APRIMORANDO A EXPERIÊNCIA DO USUÁRIO EM APLICATIVOS MÓVEIS ATRAVÉS DA MINERAÇÃO DE DADOS

Tássio de Oliveira Silva AUAD
Centro de Ensino Superior de Juiz de Fora, Juiz de Fora, MG
Luiz Felipe MENDES

Resumo: Com o avanço do mercado de aplicativos móveis e com os desafios encontrados em desenvolver produtos que atendam a inúmeros tipos de usuários, que harmonizem com os ambientes de uso e, consequentemente, proporcionem uma boa experiência ao usuário, surge a necessidade de se desenvolver um framework que torne as aplicações sensíveis ao contexto e realize o auto-aprendizado da navegação do usuário. Esse artigo apresenta uma base conceitual e o processo de atuação do TURAP, um framework para aplicações Android que as auxilia a enfrentar tais desafios levantados.

Palavras-chave: Experiência do Usuário, Mineração de Dados, Computação Sensível ao Contexto, Auto-Aprendizado, Aplicativos Móveis, Android.

1 INTRODUÇÃO

Todo e qualquer produto ou serviço que se faça uso gera uma experiência a quem realizou a interação, a chamada experiência do usuário. Após a interação com os produtos há uma reação no usuário que o fará aceitar ou rejeitar os aspectos do produto em mãos, sendo que essa decisão sofre influência de atributos externos, vindos do contexto de uso, e seus próprios aspectos internos do usuário naquele instante.

De uma forma resumida, Roto (2007) diz que pessoas, produtos e contextos são então protagonistas nos estudos de experiência do usuário e aspectos específicos de cada um são levantados e avaliados durante a interação, como por exemplo, do usuário são avaliadas motivações, estado mental e conhecimento, enquanto do produto são observados a infraestrutura, as funcionalidades e os elementos visuais e, finalmente, do contexto são avaliados o estado social, a cultura, o local físico e até mesmo a luminosidade pode influenciar na relação.

Dentre esses possíveis estudos a serem feitos, os que possuem fins de mensurar essa experiência procuram ter um maior enfoque no usuário, já que é em seu interior, nos fatores psicológicos, que acontece a tal experiência. Quando os fins são de aprimoramento, o maior foco passa a ser a interação e as alterações necessárias recaem sobre o produto. Se é desejado aprimorar a experiência do usuário, é neces-

sário adaptar o produto ao contexto de uso e as características do usuário para que seja natural ou intuitivo seu uso (ROTO, 2007).

Então, ter conhecimento do ambiente em que acontecerá a interação e de características específicas do público-alvo é então importante para se desenvolver uma experiência de uso. Dentro da Informática, a computação móvel é a área que mais possui desafios para realizar isso. Os dispositivos móveis trouxeram a liberdade com o acesso a recursos e aplicativos em qualquer lugar, e, como consequência, veio a dificuldade de se determinar o ambiente em que ocorrerão estas interações, de acompanhar as mudanças no ambiente causadas pela movimentação do usuário, e determinar quem estará interagindo.

E diante desse dinamismo e indefinições, como desenvolver um produto de modo que ele traga boa experiência? Para Schilit *et al* (1994), há certas coisas que os usuários regularmente fazem quando estão na biblioteca, cozinha ou escritório. Entender seus hábitos e rotinas é entender a relação entre contexto, usuário e produto, e nos possibilita chegar a conclusões como que o usuário “x” usa sempre o produto “y” de uma maneira “n” no contexto “k”. Isso nos leva a possibilidade de prever os desejos do usuário em um contexto de acordo com os padrões de navegação identificados, facilitar seu caminho até chegar ao que deseja e assim proporcionar a ele uma boa experiência.

O projeto propõem a criação do TURAP ou “*Tracker of Users’ Routines and Aware Pointer*”, um framework para aplicações Android que possibilita identificar as rotinas de interação dos usuários e, com esses padrões de uso, apontar a aplicação previsões de interações de acordo com o contexto. O framework tem como objetivo ajudar a instrumentalizar os profissionais envolvidos no desenvolvimento de aplicativos móveis a criarem produtos que gerem boas experiências de usuário.

2 A EXPERIÊNCIA DO USUÁRIO

No decorrer do dia as pessoas se tornam ‘usuários’ e tem experiências com vários objetos, como por exemplo, a cadeira, o carro, o controle remoto do ar condicionado, redes sociais, os talheres, o caixa eletrônico, o computador no trabalho, ou seja, objetos e produtos, digitais ou não, que são ‘usados’ e que são projetados para cumprir alguma função. Todos esses tipos de produtos ao serem utilizados geram reações no usuário, como sentimentos, pensamentos, sensações e vontades, que podem ser negativas ou positivas (TEIXEIRA, 2013).

Para Garrett (2010), a experiência do usuário é a experiência que o produto cria para a pessoa que o usa no mundo real. Simplificadamente, essa experiência é, como o próprio nome já diz, a experiência de uso de um produto ou a experiência que é gerada ao usar um produto qualquer.

Onde acontece a experiência é o interior do usuário, porém onde acontece a interação necessária para a experiência existir é no contexto. Garrett (2010) comenta que experiência do usuário não é sobre o funcionamento interno de um produto ou serviço, mas sim como funciona do lado de fora, onde uma pessoa entra em contato com o produto, ou seja, o contexto.

A respeito dessas composições, a palavra contexto nos remete de imediato a ideia de localização geográfica, porém, como afirmado por Schilit *et al.* (1994), o contexto abrange mais do que apenas isso, porque não é apenas a localização que modifica quando o contexto muda, mas também é possível citar como exemplos apontados por eles a iluminação, o nível de ruído, a conectividade de rede, os custos de comunicação, a largura de banda de comunicação, e até mesmo a situação social; por exemplo, se você está com o seu gerente ou com um colega de trabalho.

Abstraindo, para Schilit *et al.* (1994), o contexto pode ser definido à partir das respostas de três questões, onde você está, com quem você está e quais recursos estão próximos. Já sob o ponto de vista de computação móvel, Mendoza (2013) abstrai ao definir a composição do que ele chama de “equação móvel” ao dizer em dispositivos móveis deve ser levado em conta a rede, o tipo de dispositivo que se encontra, o sistema operacional e o tamanho da tela.

Mas não é só o contexto, o produto e o usuário, são também complexos e compostos por vários aspectos. Arhippainen e Tähti (2003) citam aspectos como valores, emoções, expectativas, experiências anterior e, na mesma afirmativa, para os produtos ele comenta características como mobilidade e adaptatividade.

3 A IMPORTÂNCIA DA EXPERIÊNCIA DO USUÁRIO

É comum encarregar as questões funcionais de um produto a responsabilidade de definir a sua qualidade, ou seja, um produto bem projetado é um com boas funções ou aquele que faz o que promete. As pessoas também pensam que um produto bem projetado possui um grande apelo estético ou que agrade aos olhos e com boa sensação ao toque. Todos os consumidores alguma vez escolheram um produto ao invés de outro pelo visual e, caso o usuário possua pouca idade, aspectos visuais

passam até mesmo a sobressair aos aspectos funcionais. Sendo assim, um produto bem elaborado também é um com bom visual (GARRETT, 2010).

Mas nenhuma dessas duas questões são suficientes para garantir um produto sem falhas em seu uso e assegurar o sucesso da marca, porque são visões incapazes de identificar erros na interação. A concepção de produtos com a experiência do usuário como um resultado explícito excede o estético ou funcional, buscando uma interação intuitiva e identificar possíveis falhas no relacionamento ou harmonia dos três elementos: usuário, produto e contexto. Melhorar a interação, é ajudar as pessoas a trabalharem mais rápido e ajudá-las a cometerem menos erros, ou seja, melhorar a interação é buscar eficiência no uso (GARRETT, 2010).

O produto pode falhar em proporcionar uma experiência satisfatória por não estar harmonizado com o contexto, como no exemplo dado por Garrett (2010), uma cadeira poderia ser revestida de couro, dando um apelo visual agradável, realizar bem sua funcionalidade de permitir que alguém se sente, mas se o contexto, que é uma palestra sobre dietas onde a maioria dos participantes está acima do peso, for ignorado e a cadeira não foi desenvolvida pensando na possibilidade do produto ser usado nesse contexto por esse tipo de usuário, ela pode se quebrar e levar uma má experiência as pessoas.

Além de não garantirem boa eficiência na interação, o foco nas funcionalidades e o apelo visual também não são suficientes para determinar a lealdade de uma pessoa, nem mesmo que ele saia da condição de visitante e se torne um usuário. A pessoa que vive uma má experiência com um produto não voltará a usá-lo e, caso encontre uma experiência positiva no produto concorrente, passará a ser usuário dele. Nesses casos negativos, aconselha-se a não esperar uma segunda chance para acertar, pois nem mesmo com uso de tecnologias sofisticadas ou marketing trarão o usuário de volta. Então, características e funções sempre importam, mas a experiência do usuário tem um efeito muito maior sobre a lealdade do usuário (GARRETT, 2010).

Essas reações negativas são entendidas quando pensa-se no comportamento do usuário que não conseguiu realizar o que desejava. Um usuário que não consegue seguir o processo para chegar ao resultado traz para si o sentimento de culpa e vergonha. Um resultado obtido que esteja errado terá como consequência uma acusação por parte do usuário a funcionalidade do produto, mas quando o problema passa a ser não conseguir chegar ao resultado, comumente o usuário trará a culpa para si e nunca culpa o produto por ter uma interação pouco intuitiva ou pelo produto

não ser eficiente. Os usuários sentem como se tivessem feito algo errado, eles passam a acreditar que não estavam prestando a atenção suficientemente. (GARRETT, 2010).

Garret (2010) afirma que quanto mais fácil de usar e mais natural a interação das ferramentas de trabalho, mais as pessoas gostarão de seus empregos, sendo o inverso também verdadeiro, ou seja, quanto mais frustrante e desnecessariamente complexa a ferramenta, mais frustradas ficarão as pessoas. Esses tipos de ferramentas podem fazer a diferença entre voltar para casa satisfeito no final do dia e voltar para casa exausto e odiar seu trabalho. Um software que seja utilizado de forma intensa durante todo o dia de trabalho de um funcionário e que não seja natural o seu uso pode causar um estresse acumulado no final do dia. Um usuário se culpando de não conseguir alcançar o resultado somado à talvez prazos apertados e reclamações de baixa produtividade podem tornar desagradável sua rotina. Então, a experiência de uso pode ser o divisor entre o funcionário ir para casa infeliz ou satisfeito com o seu trabalho ou satisfeito com o que faz.

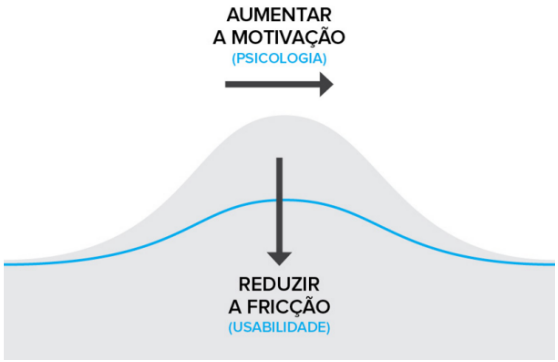
Gualtieri (2009) traz algumas estatísticas a respeito das melhorias que uma experiência do usuário bem projetada pode trazer. Em média, as empresas que fornecem uma experiência mais rica têm 14,4% mais usuários que estão dispostos a considerá-las para uma nova compra do que empresas do mesmo setor que oferecem uma experiência mais pobre. Além disso, em comparação com as empresas que oferecem uma experiência mais pobre, as que oferecem a melhor experiência têm 15,8% menos usuários que estão propensos a pensar em fazer negócio com um concorrente. E por fim, as empresas com as maiores pontuações de experiência têm 16,6% mais usuários que estão propensos a recomendar seus produtos ou serviços do que os seus concorrentes de menor pontuação.

Então, a importância da experiência do usuário está no fato de que desenvolver apenas os recursos funcionais e visuais não garantem o sucesso da marca, se estendendo ao caso que uma má experiência causa reações desagradáveis ao psicológico de quem faz uso e faz com que o usuário perca sua lealdade ao produto tendendo a adotar o da concorrência. O resultado final desses fatores negativos sobre o usuário é a perda de lucro e torna a sobrevivência da empresa em um mercado competitivo pouco possível.

4 APRIMORANDO A EXPERIÊNCIA DO USUÁRIO

Para um projetista da experiência do usuário, o mais importante é definir como as pessoas irão interagir com produto, trabalhando para construir produtos que sejam fáceis de usar (usabilidade), reduzindo a fricção na interação e permitindo que os usuários completem a tarefa desejada em menos tempo, com menos ruído e obstáculos, ao mesmo tempo que esses profissionais apoiam-se em princípios da psicologia para motivar o usuário e incentivá-lo a seguir adiante (TEIXEIRA, 2013). A figura 1 ilustra os fatores psicológicos.

FIGURA 1 - Procedimento para Alcançar a Eficiência na Interação



Fonte: Teixeira (2013, p.4)

Roto (2007) diz que para melhorar a experiência de uso do produto, o foco são as partes que compõem o sistema, ou seja, o objetivo deve ser aperfeiçoar o produto, para que seus aspectos não criem atritos na interação, mas catalisem o processo de chegada ao objetivo. Apesar da maior atenção ser voltada para o produto, o contexto e o usuário não podem ser esquecidos, já que não existe experiência sem ambos e também porque se é desejado eliminar atritos na interação o sistema deve ajustar-se ao contexto bem como às necessidades e expectativas atuais do usuário.

Então, chega-se a um modelo de aprimoramento de experiência de uso. Deve-se estimular o usuário a percorrer o caminho de interações até realizar o que deseja com o uso da psicologia e reduzir possíveis atritos nesse percurso através da melhoria da usabilidade com adaptação do produto expectativas do usuário e ao contexto de uso. Sendo assim, o usuário falhará menos, realizará suas tarefas com maior eficiência e possivelmente terá uma ótima experiência.

Mas há um problema para realizar isso em aplicações móveis. Entender o público-alvo suficientemente é um desafio, assim como supor o contexto em que o

relacionamento irá acontecer. Quando uma aplicação móvel é desenvolvida e disponibilizada em lojas online como Google Play e App Store, facilita-se o acesso ao produto e consequentemente há um aumento significativo no número de possíveis contextos de usos e de tipos de usuário.

Schilit *et al.* (1994) confirmam o problema com o contexto ao dizer que neste modelo de computação não ocorre em um único local, em um único contexto, como em computadores de mesa, mas abrange uma infinidade de situações e locais que cobrem o escritório, sala de reuniões, em casa, aeroporto, hotel, sala de aula, mercado, ônibus, etc. Dourish (2004) diz que os cientistas sociais têm argumentado que o desenvolvimento tradicional de sistemas, muitas vezes falha ao responder ao cenário em que se desenrola a ação, ou seja, os sistemas são planejados sem buscar uma adaptação ao contexto e suas mudanças de aspectos, e que tem surgido uma necessidade dos sistemas se ajustarem a diferentes configurações sociais.

Para que esses sistemas se ajustem automaticamente ao contexto, uma possível solução é dizer que quando computação é movida 'fora do ambiente de trabalho' é preciso manter o controle para onde ela foi. Unindo seus dois pensamentos, é possível resolver a dinamicidade do contexto através do controle ou rastreamento de onde a aplicação está sendo usada, usar esses dados para fazer com que ela se ajuste a cada mudança desse contexto e assim resultar na melhoria da usabilidade e na redução do atrito de uso (DOURISH, 2004).

Mas ainda resta a identificação das expectativas e desejos de uma gama pouco definida de usuários. Isso pode ser resolvido através da identificação das rotinas dos usuários e da antecipação de suas ações, o que fará com que seja atendido suas expectativas.

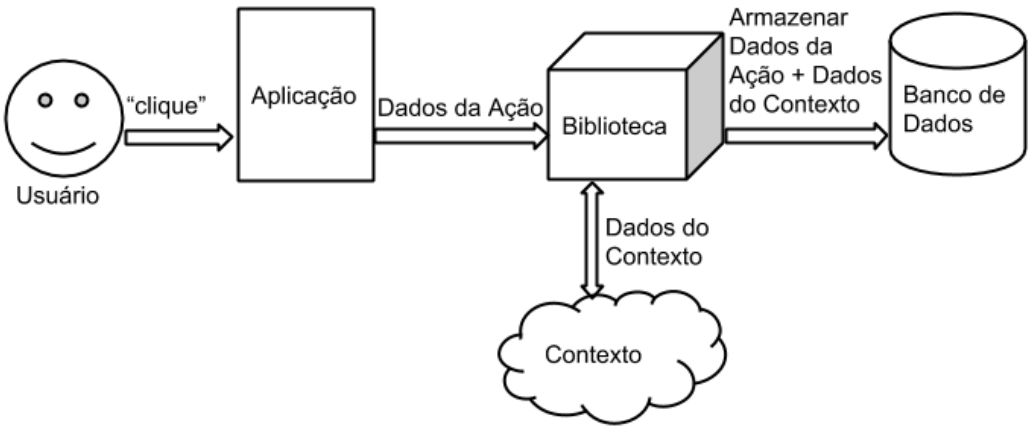
Resolver a dinâmica do contexto através de acompanhamento das mudanças de ambiente e adaptação em tempo real auxilia a resolver a necessidade de entender as expectativas do usuário, que estão ligadas ao contexto. Uma mudança de ambiente pode resultar em mudanças de necessidades e essas podem ser entendidas através de padrões de uso no contexto. A solução para o problema das variações está no rastreamento do contexto e na correta adaptação, em tempo real da usabilidade do software baseada nos padrões de uso em cada ambiente.

5 ATUAÇÃO DO FRAMEWORK NA EXPERIÊNCIA DO USUÁRIO

Diante desses aspectos dinâmicos dos elementos que compõem a experiência em aplicativos móveis, e de uma possível solução levantada, é proposto um framework com três papéis ou funções que encapsulam a solução e leva as aplicações móveis uma oportunidade de capturar estas informações de forma estruturada.

O primeiro papel é de armazenar dados do contexto juntamente com um dado que esteja relacionado a alguma ação que o usuário tomou durante o uso do aplicativo naquele ambiente, como está ilustrado na figura 02. Por exemplo, ao usar um aplicativo de leitura de e-books, o usuário realiza um clique no botão de modificar modo de leitura para noturno e, então, o framework fica encarregado de capturar dados do ambiente daquele momento, como por exemplo, o fato de estar a noite e de estar em uma localização que possivelmente não há muita luminosidade, e armazená-los juntamente a um valor que informe o acionamento do modo noturno.

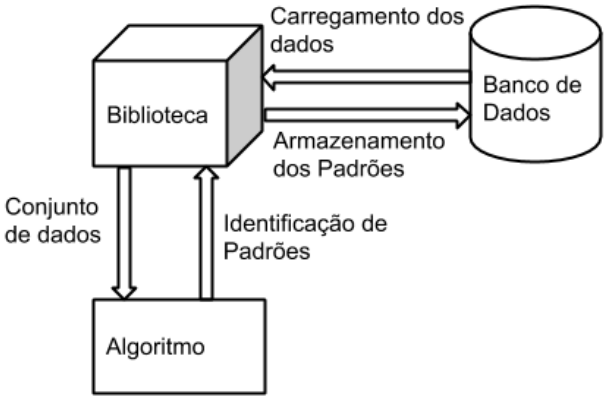
FIGURA 2 - Primeira Função do Framework



Fonte: Elaborado pelo autor

Com o passar do tempo de uso, haverá vários tipos de ambientes e de ações armazenados, cada ambiente relacionado a ação que foi tomada nele. Através desse conjunto de dados e do uso de algoritmos específicos, pode-se definir padrões de uso de um aplicativo em cada contexto da rotina do usuário, como por exemplo, o acionamento do modo noturno no contexto “noite” não aconteceu apenas uma vez, mas é uma rotina diária, é um padrão de ação do usuário naquele contexto. O segundo papel do framework é então identificar padrões no conjunto de dados criado pela execução de sua primeira função e armazená-los para possibilitar que o terceiro papel seja realizado, como ilustrado na figura 03.

FIGURA 3 - Segunda Função do Framework.

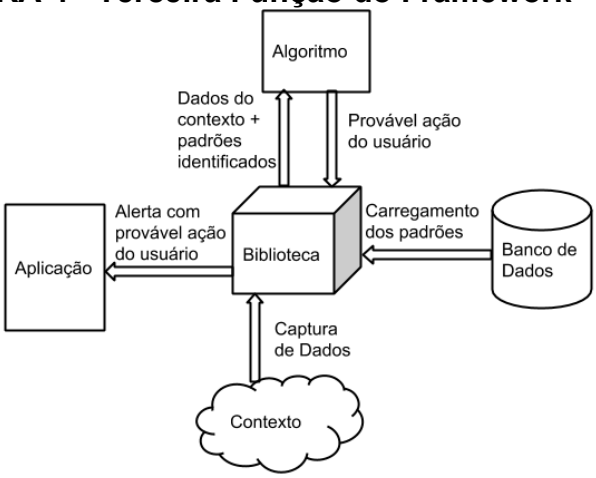


Fonte: Elaborado pelo autor

Se um usuário sempre aciona o modo noturno durante a noite, é porque ele possui a necessidade de que isso aconteça ou possui uma expectativa de realizar isso. Com esses padrões levantados é possível definir necessidades ou expectativas de uso de um usuário em cada contexto de sua rotina e, sabendo disso, quando um contexto dentro de um padrão se repetir, é possível informar a aplicação qual tipo de ação aquele usuário geralmente toma.

A terceira e última função do framework é de estar atento as mudanças do contexto e, sabendo dos padrões, de avisar a aplicação qual ação o usuário provavelmente tomará no contexto em que ele está. Com esse aviso, a aplicação poderá ser adaptar aos desejos do usuário, mudando suas configurações de navegação. A figura 04 ilustra tal procedimento.

FIGURA 4 - Terceira Função do Framework



Fonte: Elaborado pelo autor

Seguindo o exemplo, quando for a noite, o framework ficará encarregado de enviar um alerta a aplicação de que o modo noturno de leitura é uma ação comum naquele contexto e, sendo assim, a aplicação poderá tomar uma atitude para facilitar

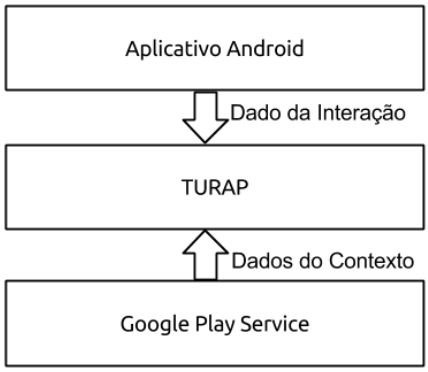
o acesso do usuário a essa ação, reduzindo assim a fricção no uso do aplicativo e tornando a interação de leitura mais agradável e motivador ao psicológico.

5.1 TURAP

Seguindo o modelo genérico de como um framework poderia atuar satisfazendo a solução teórica, o TURAP faz uso de tecnologias da Google e de algoritmos de mineração de dados já conhecidos na bibliografia de referência para alcançar os objetivos propostos.

A primeira função do modelo exige que o dado recebido da aplicação seja armazenada juntamente com dados do contexto. Os dispositivos móveis possuem sensores que possibilitam a captura de dados do contexto, e, em específico para a plataforma Android, o Google Play Service disponibiliza as APIs para a captura dos dados do contexto. O TURAP faz uso dessas APIs como ilustrado na figura 5.

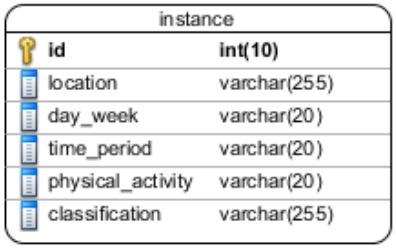
FIGURA 5 – USO DO GOOGLE PLAY SERVICES



Fonte: Elaborado pelo autor.

Dentre as APIs que compõem esse serviço da Google, o TURAP faz uso da *Activity Recognition API* que captura a atividade física do usuário, como por exemplo, se o usuário está andando ou parado, e também faz uso do *Location APIs* através da classe *LocationListener* para captura da localização geográfica. Também como informação do contexto é indentificado através das classes *Calendar* e *Date* do SDK da linguagem Java o horário e convertido para períodos do dia, como por exemplo, manhã, tarde e noite, e também é identificado o dia da semana. Por fim, chega-se a um modelo de tabela de banco de dados representada pela figura 6. O TURAP faz uso do banco de dados SQLite, possuindo uma instância própria, separado das aplicações que fizerem seu uso, por questões de sigilo das informações.

FIGURA 6 – MODELO DE TABELA



Fonte: Elaborado pelo autor.

No TURAP, o levantamento de padrões no conjunto de dados e a identificação da possível ação do usuário naquele contexto são procedimentos que são realizados com o auxílio da mineração de dados. Diante das várias tarefas que existem dentro desse campo como descrição, agrupamento entre outras, para o escopo do problema citado, a classificação foi a escolhida por se adequar melhor ao escopo.

Camilo e Silva (2009) definem classificação como uma das tarefas mais comuns que visa identificar a qual classe ou categoria um determinado registro pertence. Para esse campo, há sempre um conjunto de registros ou dataset, onde um registro é chamado de instance e cada registro é composto por campos ou atributos chamados attributes, que possuem um valor. Comparando essa estrutura com uma tabela, a própria tabela seria um conjunto de registros, e um registro seria uma linha da tabela, um atributo seria uma coluna e o valor de um atributo seria uma célula, como demonstra a figura 7 que demontra uma tabela do banco de dados do TURAP.

FIGURA 7 - Analogia de um *Dataset* com uma Tabela

id	location	day_week	time_period	physical_activity	Classification
0	40.712784 -74.005941	MONDAY	MORNING	STILL	JAZZ
1	40.712784 -74.005941	MONDAY	MORNING	STILL	JAZZ
2	40.713241 -74.015962	MONDAY	NIGHT	STILL	ROCK
3	40.713241 -74.015962	MONDAY	NIGHT	IN_VEHICLE	BLUES
4	40.713241 -74.015962	MONDAY	NIGHT	STILL	ROCK

Fonte: Elaborado pelo autor

O motivo da escolha dessa tarefa está na relação da classificação com o projeto proposto. Há a necessidade de classificar o contexto do momento ou de associar o tal contexto a uma ação do usuário. O que categoriza um contexto é um dado da

navegação do usuário ou, como no exemplo citado anteriormente, a classificação do registro onde o atributo “período do dia” é igual a “noite” possui o valor “modo noturno”.

As técnicas de classificação podem ser supervisionadas e não-supervisionadas e que são usadas para prever valores de variáveis do tipo categóricas. Pode-se, por exemplo, criar um modelo que classifica os usuários de um banco como especiais ou de risco e um laboratório pode usar sua base histórica de voluntários e verificar em quais indivíduos uma nova droga pode ser melhor ministrada. Em ambos os cenários um modelo é criado para classificar a qual categoria um certo registro pertence, porém, o modelo do banco não possui um histórico de usuários já categorizados para servir como base para categorizar os novos, o que ocorre no modelo do laboratório (CAMILO e SILVA, 2009).

A classificação supervisionada é a análise do conjunto de registros fornecidos, com cada registro já contendo a indicação à qual classe pertence, a fim de ’aprender’ como classificar um novo registro. O modelo do banco seria uma classificação não supervisionada, enquanto o modelo do laboratório seria uma classificação supervisionada (CAMILO e SILVA, 2009). O framework proposto segue o mesmo princípio do modelo do laboratório, ter uma base de dados história com registros já classificados, e, através dele, será usada a tarefa de classificação supervisionada para categorizar novos registros.

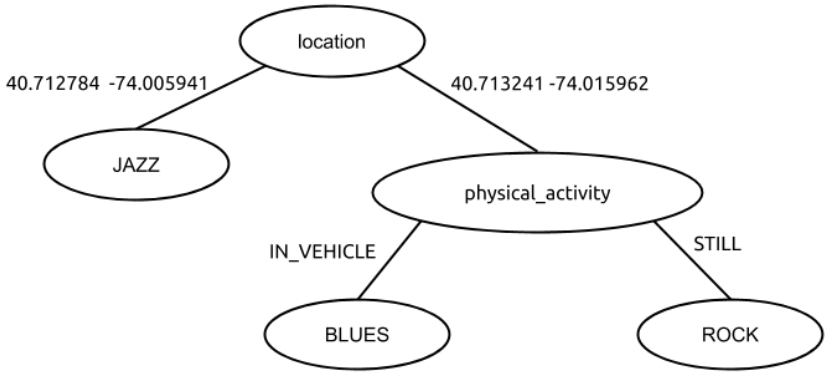
No *dataset* desse tipo de classificação há um atributo especial chamado de “*class*” que conterá o valor indicador da categoria a qual o registro pertence. E, sendo assim, cada registro que o framework realizar conterá atributos e valores do contexto e um atributo *class* contendo uma cadeia de caracteres representando uma ação do usuário que será fornecida e reconhecida pela aplicação.

O primeiro papel do framework então tecnicamente passa a ser capturar dados do contexto e adicionar em seus devidos campos do registro e receber uma palavra-chave de identificação da interação que o usuário realizou na aplicação para adicioná-la ao campo referente a classificação daquele registro. Essa palavra-chave é de reconhecimento da aplicação e o framework independe de conhecê-la para funcionar e, sendo assim, há uma flexibilidade a adaptação do framework a vários escopos de aplicações.

A segunda função determina a criação de padrões a partir desses dados armazenados. Os padrões estarão em formato de árvores de decisão, ou seja, um conjunto de regras em formato de árvore que serão criados através do algoritmo “*Top-*

Down Induction of Decision Trees" ou TDIDT, responsável pela grande adoção desse método pela sua simplicidade e poder. Camilo e Silva (2009) dizem que o sucesso das árvores de decisão deve-se ao fato de ser uma técnica extremamente simples, não necessitar de parâmetros de configuração e geralmente tem um bom grau de assertividade. A figura 8 ilustra uma árvore de decisão gerada a partir do *dataset*.

FIGURA 8 - Exemplo de Árvore de Decisão

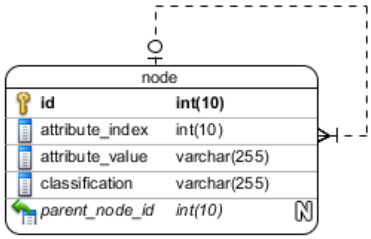


Fonte: Elaborado pelo autor

A escolha das árvores de decisão está ligada ao fato de que suas funções além de satisfazerem a necessidade de classificar registros, estão de acordo com o tipo de dispositivo onde todo o processamento ocorrerá. Bramer (2013) relata que árvores de decisão possuem duas funções, comprimir os dados em regras e assim possibilitar previsões. Todo o conjunto de dados será transformado em um pequeno conjunto de regras compatíveis e que podem representar os dados, assim comprimindo o tamanho sem perder a representatividade. Considerando que o framework funcionará em dispositivos móveis, a possibilidade de comprimir os dados em regras é de grande valia devido as limitações na capacidade de hardware desses dispositivos, além de facilitar seu armazenamento.

As árvores de decisão, diferente dos outros métodos, se adequam a imposição do modelo de que o procedimento de identificação de padrões seja separado do procedimento de classificação, para que sejam realizados em momentos diferentes e mais oportunos. Cada nó da árvore gerada é armazenado como um registro no banco de dados e estando relacionado com o registro do seu nó pai. O modelo dessa tabela de banco de dados está ilustrado na Figura 9.

FIGURA 9 – TABELA PARA ARMAZENAMENTO DE NÓS DA ÁRVORE



Fonte: Elaborado pelo autor.

Um registro da tabela *node* pode ser tanto um nó folha, contendo a classificação na coluna *classification*, como pode ser um nó intermediário ou pai, que conterá uma regra constituída pela coluna *attribute_index* e *attribute_value*. A coluna *attribute_index* recebe o número da coluna da tabela *instance* a quem o nó referencia e a coluna *attribute_value* é o valor necessário para que seja verdadeiro a verificação no nó.

Para Bramer (2013), o algoritmo responsável por criar essas árvores, o TDIDT, promete gerar as regras de classificação através do particionamento recursivo do conjunto de dados tendo como parâmetro os valores de cada atributo, formando assim vários subconjuntos a cada divisão. Ele descreve a forma básica do algoritmo da seguinte forma:

> SE todas os registros no conjunto de dados pertencerem a mesma classificação
> ENTÃO
> (a) Retorne o valor da classificação.
> SENÃO
> (a) Selecione um atributo A para subdividir o conjunto de dados
> (b) Subdividir o conjunto de dados, um subconjunto para cada valor do campo selecionado.
> (c) Retornar uma árvore com um braço para cada subconjunto não-vazio, sendo que cada braço deve conter uma subárvore descendente ou o valor da classe produzida pela aplicação recursiva do algoritmo.

O algoritmo TDIDT garante formular uma árvore de decisão que verdadeiramente corresponde a um *dataset*, mas para isso é necessário que esse conjunto de dados satisfaça a condição de adequação, que impõe que duas instâncias com mesmo valores em seus atributos não devem possuir classificações diferentes. Outra condição do algoritmo é que um atributo nunca deve ser selecionado duas vezes em um mesmo braço durante o processo de classificação (BRAMER, 2013).

Sendo baseado no algoritmo do TDIDT e seguindo essas regras, o algoritmo de criação dos padrões usado no framework possui algumas modificações devido a arquitetura de classes. Além disso, o algoritmo considera a possibilidade de retornar

ilimitadas classificações caso as características do contexto estejam de acordo com vários padrões e possa ser associados vários tipos de categoria. Segue o algoritmo:

(1) Selecionar o melhor atributo para subdividir o conjunto.
(2) Para cada tipo de valor que existe no conjunto para esse melhor atributo:
    a. Subdividir o conjunto usando como parâmetro o atributo selecionado e o valor do momento.
    b. Se no novo subconjunto o atributo referente a classificação possuir um único tipo de valor em todos os registros.
        i. Criar um nó com o atributo selecionado e o valor do momento, adicionar no atributo referente a classificação o valor que é repetido em todos os registros e indicar esse nó como folha.
    c. Senão
        ii. Criar um nó com o atributo selecionado e o valor do momento.
        iii. Repetir todo o processo para capturar nós desse subconjunto e adicionar como filho ao nó gerado nessa etapa.

O TDIDT é genérico e sua falta de especificidade pode ser vista na etapa de escolher um atributo para dividir o conjunto de instâncias em vários subconjuntos, onde o algoritmo não especifica como realizar tal tarefa. Existem métodos para essa seleção e um deve ser adotado para suprir esse necessidade do algoritmo, mas o fato de podermos usar qualquer método, desde que satisfaça a condição de adequação, para criar um árvore de decisão não torna a escolha de um deles irrelevante.

Em Bramer (2013) a escolha desse método para selecionar um atributo para servir de base no particionamento do conjunto de registros passa a ser crucial quando entendemos que existem métodos mais úteis que outros, com soluções melhores que outros, e que influenciarão na precisão das regras a serem geradas ao final. Além disso, as experiências mostram que a precisão de uma árvore aumenta de acordo com que seu tamanho diminui, e isso é importante levando novamente em conta a limitação de hardware de dispositivos móveis. Então, a relevância da escolha de um método a ser usado para selecionar atributos no algoritmo TDIDT está relacionada a qualidade da solução, e essa qualidade está relacionada com o tamanho da árvore de decisão final do algoritmo.

Diante desses fatores, o método escolhido foi a tal seleção através de entropia. Não existem garantias de que através do uso da entropia o número de regras geradas será pequena, mas em todos os casos o número de decisões levantadas para árvore de decisão através da entropia é menor do que em qualquer outro crité-

rio de seleção de atributos. A entropia é uma medida de informação teórica a respeito da incerteza de um conjunto de dados, devido a presença de mais de um tipo de classificação no conjunto. Em outras palavras, a entropia é uma medida da falta de homogeneidade de classificação dos registro do conjunto (BRAMER, 2013).

Para se calcular a entropia, usa-se a fórmula E = $\sum_{i=1}^{k} pi\ log_2\ pi$. No conjunto de dados há K classes e para cada classe "i" há o "pi" que é o número de ocorrências de registros ligados a essa classe no conjunto dividido pelo número de registros do conjunto. Esse cálculo será usado na parte indefinida de seleção de atributo do algoritmo TDIDT. Através dele é possível identificar qual o atributo será usado para subdividir o conjunto de forma que a classificação dos registros seja mais homogênea nos subconjuntos, ou seja, definir qual atributo deve ser usado para trazer maior ganho de informação quando houver a subdivisão.

Dado um atributo, deverá ser subdividido os registros do conjunto em um subconjunto de registros para cada valor existente para esse atributo. Após isso, será somado a entropia de cada um desses subconjuntos multiplicadas pelo número de registros no subconjunto que é dividido pelo número de registros no conjunto. Ao final, teremos a nova entropia, a entropia caso esse atributo seja usado como parâmetro para subdivisões. Quando a entropia do conjunto for substraída por essa nova entropia, o resultado será o ganho de informação. Para se escolher o melhor atributo para realizar a subdivisão, basta escolher o que tiver como resultado o melhor ganho de informação (BRAMER, 2013).

Unindo o algoritmo do TDIDT e a Entropia para suprir a indefinição que há, teremos um método para se definir as árvores de decisão que conterão as regras ou padrões de uso das aplicações em cada contexto da rotina dos usuários. Basta então que essas regras sejam armazenadas no banco de dados e carregadas quando houver a classificação do contexto, que é a terceira função.

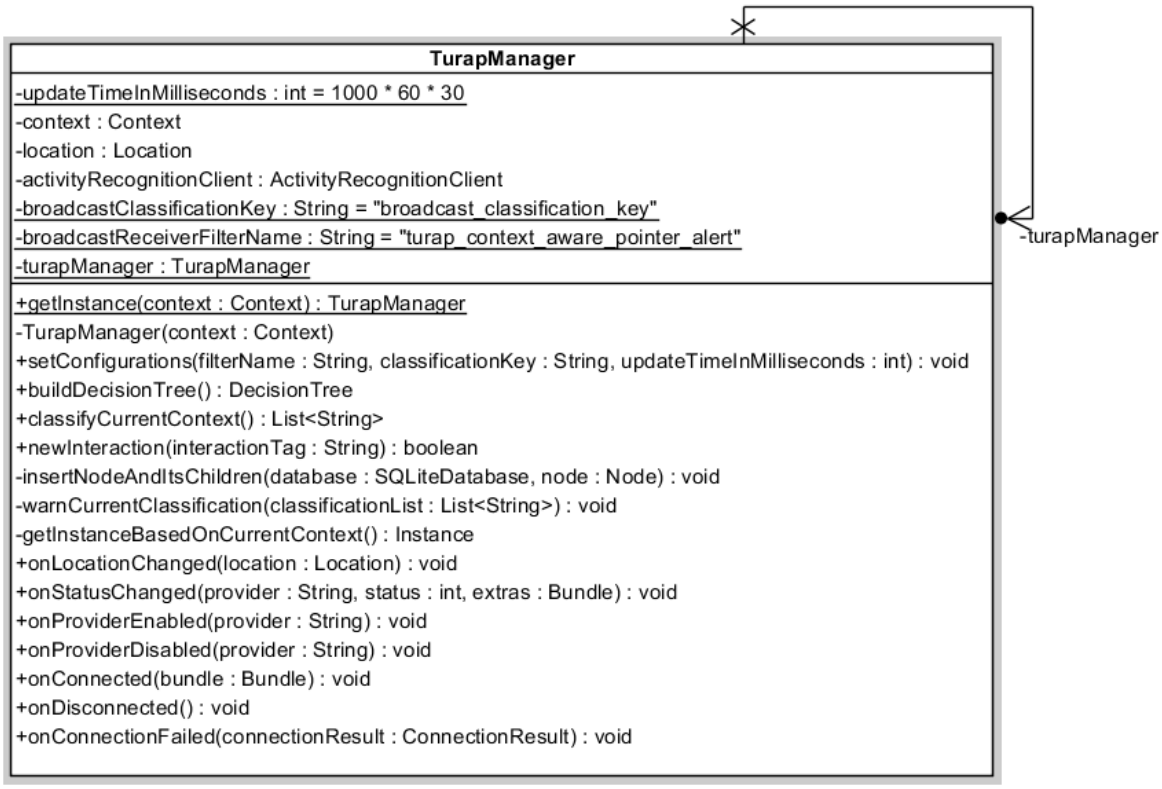
A realização do terceiro papel ocorre através do mecanismo denominado *Broadcast,* onde é possível enviar mensagens referenciadas por uma palavra-chave ou filtro a aplicação que possua um receptor configurado para receber transmissões ligadas ao tal filtro. Por padrão, o framework realizará a transmissão da classificação a cada 20 minutos ou quando o dispositivo se deslocar mais do que 100 metros no globo terrestre. Também por padrão, o filtro do disparo é *turap_context_aware_pointer_alert* e a palavra-chave que referencia a mensagem é *broadcast_classification_key*. Estas configurações podem ser modificadas através do

método *setConfigurations* da classe *TurapManager*, ilustrada na figura 10, permitindo que o desenvolvedor use filtros que apenas sua classe receptora conheça.

As aplicações que desejem fazer uso do TURAP, devem possuir uma sub-classe de *BroadcastReceiver,* classe pertencente ao SDK do Android com função de recepção de broadcasts, e também devem informar no arquivo de configuração *AndroidManifest.xml* o relacionamento do filtro, que pode ser padrão ou configurado, com a classe criada.

É importante salientar que apesar do funcionamento do framework possibilitar o auto-aprendizado das aplicações conforme o seu tempo de uso, a precisão dos padrões não é de 100% e as previsões do futuro através da classificação não podem ser consideradas certas de acontecerem.

FIGURA 10 – CLASSE PRINCIPAL DO FRAMEWORK TURAP



Fonte: Elaborado pelo autor

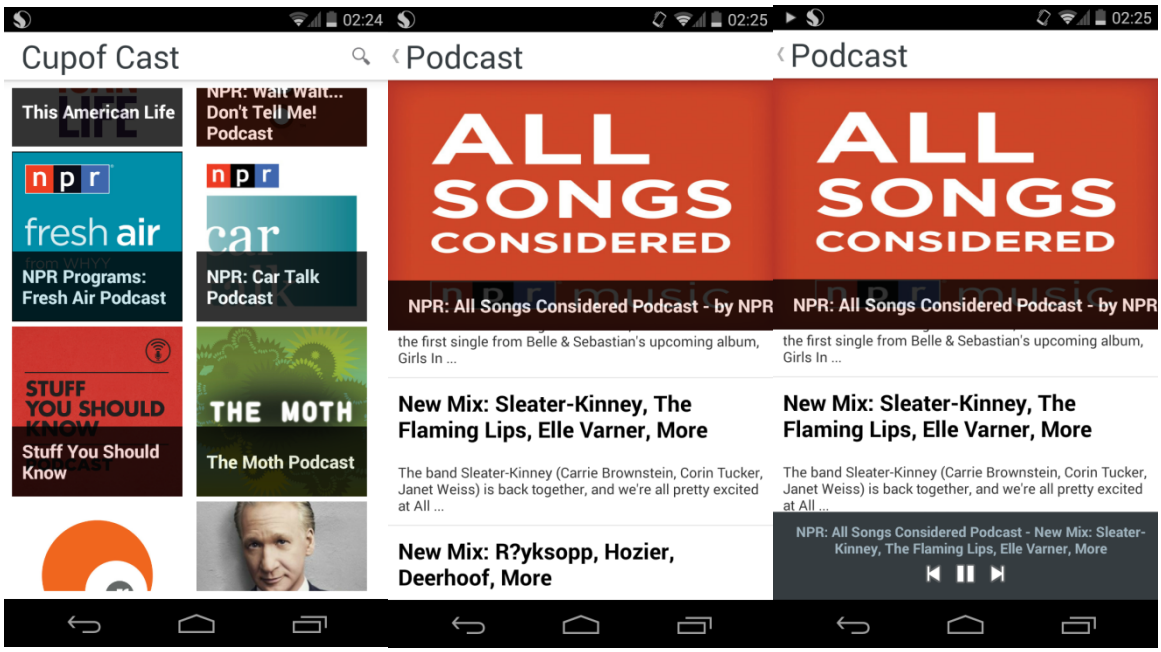
6 ESTUDO DE CASO

Como estudo de caso do uso do framework, foi desenvolvido a aplicação Android chamada *Cupof Cast*, que tem como objetivo possibilitar que pessoas busquem e escutem *podcasts*, que são séries de arquivos de áudio publicados ou uma espécie de programa de rádio publicados na internet. Esse aplicativo faz uso do framework

TURAP, além do *iTunesConnection*, um outro framework criado durante o desenvolvimento desse aplicativo para conversa com o *webservice* do *iTunes*, e uso de outros frameworks.

Na Figura 11, a tela da esquerda é a principal, pois, é onde os *podcasts* recomendados ou os mais ouvidos são exibidos. A tela central mostra um *podcast* selecionado, sendo responsável por exibir detalhes do mesmo e todos os episódios. A tela da direita mostra o episódio que está sendo tocado.

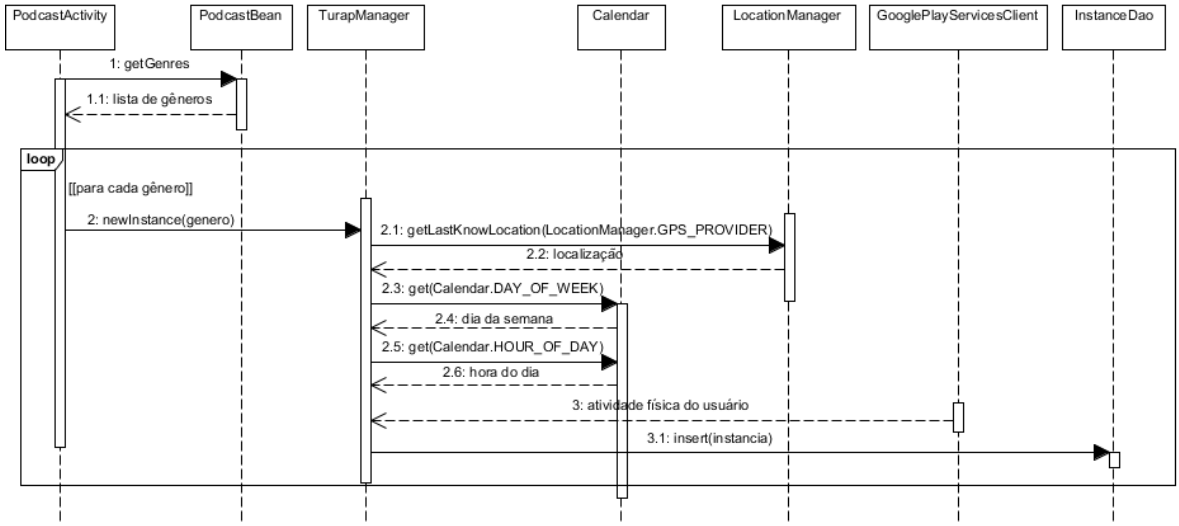
FIGURA 11 – ALGUMAS TELAS DO APLICATIVO CUPOF CAST



Fonte: Elaborado pelo autor

Como ilustra a Figura 12, quando há um clique para ouvir um episódio de um *podcast*, a classe da camada view *PodcastActivity* captura os gêneros do *podcast* contido na classe de entidade *PodcastBean* e envia cada um a classe *TurapManager*. Esse gênero será adicionado juntamente com dados do ambiente vindos das classes *Calendar*, *LocationManager* e *GooglePlayServiceClient*, sendo armazenados no banco de dados logo em seguida pela classe *InstanceDao*.

FIGURA 12 – ARMAZENAMENTO DA INTERAÇÃO DE OUVIR PODCAST



Fonte: Elaborado pelo autor.

Os *broadcasts* contendo os gêneros de *podcast* usado como classificação do contexto são recebidos na aplicação pela classe *TurapClassificationReceiver*, que está configurada no arquivo *AndroidManifest.xml* para a receptação de *broadcasts* de filtro *turap_broadcast_filter*, sendo o mesmo filtro informado a classe *TurapManager* do framework.

Após receber os *broadcasts*, a aplicação irá buscar por *podcasts* que tenham esses gêneros e irá exibí-los na tela principal com intuito de facilitar sua navegação até onde provavelmente quer chegar. Ou seja, a tela principal são todos os podcasts que provavelmente o usuário estará interessado em ouvir no ambiente em que está, como se a aplicação entendesse as vontades de que faz seu uso.

O TURAP foi inserido na aplicação em poucas linhas de código. O TURAP instrumentalizou o desenvolvimento da aplicação com informações e permitiu a liberdade de decidir como usar essa informação ou quais medidas seriam cabíveis para se aprimorar a experiência do usuário, e foi escolhido o aprimoramento da navegação do aplicativo para uma interação mais intuitiva.

7 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

O objetivo do trabalho era criar um framework que fizesse a coleta dos dados de contexto através da Google Services API, e, associar aos dados de um aplicativo teste dentro de uma estrutura de árvore de decisão combinando contexto e preferên-

cias e atuar de forma determinada por esta árvore indicando ao usuário caminhos a tomar. Este objetivo foi cumprido. Porém, existem trabalhos futuros a considerar.

Pode-se citar o aumento de número de aspectos do contexto guardados na base de dados e minerados pelo processo de classificação. Também é desejado que aspectos do usuário como humor e batimento cardíaco sejam armazenados e participem da determinação de padrões na mineração de dados já que são características dos dispositivos móveis de nova geração.

A migração do framework para outras plataformas móveis como Windows Phone e iOS é desejável atingindo assim um maior público alvo de desenvolvedores de aplicativos e uma base ainda maior de usuários.

O algoritmo deve ser otimizado para que o volume original de dados armazenados passe a ser menor sem perder a representatividade e melhorando assim o desempenho do processamento em dispositivos móveis limitados em hardware.

E por fim, devem ser realizados testes e levantamentos estatísticos para mensurar a eficiência da técnica, e identificar possíveis correções.

REFERÊNCIAS

ARHIPPAINEN, Leena; TÄHTI, Marika. Empirical evaluation of user experience in two adaptive mobile application prototypes. In: **Proceedings of the 2nd international conference on mobile and ubiquitous multimedia**. 2003. p. 27-34.

BRAMER, Max. **Principles of data mining**. Springer, 2013.

CAMILO, Cássio Oliveira; SILVA, J. C. Mineração de Dados: Conceitos, tarefas, métodos e ferramentas. **Universidade Federal de Goiás (UFC)**, p. 1-29, 2009.

DOURISH, Paul. What we talk about when we talk about context. **Personal and ubiquitous computing**, v. 8, n. 1, p. 19-30, 2004.

GARRETT, Jesse James. **Elements of User Experience, The: User-Centered Design for the Web and Beyond**. Pearson Education, 2010.

GOOGLE, **Android SDK**. Disponível em: <https://developer.android.com/reference/packages.html>. Acesso em Novembro 2014.

GOOGLE, **Google Play Service**. Disponível em: <https://developer.android.com/google/play-services/index.html>. Acesso em Novembro 2014.

GUALTIERI, Mike. Best Practices In User Experience (UX) Design. 2009.

MENDOZA, Adrian. **Mobile user experience: patterns to make sense of it all**. Newnes, 2013.

ROTO, Virpi. User experience from product creation perspective. In: **Towards a UX Manifesto workshop**. 2007. p. 31-34.

SCHILIT, Bill; ADAMS, Norman; WANT, Roy. Context-aware computing applications. In: **Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on**. IEEE, 1994. p. 85-90.

TEIXEIRA, Fabrício. **Introdução e boas práticas em UX Design**. Casa do Código, 2013.