Adaptação Dinâmica de Dispositivos Móveis em Cenários de Computação Pervasiva

Alexandre Pergher Stefani[1], Reiner Franthesco Perozzo[1]

[1]Curso de Ciência da Computação – Centro Universitário Franciscano - Caixa Postal 97010-032 – Santa Maria – RS – Brasil

pergherstefani@gmail.com, reiner.perozzo@unifra.br

Abstract. *This work presents a proposal for dynamic changing settings of a mobile device based on user's preferences and the environment in which one is located. These changes intend to adapt mobile behavior in different user's environments and contexts. The identification of environments is performed by a Service Set Identifier (SSID) of the local network compared with pre-defined configuration on the mobile device profiles. Among the customizations adjustments that can be made, the main emphasizes are; sounds, vibrations and strategies to silence the device without the user's manual interference.*

Resumo. *Este trabalho apresenta uma proposta para alteração dinâmica das configurações de um dispositivo móvel com base nas preferências do usuário e no ambiente em que o mesmo se encontra. Essas alterações visam adaptar o comportamento do dispositivo móvel em diferentes ambientes e contextos em que o usuário se encontra. A identificação dos ambientes é efetuada por um Service Set Identifier (SSID) da rede local comparado com perfis de configuração pré-definidos no dispositivo móvel. Dentre os ajustes das customizações que podem ser realizadas, destacam-se, sons, vibrações e estratégias para silenciar o dispositivo sem a interferência manual do usuário.*

1. Introdução

A utilização de dispositivos móveis tais como celulares, *notebooks* e *tablets* tem se tornado cada vez maior no mercado digital. Para satisfazer a necessidade dos usuários, torna-se imprescindível a implementação de aplicações para auxiliar no gerenciamento dos recursos disponibilizados nos dispositivos móveis.

Nesse contexto insere-se a comunicação entre estes dispositivos, os quais estão cada vez menores, mais ágeis e com mais benefícios computacionais, trazendo em foco a inclusão da computação invisível ao mundo físico. Segundo Weiser (1991), as tecnologias mais profundas e duradouras são aquelas que desaparecem, pois elas se mesclam na vida cotidiana até tornarem-se indissociáveis da mesma.

Para se construir o cenário visualizado por Weiser (1991), fazem-se necessárias pesquisas de âmbito multidisciplinar envolvendo algumas áreas da computação, entre elas os sistemas distribuídos, os sistemas móveis, redes de computadores e engenharia de *software*.

Com o passar dos anos, muitos dispositivos foram fabricados, e para eles, muitos aplicativos foram desenvolvidos. Com a disponibilidade de diversos aplicativos surge a necessidade de automatizar algumas funcionalidades do dispositivo móvel de acordo com a sua preferência e o ambiente onde o usuário se encontra. Visando promover

praticidade na gerência de um dispositivo móvel, propostas com a finalidade de automatizar funcionalidades dos dispositivos móveis podem ser bem-vindas nesse segmento.

A utilização de dispositivos móveis vem aumentando com a evolução da informatização, mas a adaptação com o uso desses aparelhos eletrônicos pode ser um problema quando o mesmo sofre influências de um alarme indesejado durante uma reunião de extrema importância para o usuário ou chamadas não desejadas em horários de trabalho.

Assim, este trabalho visa desenvolver uma aplicação para auxiliar o usuário no controle de funcionalidades de um dispositivo móvel durante as variações de ambiente ligadas à rotina do mesmo, trazendo a ele comodidade e automatização de algumas funcionalidades efetuadas por seu dispositivo.

Para isto, é necessário extrair informações do ambiente para auxiliar na tomada de decisões do dispositivo móvel, as quais a Computação Pervasiva permite. Para definir o que é realmente importante para o usuário em um determinado momento, mostrando uma mudança de perfil do dispositivo móvel diretamente ligada às preferências do usuário.

1.1. Objetivo

Desenvolver um aplicativo para alterar o comportamento de um dispositivo móvel em ambientes pervasivos conforme o perfil do usuário. De acordo com o ambiente em que o usuário se encontra, o dispositivo móvel poderá se adaptar de acordo com as preferências do usuário, alterando configurações do dispositivo, tais como restrições de chamadas e alterações de perfil.

1.2. Objetivos Específicos

O trabalho possui os seguintes objetivos específicos:

- Investigar métodos de identificação de uma rede local por meio do SSID;
- Identificar métodos de manipulação de dados em um dispositivo móvel na plataforma *Android*;
- Implementar interfaces para *Android* para inserir as características de comportamento de um dispositivo móvel conforme o perfil do usuário.

2. Referencial Teórico

Nesta Seção é apresentado um estudo sobre computação pervasiva, conceituando-a e visando buscar características importantes para o trabalho em questão. Esta Seção traz, também, algumas referências sobre Computação Móvel, Computação Sensível ao Contexto e Perfil do Usuário. Por fim, uma descrição da plataforma *Android* e sua arquitetura.

2.1. Computação Pervasiva

O conceito de Computação Pervasiva implica que o computador está inserido no ambiente de forma invisível para o usuário, ou seja, o computador possui a capacidade de extrair informações do ambiente no qual ele está para então utilizá-las e construir modelos computacionais, controlando, configurando e ajustando a aplicação para

melhor atender as necessidades dos usuários. O ambiente também pode ser capaz de detectar outros dispositivos que venham a fazer parte dele. Desta interação surge a capacidade de computadores agirem de forma "inteligente" em um ambiente povoado por sensores e serviços computacionais [Araújo 2003].

A Computação Pervasiva tem como objetivo fazer com que tarefas computacionais ligadas às pessoas se tornem onipresentes de maneira que não necessitam saber como ela está disposta no ambiente. Deste modo é feita uma coleta de dados com as características do ambiente, buscando estados diferentes para ocasiões momentâneas, trazendo ao homem informações e configurações necessárias interligadas ao meio [Martini 2012].

Utilizando recursos de rede, a Computação Pervasiva visa interligar dispositivos no ambiente com a missão de executar tarefas ali encontradas sem que o usuário perceba visivelmente [Librelotto et. al. 2011].

A utilização da computação inconscientemente é demonstrada pela gama de dispositivos computacionais no ambiente, interagindo e realizando tarefas mais rápidas, tornando assim mais fácil a realização de determinadas tarefas no meio [Weiser 1991].

Segundo Ahamed, Zulkernine, Haque (2008), um estudo realizado por diversos pesquisadores identificou as características para a obtenção de Computação Pervasiva, listados a seguir:

- Ambiente Volátil: dispositivos móveis entram e saem do ambiente por conta própria;
- Desaparecimento de Limites: diferentes dispositivos interligados;
- Escalabilidade: suporte a diferentes plataformas e um número alto de dispositivos ligados na mesma rede;
- Espaço Ativo: a utilização de um sensor que possa identificar a presença de uma pessoa qualquer quando ela entra no ambiente, um espaço será designado para o sensor captar a informação e consequentemente executar uma ação;
- Mobilidade: dispositivos móveis, como por exemplo, celulares, podem estar em movimento constante no meio;
- Sensibilidade ao Contexto: o ambiente deve coletar informações e assim interagir no meio tomando algumas decisões conforme a aplicação necessita em determinadas ocasiões;
- Serviço Contínuo: manter a rede funcionando em qualquer situação, pois o usuário conta com resposta do ambiente conforme a aplicação ali presente.

Dentre as características listadas anteriormente, será analisado a seguir alguns temas semelhantes que consequentemente têm maior importância para este trabalho.

2.2. Computação Móvel

A evolução tecnológica chegou ao ponto em que é possível acessar informações praticamente em qualquer lugar do planeta e em qualquer momento [Mateus e Loureiro 1998]. Há um aumento diário no número de pessoas que passam a utilizar computação móvel, tornando os dispositivos móveis parte de sua rotina [Junior e Fernandes 2006].

A computação móvel (CM) é a área da tecnologia que amplia o domínio da Computação Distribuída, pois faz uso da comunicação sem fio para eliminar a limitação da mobilidade, onde através de um dispositivo portátil é possível se comunicar com a parte fixa da rede e com outros computadores móveis. A esse ambiente de computação se dá o nome de computação móvel [Barbosa et. al. 2007].

Estudos sobre mobilidade em sistemas distribuídos bem como a proliferação de dispositivos eletrônicos móveis e a exploração de novas tecnologias de rede sem fio, juntos, são considerados um novo paradigma denominado, então, Computação Móvel [Barbosa et. al. 2007]. Com isso, a capacidade de utilizar serviços computacionais se expande conforme a mobilidade oferecida por esses dispositivos. Por outro lado, podem limitar a relação com o ambiente físico onde o usuário se encontra [Araújo 2003].

2.3. Computação Sensível ao Contexto

Contexto é tudo que faz parte de um determinado ambiente, o contexto pode ser descrito como o local, a identidade de uma pessoa ou um objeto presente em um determinado contexto e, também, as mudanças que podem ocorrer no local (Schilit e Theimer 1994). Qualquer informação gerada pelos dispositivos eletrônicos que possam mudar algum estado atual da aplicação, ou seja, relevante ao meio, é considerada um contexto [Yamin 2004].

A Computação Pervasiva tem como importante característica a Sensibilidade de Contexto, a qual traz uma gama de informações que auxiliam na tomada de decisões com autonomia dentre as possíveis situações no ambiente [Martini 2012]. Uma informação relevante pode caracterizar uma situação, tornando a sensibilidade ao contexto uma noção fundamental para a computação Pervasiva [Dey 2001].

Na computação móvel o contexto é formado por dois aspectos, o primeiro é referente às características do ambiente, que determinam comportamento das aplicações. E o segundo aspecto trata o contexto como um conjunto de estados e configurações que determina um comportamento da aplicação [Chen e Kotz 2000].

O objetivo da Computação Sensível ao contexto é fazer com que informações ligadas ao contexto sejam capturadas e aplicadas ao ambiente, com o intuito de melhorar o desempenho, a disponibilidade e a usabilidade do sistema ali presente [Freitas 2011].

2.4. Perfil do Usuário

Esta seção trata de referenciar perfis de usuário, de como a gama de informações leva a pesquisa para diferentes estilos de perfil, de acordo com a sua aplicação em torno do exemplo. Foram filtradas as buscas e perfis caracterizados por fazer parte de exemplos que utilizam computação Pervasiva e estão localizados em dispositivos móveis.

Informações localizadas em um dispositivo móvel como, por exemplo, peso, altura, idade ou até mesmo a própria identificação do usuário, diretamente não parecem relevantes, contudo estas informações podem categorizar a pessoa em um estereótipo, tornando essas características relevantes e criando um sistema Pervasivo, capaz de antecipar decisões em determinadas ocasiões [Barth e Gomi 2004].

Sistemas adaptativos se ajustam a expectativas de usuários a partir de modelos que representam seu perfil, chamado de modelo de usuário, estes que consistem das

preferências individuais que determinam o comportamento do usuário. Preferências são todas aquelas formações que são diretamente necessárias para a adaptação do comportamento do sistema aos interesses do usuário [Barth e Gomi 2004]. A possibilidade de adaptar o comportamento da aplicação baseada nas preferências pessoais é uma das principais motivações para a construção de um sistema que é capaz de se adaptar as características do usuário [Papatheodorou 2001].

No Ambiente Pervasivo, os perfis permitem a exploração de novas oportunidades, baseadas nas características do usuário e nas informações dos contextos, por onde ele se desloca [Barbosa 2005].

2.6. Plataforma *Android*

Nesta Subseção serão especificadas algumas características da plataforma *Android*, com a finalidade de apresentar a arquitetura da plataforma.

A plataforma *Android* possui uma arquitetura formada por camadas e componentes. As camadas são:

- *Applications:* Essa camada é formada pelo conjunto de aplicações tais como calculadora, navegador, aplicações de interação da interface do dispositivo como o usuário, e funções como a realização de chamadas, envio de mensagens [Google 2013];
- *Application Framework:* Camada composta por aplicações que gerenciam o aparelho, permitindo acesso dos desenvolvedores conforme as capacidades do *hardware*, projetada para auxiliar na reutilização de componentes. São aplicações que compõe esta camada [Google 2013]: A *Activity Manager*, responsável por controlar o ciclo de vida das aplicações. A *Content Providers* que encapsula dados que precisam ser compartilhados entra diferentes aplicações. E a *Notification Manager* responsável por notificar o usuário um evento, tais como alarmes e mensagens;
- *Libraries:* Composta por bibliotecas escritas em C e C++, compiladas e pré-instaladas pelo fabricante do aparelho, disponíveis aos desenvolvedores, permitindo acesso a componentes do sistema, como banco de dados, aceleradores gráficos, e outros recursos voltados as limitações do aparelho [Google 2013];
- *Android Runtime:* Camada Responsável pela execução do código compilado, composta pela máquina virtual *Dalvik*. Pois o código escrito para *Android* é escrito em Java e executados nesta máquina virtual [Google 2013];
- *Linux Kernel:* O *Kernel* do *Android* é usado para gerenciamento de memória, gerenciamento de processos e serviços do sistema operacional [Google 2013].

3. Proposta

Este trabalho desenvolveu um aplicativo para a realização de alteração de configurações de um dispositivo móvel de forma automática. A aplicação tem como funcionalidade principal especificar qual perfil do dispositivo será ativado, ajustando modos sonoros, vibrações ou silenciar conforme o contexto do usuário.

Para orientar quando o comportamento do dispositivo móvel deverá ser alterado automaticamente, é utilizada uma gama de informações inicialmente inseridas na

aplicação e o local onde o usuário se encontra. Apresentando um contexto com capacidade de auxiliar na tomada de decisões em relação ao ambiente. Para isso, é necessário extrair informações do ambiente para auxiliar na tomada de decisões do dispositivo móvel, as quais a Computação Pervasiva permite. Assim, busca-se definir o que, realmente, é importante para o usuário em um determinado momento, mostrando uma mudança de perfil do dispositivo móvel diretamente ligada às preferências de configuração do dispositivo naquele dado momento conforme ilustra a Figura 1.

Figura 1 – Proposta

Dentre os recursos de configuração presentes nos dispositivos móveis, podem-se destacar o modo de controle de alertas de chamadas, campainhas e volumes de determinadas aplicações. Essas são funcionalidades passíveis de ajustes dentro da plataforma *Android*, pois podem proporcionar um modo Silencioso - muito utilizado em casos que o usuário prefira que o aparelho não interfira sonoramente [Google 2013]. A plataforma *Android* foi escolhida na implementação da aplicação por ter sido amplamente usada no cenário comercial e acadêmico, aliada com a disponibilidade de manipulação de aplicações ligadas a ela.

No que tange a metodologia de desenvolvimento desta proposta, observa-se que o surgimento de metodologias ágeis pode minimizar os impactos relacionados com problemas de atrasos no desenvolvimento de *software* encontrados em metodologias clássicas [Barbosa et al. 2008]. Diante desse contexto e visando a facilidade de inserção de novas funcionalidades no sistema ao longo do tempo, optou-se por utilizar neste trabalho a metodologia ágil *Feature Driven Development* (FDD).

A metodologia é dividida em cinco processos sequenciais conforme ilustra a Figura 2.

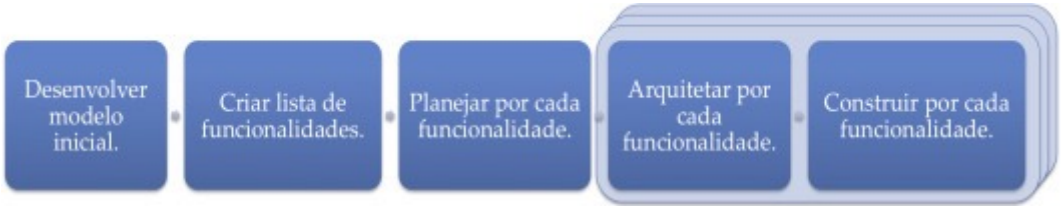

Figura 2 – Processos FDD [Palmer e Felsing 2012].

Dos processos apresentados na Figura 2, apenas uma vez são executados os processos de desenvolvimento do modelo inicial, da criação da lista de funcionalidade, e do planejamento por funcionalidades. Logo, os processos de detalhamento por funcionalidade e de construção por funcionalidade são considerados incrementais, pois são executadas inúmeras vezes, uma para cada funcionalidade que compõe a aplicação [Palmer e Felsing 2012].

3.1. Desenvolvimento de um modelo

A primeira fase consiste em entender o problema proposto, e de acordo com os requisitos coletados e as funcionalidades solicitadas pelo cliente, é efetuado um estudo para montar a estrutura do sistema através de um diagrama [Barbosa et al. 2008].

Para a elaboração deste sistema, com a intenção de auxiliar no desenvolvimento do mesmo, foi elaborado um diagrama de classes com atributos e métodos ilustrado na Figura 9 Apêndice 1.

A classe principal do projeto está centrada na MainActivity responsável por instanciar objetos de leitura e gestão da classe. Essa por sua vez, possui os recursos necessários para reconhecer um contexto diante do SSID das redes e quando necessário efetuar as alterações oportunas para adaptar, de forma dinâmica, o comportamento do dispositivo móvel. Para a manipulação do contexto armazenado no aplicativo, a classe ConfigAtivity é responsável pelo gerenciamento de uma interface secundária onde estão contidos os campos de preenchimento do contexto relacionado às preferencias do usuário. As demais classes auxiliam na relação da aplicação com os recursos disponibilizados pela plataforma *Android*.

3.2. Criação de uma lista de funcionalidades

Na segunda fase é construída uma lista de funcionalidades detalhadas e ordenadas de forma hierárquica, conforme modelo inicial [Barbosa et al. 2008]. Cada funcionalidade proposta na lista deve produzir um produto final ao cliente [Palmer e Felsing 2012].

A lista de funcionalidade para contribuir com a elaboração do projeto é constituída por:

- Criação do contexto;
- Armazenamento do contexto;
- Adaptação de Cenário;
- Reconhecimento do contexto;
- Gestão de Cenários.

3.3. Planejamento por funcionalidades

Na terceira fase é feito um planejamento para auxiliar no desenvolvimento das funcionalidades [Barbosa et al. 2008]. É nesta fase que são definidas sequências e os prazos para as funcionalidades [Palmer e Felsing 2012].

Nesta etapa são inseridos os dados relevantes às preferências do usuário, ela é efetuada inicialmente e quando o usuário interpretar uma mudança em seu perfil. O sistema deverá proporcionar para o usuário, acesso ao cadastro e manipulação de ambientes e suas características. Apenas nesta etapa, o usuário deverá manualmente especificar um nome de um ambiente e o SSID da rede local. Para cada ambiente é necessário especificar se o perfil de chamadas ficará em modo normal, vibrar ou silencioso e nessa mesma etapa o usuário poderá escolher se a função *bluetooth* permanecerá habilitada ou desabilitada.

As preferências inseridas pelo usuário serão armazenadas em arquivos que permanecerão com os documentos onde está instalado o aplicativo. Para cada cenário cadastrado automaticamente será criado um arquivo contendo os dados para caracterizar o cenário e este arquivo será apagado sempre que o cenário for excluído do sistema. O usuário tem como função inserir, alterar e excluir os cenários manualmente, as outras funções executas na aplicação serão iniciadas automaticamente conforme o contexto estabelecido momentaneamente.

Através da utilização de componentes da plataforma *Android*, um método faz a verificação da rede local conectada ao dispositivo móvel, consequentemente, será feita uma comparação de dados, e caso a rede for reconhecida como característica de um dos cenários cadastrados pelo usuário, a aplicação deverá informar a gestão de cenário. A aplicação deverá realizar de forma automática as alterações de comportamento do dispositivo móvel relacionadas com o ambiente onde se encontra. A seguir pode-se ver quais as alterações de comportamento serão alteradas pelo sistema.

3.4. Detalhamento por funcionalidades

Na quarta fase, cada funcionalidade é analisada e estudada separadamente, para auxiliar na construção da *feature* [Barbosa et al. 2008].

Na criação de cenários, uma interface secundária é implementada com a intenção de promover ao usuário campos para inserir o contexto desejado. Para armazenar os valores que compões as características dos cenários cadastros pelo usuário, a utilização de arquivos foi implementada utilizando métodos de gravação com a auxílio da classe *FileOutputStream* da plataforma *Android*.

Os métodos de manipulação de dados em arquivos proporcionarão ao usuário a criação de até 3 cenários e a oportunidade de visualizar em uma interface secundária cada senário, quando for necessário poderão alterar qualquer característica proposta pelos cenários.

Através de métodos de comparação e busca de dados, é efetuada uma verificação constante na situação de qual rede local o dispositivo móvel está conectado, relacionado a uma verificação do SSID que compõe parte do contexto de cada cenário, para estabelecer um reconhecimento do ambiente conhecido ou não onde o *Smartphone* encontra-se.

É modificado o perfil sonoro do dispositivo sempre que modificar o contexto. Com as devidas permissões de acesso a aplicação deverá realizar, de forma automática, a alteração do perfil do dispositivo para normal, vibrar ou silencioso. Essa função é realizada através de manipulação de componentes da plataforma, sendo necessário a utilização da classe *AudioManager* para obter acesso ao controle de áudio. As condições de habilitar ou desabilitar o modo *Bluetooth* são estabelecidas na modificação de contexto conforme a sugestão caracterizada pelo usuário e o acesso a essa funcionalidade é permitida através componentes da classe *BluetoothAdapter* encontrada na plataforma.

3.5. Construção por funcionalidades

A quinta fase é a fase final, e é onde cada funcionalidade é implementada, os códigos são revisados, para cada uma das *features*. O objetivo é obter a garantia de confiabilidade e segurança em todo sistema proposto [Palmer e Felsing 2012].

A implementação do sistema projetado inicialmente, foi construída com o *Android Studio*, um ambiente de desenvolvimento de *software* com um editor de *layouts* e componentes gráficos [Google 2015]. Escolhido por ser um *software* desenvolvido pela mesma empresa que desenvolveu o *Android* e por proporcionar uma instalação mais simples que outros ambientes para programação em *Android*.

A Figura 3 ilustra as interfaces gráficas para a manipulação do aplicativo.

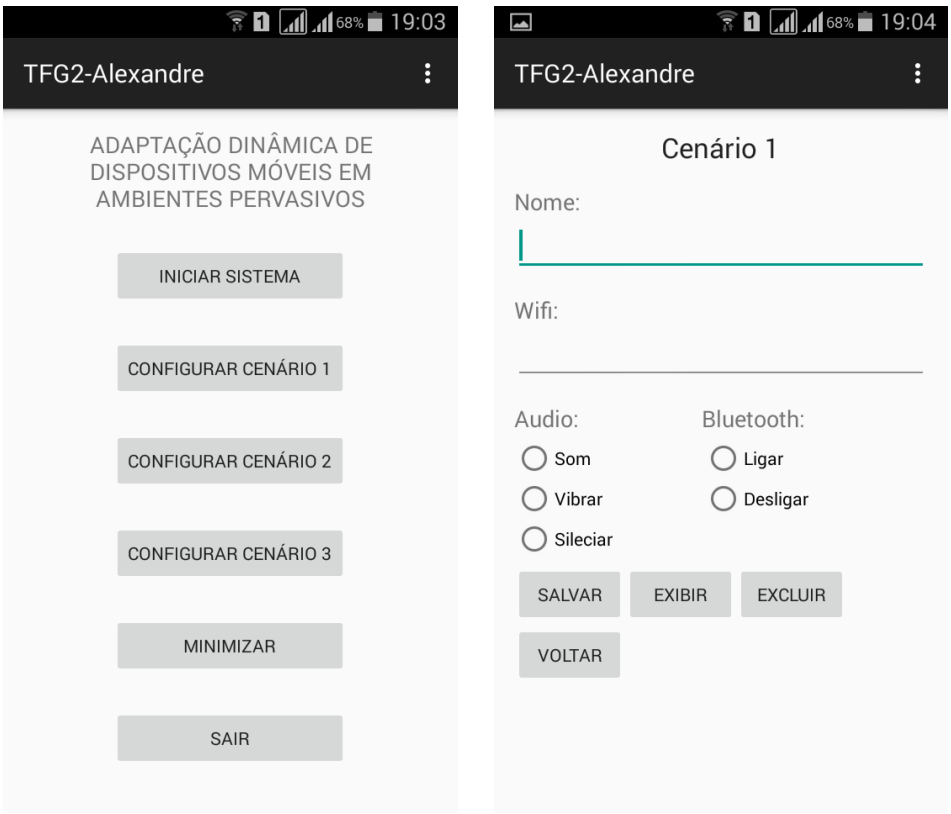

Figura 3 – Interfaces do aplicativo

A Figura 3 é composta pela Atividade principal e a Atividade de manipulação de dados do aplicativo, permitindo através destas a manipulação de cenários e o controle de execução do aplicativo.

Para a criação e armazenamento de contextos estabelecidos por características estabelecidas pelo usuário, foi construída uma interface de manipulação de dados (ilustrada na Figura 3), através de uma interface para receber valores inserido manualmente pelo usuário e posteriormente serem armazenados em arquivos.

O aplicativo faz uma busca constante (por intermédio de um método thread) para reconhecer qual rede *Wifi* o dispositivo móvel está conectado, com a intenção de reconhecer o ambiente. Para contemplar a verificação de rede no sistema do dispositivo uma classe (ilustrada na Figura 4) foi elaborada para a verificação de redes locais.

```
public class Wifi {

    private String wifiDispositivo = "";

    Context ctxwf;

    public Wifi(Context ctx){

        this.ctxwf = ctx;
    }

    public String getWifiDispositivo() {

        return wifiDispositivo;
    }

    public void setWifiDispositivo(String wifiDispositivo) {

        this.wifiDispositivo = wifiDispositivo;
    }

    public String VerificarWifiDispositivo(){

        WifiManager wifiManager = (WifiManager) ctxwf.getSystemService(ctxwf.WIFI_SERVICE);
        WifiInfo wifiInfo = wifiManager.getConnectionInfo();
        setWifiDispositivo(wifiInfo.getSSID());
        return getWifiDispositivo();
    }
}
```

Figura 4 – Classe *Wifi*

A Figura 4 é constituída de uma classe contendo um método que quando solicitado pela aplicação busca e armazena o nome da rede local em que o dispositivo móvel está conectado momentaneamente.

Para efetuar a ligação entre cenários cadastrados e o ambiente onde se encontra o dispositivo, os valores extraídos são armazenados em variáveis para comparações. Os dados são comparados constantemente com o auxilio do método que implementa um *thread* responsável por solicitar outros métodos de busca e comparação.

A condição de igualdade entre redes de ambientes e cenários cadastrados, motiva a aplicação a efetuar a mudança de comportamento no dispositivo, alterando seu status de áudio e *bluetooth* com a auxilio de classes implementadas pelo sistema operacional conforme as Figuras 5 e 6.

```
public class Audio {

    private AudioManager mAudioManager;

    Context ctxaud;
    public Audio(Context ctx){
        this.ctxaud = ctx;
        mAudioManager = (AudioManager) ctxaud.getSystemService(ctxaud.AUDIO_SERVICE);

    }

    public void AtivarNormalAudio(){

        mAudioManager.setRingerMode(AudioManager.RINGER_MODE_NORMAL);
    }

    public void AtivarVibrarAudio(){

        mAudioManager.setRingerMode(AudioManager.RINGER_MODE_VIBRATE);
    }

    public void AtivarSilenciarAudio(){

        mAudioManager.setRingerMode(AudioManager.RINGER_MODE_SILENT);
    }

}
```

Figura 5 – Classe Audio

A Figura 5 apresenta o código de uma classe com métodos que, ao serem solicitados, são responsáveis por efetuar a alteração do modo de áudio para o modo normal, vibrar ou silencioso, independente do estado que o modo se encontrava anteriormente.

```
public class Bluetooth {

    private BluetoothAdapter btAdapter;

    Context ctxbth;

    public Bluetooth(Context ctx){

        this.ctxbth = ctx;
    }

    public void AtivarLigarBluetooth(){
        btAdapter = BluetoothAdapter.getDefaultAdapter();
        if(!btAdapter.isEnabled()) {
            btAdapter.enable();
        }
    }

    public void AtivarDesligarBluetooth(){
        btAdapter = BluetoothAdapter.getDefaultAdapter();
        if(btAdapter.isEnabled()){
            btAdapter.disable();
        }
    }
}
```

Figura 6 – Classe *Bluetooth*

A Figura 6 ilustra a classe responsável por efetuar a modificação do modo *bluetooth* que por sua vez passará para o estado desejado pela aplicação.

4. Cenário de testes/Validação

Este trabalho teve como proposta o desenvolvimento de um aplicativo na plataforma *Android* com a intenção de mudanças automáticas de características de um dispositivo móvel quando o mesmo chegar a um ambiente conhecido.

Dentro desse contexto, o sistema foi desenvolvido para atuar em um dispositivo móvel e os testes foram executados em um *smartphone* (Modelo GT-I9192, Versão Android 4.4.2). O aplicativo foi instalado no dispositivo móvel e o mesmo esteve presente em diversos ambientes caracterizados com diferentes contextos.

Para fins de validação, foram cadastrados cenários informando a aplicação com valores de SSID de redes locais reais e com as características, ou seja, o perfil que o *smartphone* deve assumir quando conectado em uma dessas redes *wifi* conhecidas. A ideia de troca de perfis de acordo com o ambiente (rede *wifi)* em que o usuário se encontra pode ser visualizada na Figura 7.

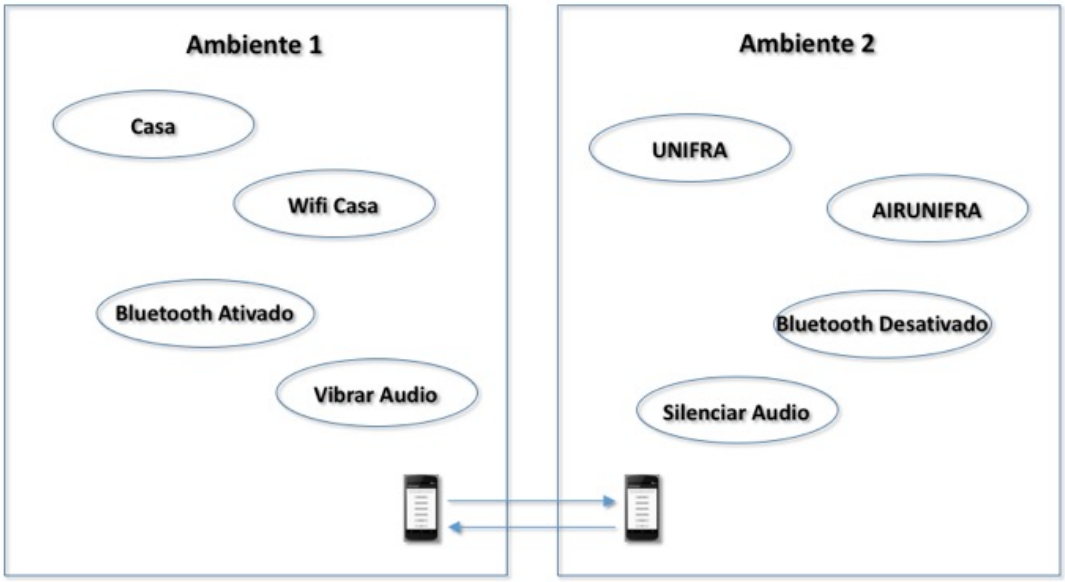

Figura 7 – Cenário de testes

Os valores ilustrados na Figura 7 representam dois ambientes diferentes caracterizados por serem dois lugares distintos geograficamente.

O *smartphone* atua no cenário de testes deslocando-se com o usuário entre o Ambiente 1 e o Ambiente 2 ilustrados na Figura 7. Contemplando os dados do aplicativo, para cada ambiente, foram cadastradas características correspondentes ao nome do local, o SSID da rede local e o modo de comportamento do áudio e do *Bluetooth* do *smartphone* que o usuário escolheu. Em cada um dos ambientes está presente uma rede local com SSID "Wifi Casa" para Ambiente 1 e "AIRUNIFRA" para Ambiente 2.

O objetivo é correspondido quando o usuário chegar em "Casa" (ilustrado nas Figuras 7 e 8 como "Ambiente 1") o *smartphone* automaticamente terá o *Bluetooth* ativado e o modo de áudio passará para "Vibrar". E quando o usuário chegar em

UNIFRA (ilustrado nas Figuras 7 e 8 como “Ambiente 2”) o *Bluetooth* será desativado e o modo de áudio passará para “Silenciar” no *smartphone*.

Durante a execução dos testes percebe-se que quando o *smartphone* altera seu comportamento ao se afastar ou se aproximar de um ambiente conhecido seu comportamento se adapta conforme ilustrado na Figura 8.

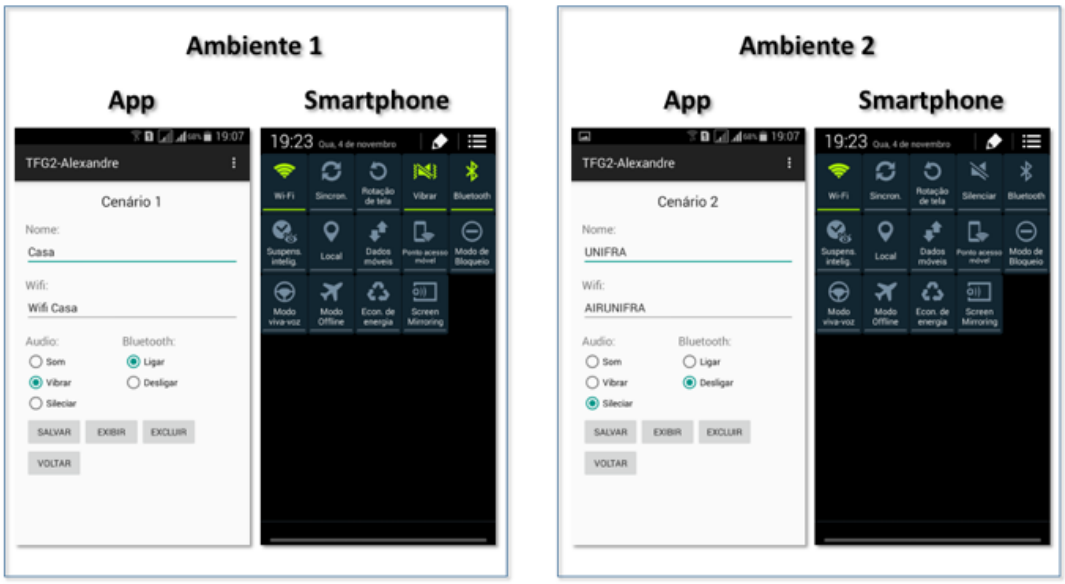

Figura 8 – Cenário de teste 2

A Figura 8 ilustra dois ambientes distintos e, consequentemente, para cada um apresenta uma configuração de cenário visualizada em “App” com seu respectivo comportamento em “Smartphone”.

5. Resultados e discussões

Todo o projeto esteve focado na criação de um sistema que, em conjunto com o ambiente no qual o usuário se encontra, permitisse que o dispositivo móvel alterasse o comportamento/configurações desse dispositivo de forma automática.

Após a implementação e os testes iniciais de validação, pode-se ser observar que a proposta de criação de um aplicativo desenvolvido para a plataforma Android é capaz de adaptar-se automaticamente, alterando características de um *smartphone,* realizando mudança de comportamento para ambientes caracterizados por um cenário pré-definido pelo usuário.

Além disso, a utilização de redes locais para identificar o cenário o qual o usuário encontra-se tornou viável o reconhecimento do ambiente de maneira que em determinadas situações de conflitos com o sinal dentro do cenário proposto, o dispositivo móvel interpretou como uma troca de ambiente alterando o seu comportamento. Entretanto, com o retorno do sinal, alterou novamente para o cenário do estado inicial.

Um dos desafios encontrados foi implementar o sistema em que a aplicação estaria sempre em busca do reconhecimento de um ambiente e, simultaneamente, novos cenários poderiam estar sendo criados ou alterados.

6. Conclusões e trabalhos futuros

Dentro da abordagem proposta neste trabalho, é possível destacar algumas vantagens no que refere-se ao tempo de resposta para a mudança do comportamento do dispositivo móvel, na veracidade das características pré-definidas e a praticidade de manusear a aplicação devido a um número otimizado de interfaces gráficas.

A metodologia ágil FDD escolhida para desenvolvimento deste trabalho, contemplou o desenvolvimento por funcionalidades, permitindo fazer testes frequentes, repetitivos e que abordassem diretamente a produção de sistemas utilizando mecanismos capazes de implementar e de acompanhar, inclusive de projetar novamente alguma funcionalidade a ser desenvolvida. Com isso, uma das principais contribuições da proposta é promover uma menor interferência dos dispositivos móveis em ambientes de comum acesso, permitindo um controle sonoro automático relacionado a lugares de rotina do usuário. Nesse contexto houve uma mescla da Computação Móvel com a Computação Pervasiva, numa tentativa de imersão e onipresença previstos por Weiser (1991).

No que tange aos trabalhos futuros, poderão ser desenvolvidas estratégias para controle de energia, inclusão dinâmica de novas características para o cenário de comportamento do dispositivo, tais como "Modo Viva-voz", "Economia de energia", "Rotação de tela" e demais itens encontrados no painel de notificações de um sistema *Android*. Ademais, poderão existir alterações no número de cenários pré-definidos pelo usuário, buscando ampliar a quantidade de cenários que o usuário deseja configurar suas necessidades. Por fim, essa aplicativo de gerenciamento e customização pode se tornar um produto, sendo explorado comercialmente.

Referências

Ahamed, S. I.; Zulkernine, M.; Haque, M. M. (2008) *"Security, Privacy, and Trust for Pervasive Computing Applications". Mobile multimedia communications: concepts, applications, and challenges,* [S.l.], p.327–342.

Araújo, R. B. D. (2003) "Computação Ubíqua: princípios, tecnologias e desafios". 21 Simpósio Brasileiro de Redes de Computadores, [S.l.], p.45–115.

Barbosa, A.; Azevedo, B.; Pereira, B.; Campo, P.; Santos, P. (2008) "Metodologia Ágil: *Feature Driven Development*". Faculdade de Engenharia da Universidade do Porto.

Barbosa, D. et al. (2005) "*GlobalEdu: An Architecture to Support Learning in a Pervasive Computing Environment*". Austrália: Springer-Verlag.

Barbosa, J.; Hahn, R.; Rabello, S.; Pinto, S. C. C. S.; Barbosa, D. N. F. (2007) "Computação Móvel e Ubíqua no Contexto de uma Graduação de Referência". Revista Brasileira de Informática na Educação: vol. 15, nº 3. Set. Dezembro.

Barth F. J.; Gomi, E. S. (2004) "Uma arquitetura para criação de interfaces adaptativas para televisão interativa". Curitiba.

Chen, G.; Kotz, D. (2000) "*A Survey of Context-Aware Mobile Computing Research*". [S.l.: s.n.].

Dey, A. K. (2001) *Understanding and Using Context. "Personal Ubiquitous Comput". London, UK, v.5, p.4–7, January.*

Junior, C.F e Fernandes, A.M. (2006) "Análise das tendências tecnológicas para Computação Móvel aplicada à área da Saúde". Brasil: Universidade do Vale do Itajaí.

Figueredo, C. M. S.; Nakamura, E. (2003) "Computação Móvel: Novas Oportunidades e Novos desafios". T&C Amazônia, Ano 1, nº 2, junho.

Freitas, L. O. (2011) "Uma metodologia para assistir pacientes em ambientes homecare pervasivos". Dissertação de Mestrado. Santa Maria: UFSM.

Google, (2013) "Android Developers". Disponível em http://developer.android.com/. Acesso em novembro.

Google, (2015) "*Android Developers*". Disponível em http://developer.android.com/. Acesso em agosto.

Librelotto, G. R.; Freitas, L. O.; Fiorin, A.; Mozzaquatro, B. A.; Pasetto, L.; Martini, R. G.; Azevedo, R. P. de; Pereira, R. T. (2011) "*Onto Health: a system to process ontologies applied to health pervasive environment*". *IEEE Computer Society*, p.59–64.

Mateus, G.R e Loureiro, A.F. (1998) "Introdução a Computação Móvel". Segunda edição. Minas Gerais.

Martini, R.G. (2012) "Uma abordagem para a personalização automática de interfaces de usuário para dispositivos móveis em ambientes pervasivos". Dissertação de Mestrado. Santa Maria: UFSM.

Palmer, S. R.; Felsing, M. (2002) "*A Practical Guide to Feature-Driven Development*". 1st. ed. [S.l.]: *Pearson Education*. ISBN 0130676152.

Papatheodorou, C. (2001) "*Machine Learning in User Modeling. Machine Learning and Applications. Lecture Notes in Artificial Intelligent". Springer Verlag*.

Schilit, B.; Theimer, M. (1994) "*Disseminating active map information to mobile hosts*". *Network, IEEE*, [S.l.] v.8, n.5, p.22-32, sep/oct.

Weiser, M. (1991) "*The computer for the 21st century". Scientific American*, [S.l.], september.

Yamin, A. C. (2004) "Arquitetura para um Ambiente de Grade Computacional Direcionado às Aplicações Distribuídas, Móveis e Conscientes do Contexto da Computação Pervasiva. 195p". Tese de Doutorado. Porto Alegre: UFRGS.

Apêndice 1

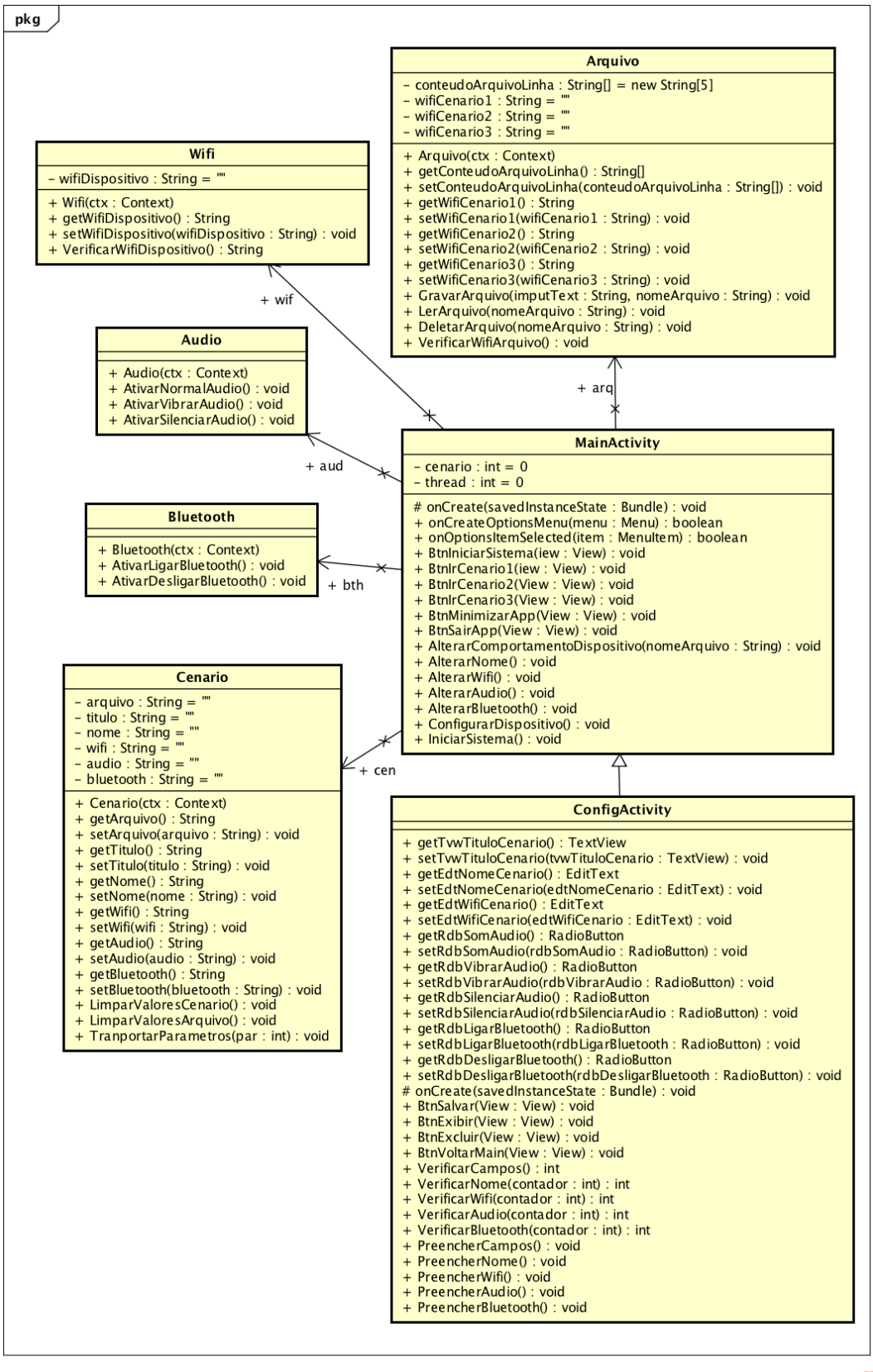

Figura 9 – Diagrama de Classes