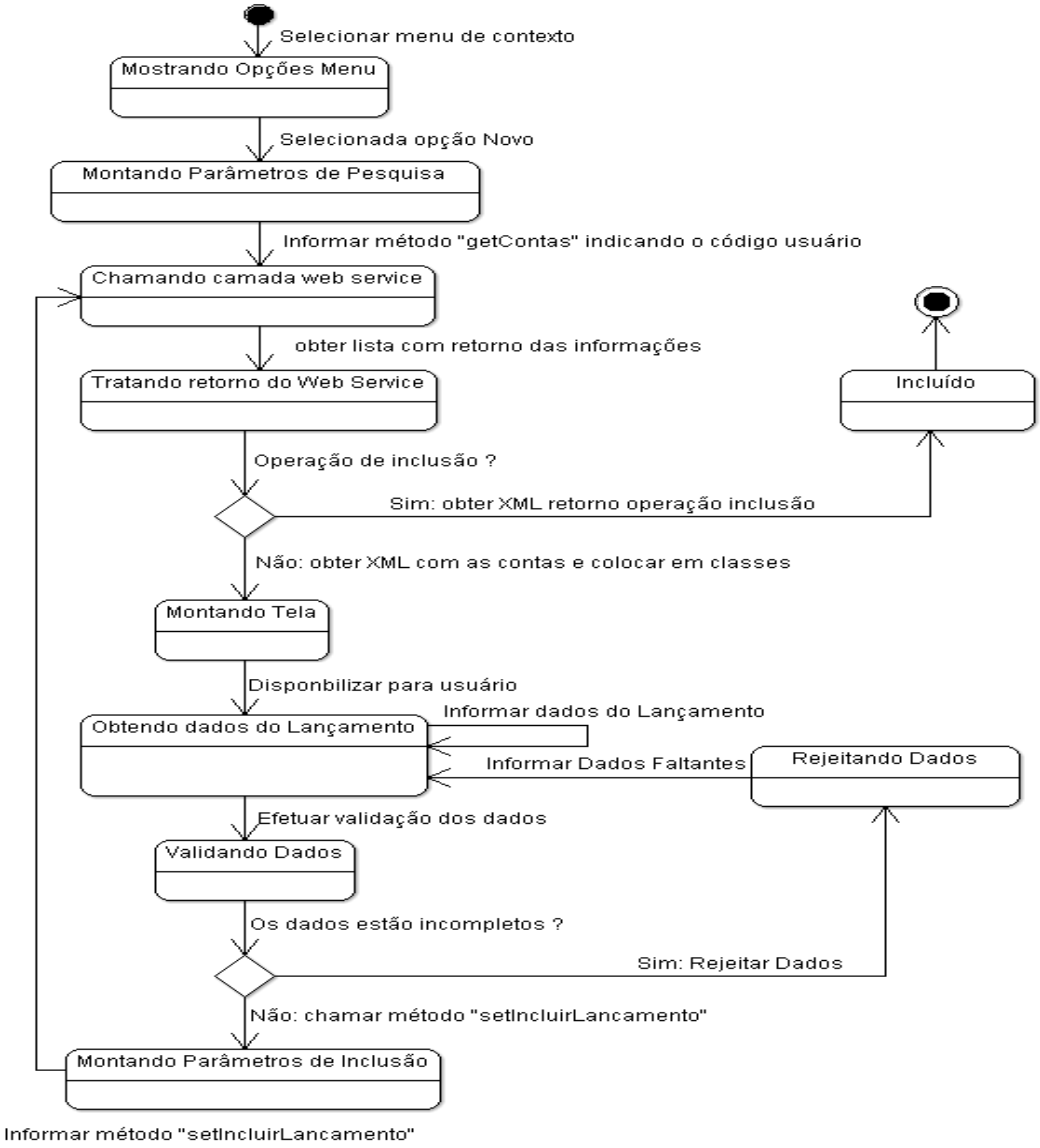

FIGURA 19 - DIAGRAMA DE ESTADOS INCLUIR LANÇAMENTO

4.4 REMOVER LANÇAMENTO

Tomando como base a lista de lançamentos disponibilizada ao usuário, uma das opções que se apresentam é a de remoção de um lançamento. Esta opção é utilizada quando o usuário deseja remover um determinado lançamento o qual não faz mais parte do objetivo do controle financeiro, seja por inclusão indevida ou até

mesmo por cancelamento de uma compra ou compromisso. A figura 20 exemplifica o menu de opções acionado assim que o usuário seleciona um dos itens da lista de lançamentos de entradas ou saídas, no qual está disponível a opção **Remover**.

FIGURA 20 - MENU REMOVER LANÇAMENTO

Assim como a maioria das operações realizadas por um sistema computacional, uma operação de remoção requer que o usuário confirme a ação. Dada essa confirmação o sistema comunica-se com o *Web Service* indicando o código identificador do lançamento selecionado o qual o usuário deseja que seja realizada a remoção. A figura 21 exemplifica a interface de diálogo utilizada para confirmação da operação de remoção do lançamento.

FIGURA 21 - DIÁLOGO REMOVER LANÇAMENTO

A figura 22 demonstra a finalização do processo, com mensagem confirmando o sucesso da operação de remoção do lançamento.

FIGURA 22 - MENSAGEM LANÇAMENTO REMOVIDO

O diagrama de estados apresentado pela figura 23 representa o processo de remoção de um lançamento.

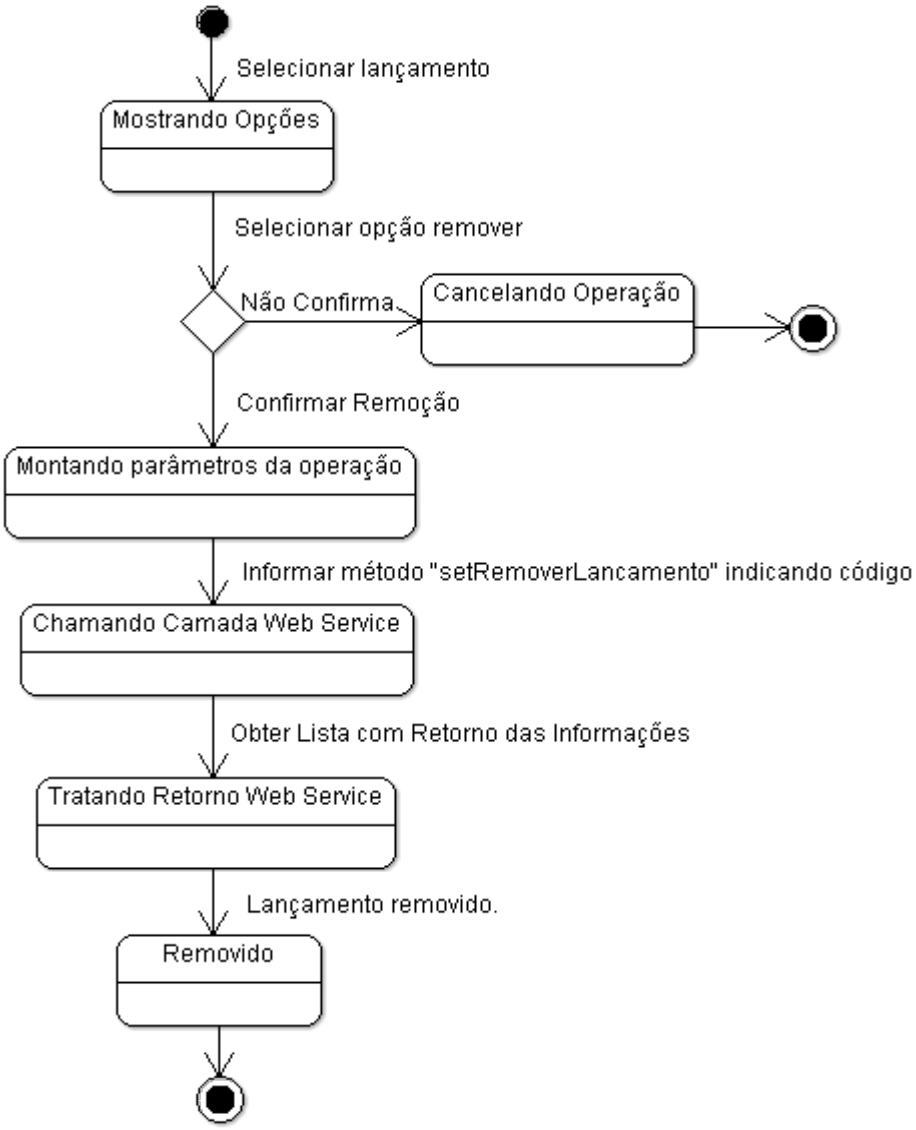

FIGURA 23 - DIAGRAMA DE ESTADOS REMOVER LANÇAMENTO

4.5 EDITAR LANÇAMENTO

Tomando como base a lista de lançamentos disponibilizada ao usuário, uma das opções que se apresentam é a de edição de um lançamento. Esta opção é utilizada quando o usuário deseja alterar as informações de um determinado lançamento o qual sofreu alteração de dados ou tão somente está sendo alterado para indicar que este lançamento não se encontra mais pendente, sendo informada a data de processamento, o que muda o status do registro para Pago. A figura 24 exemplifica o menu de opções acionado assim que o usuário seleciona um dos itens da lista de lançamentos de entradas ou saídas.

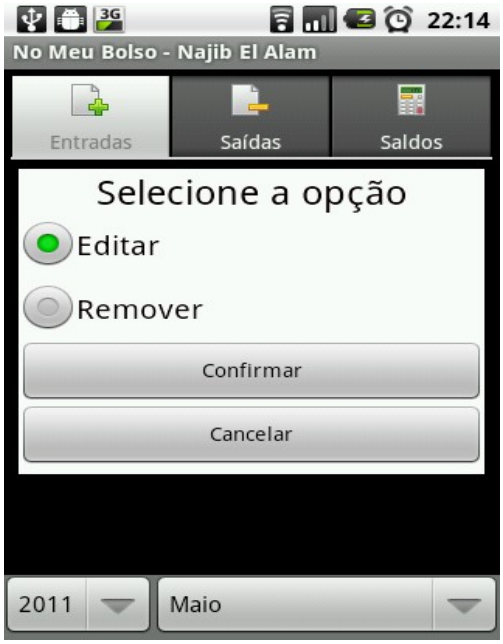

FIGURA 24 - MENU EDITAR LANÇAMENTO

Após a indicação de que o usuário deseja editar o lançamento selecionado, o sistema efetua a consulta dos detalhes desse lançamento utilizando o *Web Service*, informando qual o código do lançamento. O *Web Service* realiza a consulta do lançamento específico e devolve as informações para que sejam apresentadas em tela a fim de que o usuário possa realizar a alteração dos dados. A figura 25 demonstra a tela utilizada para edição de um lançamento, bem como momento em que o usuário está prestes a confirmar a alteração do registro.

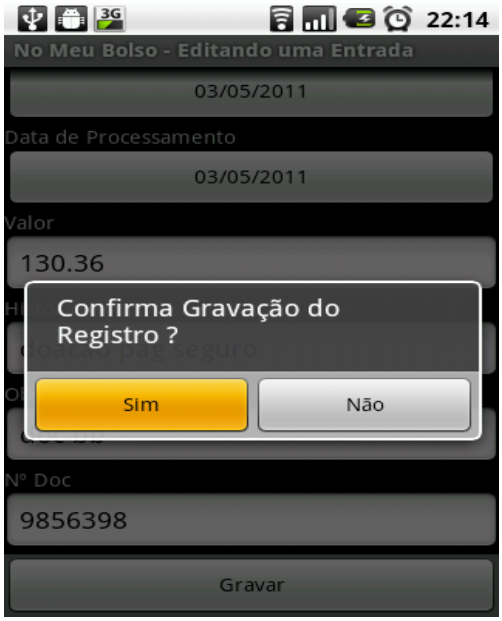

FIGURA 25 - TELA CONFIRMAÇÃO EDITAR

Após a confirmação ocorre nova chamada da camada que trata *Web Service* para que sejam persistidos os dados alterados. A figura 26 representa o diagrama de estados que demonstra como ocorre o processo de edição de um lançamento.

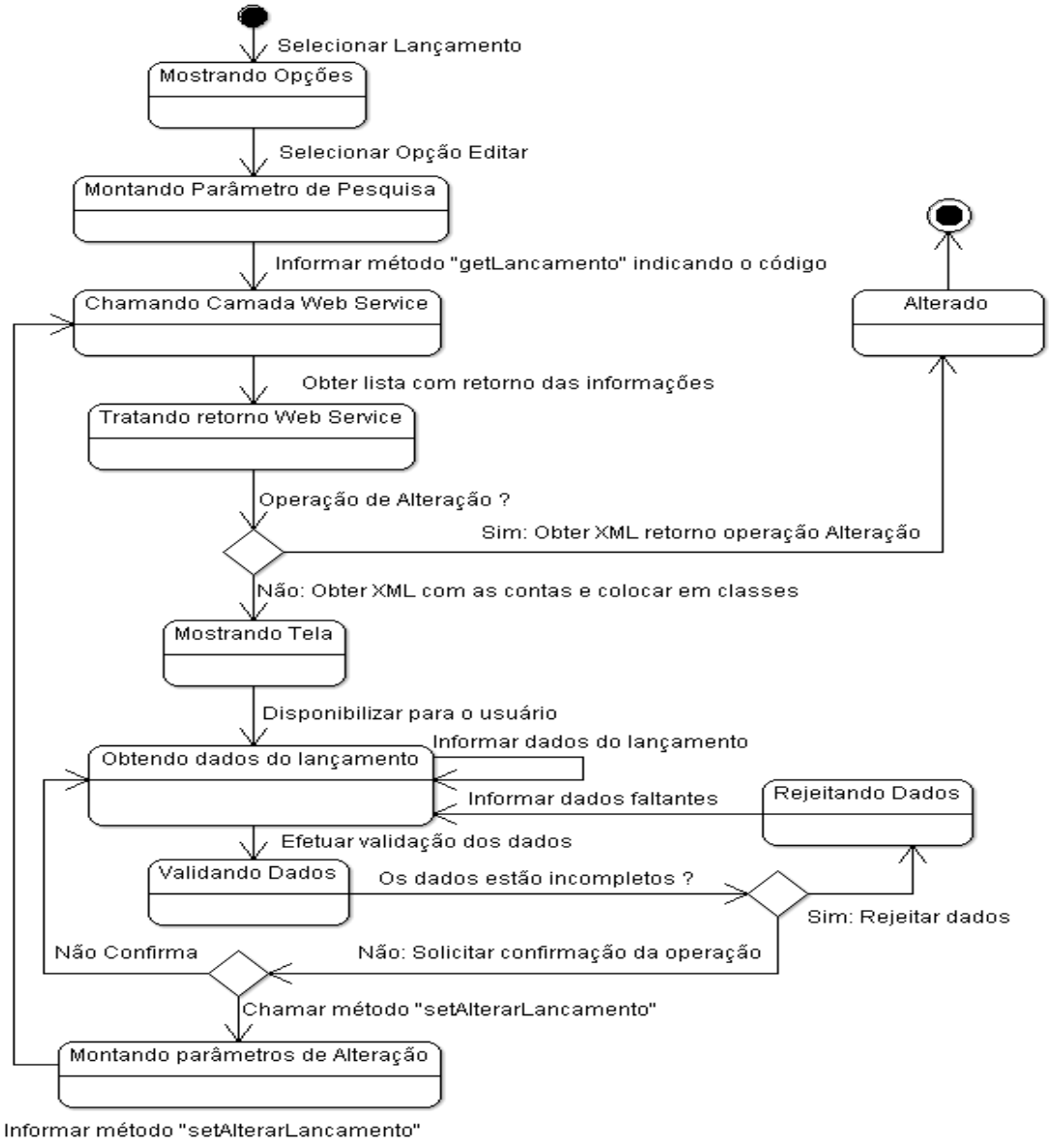

FIGURA 26 - DIAGRAMA DE ESTADOS EDITAR LANÇAMENTO

4.6 MOSTRAR SALDOS

A forma pela qual o usuário identifica quando a sua situação financeira em um determinado mês está ou não favorável é utilizando da consulta dos **Saldos**. Nesta aba específica são apresentadas informações de saldos totalizando os valores não pagos e pagos até o mês **Anterior**, bem como os valores não pagos e pagos compreendidos no período do mês **Atual** e ainda demonstra de forma consolidada a

junção desses dois cenários o qual representa a situação em que os valores pagos e não pagos do mês anterior, somados aos valores pagos e não pagos do mês atual, resultam na informação **Final** a ser considera como saldo. A figura 27 demonstra como ocorre a visualização dessas informações.

FIGURA 27 - TELA DE SALDOS

Neste caso, para ser realizada a busca das informações de saldos, o sistema considera o período representado pelo mês e ano atualmente selecionados. A qualquer momento o usuário poderá alterar a seleção de mês e ano, necessitando assim que seja realizada nova busca do saldo atualizado, considerando a nova configuração de período. A figura 28 demonstra no menu de contexto a opção **Sincronizar**, utilizada para tal finalidade.

FIGURA 28 - OPÇÃO SINCRONIZAR SALDO

4.7 ATUALIZAR PREFERÊNCIAS

Por se tratar de um sistema de uso pessoal e em ambiente totalmente restrito, acaba por se tornar de certa forma incômoda, toda vez que o aplicativo for aberto, a necessidade de sempre solicitar que o usuário informe o e-mail e senha para que possa ser realizada a autenticação. Assim, o sistema proposto conta com a funcionalidade na qual pode-se determinar como preferência pessoal o armazenamento desses dados, até que seja realizada troca da senha através do *site*.

A senha de acesso é armazenada no dispositivo Android depois de transformada em um hash utilizando o algoritmo MD5 (Message-Digest algorithm 5) (Wikipedia: MD5, 2011). Do lado servidor o processo de autenticação compara o hash da senha enviado concedendo ou não acesso a aplicação.

A opção **Preferências** está disponível no menu de contexto, conforme demonstrada pela figura 28. Ao selecionar esse item o usuário poderá marcar a opção que indica a permanência da autenticação até que seja realizada a troca da senha pelo *site*, conforme demonstrada na figura 29.

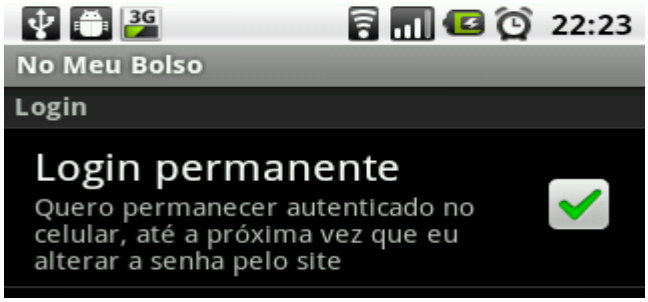

FIGURA 29 - TELA DE PREFERÊNCIAS

4.8 AMBIENTE DE TESTES

O sistema proposto foi testado em celular LG GW620 com sistema operacional Android Froyo 2.2. O computador de desenvolvimento utiliza um processador AMD Athlon(tm) 64 X2 Dual Core TK-57 com 1.90GHz em cada núcleo e 2GB de MEM. RAM.

4.9 RESULTADOS OBTIDOS

Como resultados do trabalho elaborado, é possível destacar a aplicação de conhecimentos obtidos na área de implementação para dispositivos móveis,

aprender técnicas de comunicação entre dispositivos móveis e *Web Services*, bem como enriquecer ainda mais a experiência de utilização do sistema de gestão financeira.

O sistema proposto foi testado satisfatoriamente atendendo aos objetivos traçados. O mesmo já está sendo utilizado por alguns usuários do *site*, de forma experimental. A utilização do sistema completo pelos demais usuários bem como evoluções estão em estudo para implementação pois envolve questões relacionadas ao dimensionamento do servidor *web* atual para suportar essas requisições.

5 CONCLUSÃO E TRABALHOS FUTUROS

O sistema elaborado neste trabalho se propôs a desenvolver uma aplicação para gestão financeira pessoal que pudesse disponibilizar de forma dinâmica o acesso às informações disponíveis no sistema de Gestão Financeira Pessoal, tornando também possível a integração entre dispositivos Android e *Web Services*. Para isso, foi realizada a implementação da comunicação entre um dispositivo Android e os Serviços disponíveis pelo *site* de finanças pessoais denominado No Meu Bolso, utilizando-os por meio de *Web Services*. Dessa forma, foi possível estabelecer autenticação do usuário previamente cadastrado, bem como a consulta, inclusão, alteração e exclusão de lançamentos financeiros, que compunham informações de cunho pessoal e restrito ao usuário do serviço. Para o sistema proposto realizar esta comunicação, foi necessária utilização de uma biblioteca específica para dispositivos Android, denominada Ksoap2, que proporciona de forma transparente a comunicação entre o dispositivo Android e o Serviço Web.

Como um dos objetivos dos profissionais de informática, que trabalham no desenvolvimento de sistemas é o de elaborar soluções que proporcionem agilidade e segurança na disponibilização de informações, bem como auxiliar seus usuários nas tomadas de decisões, acreditamos que o sistema proposto neste trabalho conseguiu atender aos objetivos traçados inicialmente.

Como trabalhos futuros cabe sugerir expandir a aplicação para que a sincronização das informações sejam persistidas em um banco de dados local, utilizando do SQLite nativo em dispositivos Android, a fim de tornar ainda melhor a experiência de uso, bem como de certa forma diminuir a necessidade de acessos frequentes ao *Web Service*, uma vez que os registros estejam disponíveis em base local, sanando também a necessidade do usuário estar conectado o tempo todo para ter acesso às informações remotas. Também sugere-se a criação de uma interface gráfica onde estejam disponibilizados gráficos de fácil acesso e visualização instantânea compondo as principais receitas e despesas do período selecionado pelo usuário.

6 REFERÊNCIAS BIBLIOGRÁFICAS

ROGERS, Rick; LOMBARDO John; MEDNIEKS Zigurd; MEIKE, Blake. **Android: Desenvolvimento de Aplicações Android**. São Paulo: Novatec Editora Ltda, 2009.

IDGNOW. **Opinião: Porque o Android está avançando entre os sistemas móveis.** Disponível em: http://idgnow.uol.com.br/mercado/2010/09/09/android-iphone-apple-google/. Acesso em: 10 dez. 2010

SANTOS, Alfredo Luiz dos. **Integração de Sistemas com Java**. Rio de Janeiro: Brasport, 2007.

REVERBEL, Francisco. **O que são web services**. USP, São Paulo, mai. 2006. Disponível em: <http://www.ime.usp.br/~reverbel/SOD-06/trabalhos/fachada-ws/node2.html>. Acessado em: 11 abril 2011.

ANDROID DEVELOPERS. **What is Android?**. Disponível em: http://developer.android.com/guide/basics/what-is-android.html

KSOAP2-ANDROID. **History**. Disponível em: http://code.google.com/p/ksoap2-android/wiki/History. Acessado em: 15 maio 2011.

KSOAP PROJECT. Disponível em: http://ksoap.objectweb.org/. Acessado em: 15 maio 2011.

WIKIPEDIA. **MD5**. Disponível em: http://pt.wikipedia.org/wiki/MD5. Acessado em: 12 maio 2011.

ESTADO DE MINAS. **Usuários buscam mais acessibilidade e portabilidade com os netbooks**. Disponível em: http://www.em.com.br/app/noticia/tecnologia/2011/03/24/interna_tecnologia,217318/mobilidade-contra-a-limitacao.shtml. Acessado em 26 de junho de 2011.